Losing Control of the Internet: Using the Data Plane to Attack the Control Plane

Max Schuchard
University of Minnesota
schuch @ cs.umn.edu

Eugene Y. Vasserman
Kansas State University
eyv @ ksu.edu

Abedelaziz Mohaisen
University of Minnesota
mohaisen @ cs.umn.edu

Denis Foo Kune
University of Minnesota
foo @ cs.umn.edu

Nicholas Hopper
University of Minnesota
hopper @ cs.umn.edu

Yongdae Kim
University of Minnesota
kyd @ cs.umn.edu

Abstract

In this work, we introduce the Coordinated Cross Plane Session Termination, or CXPST, attack, a distributed denial of service attack that attacks the control plane of the Internet. CXPST extends previous work that demonstrates a vulnerability in routers that allows an adversary to disconnect a pair of routers using only data plane traffic. By carefully choosing BGP sessions to terminate, CXPST generates a surge of BGP updates that are seen by nearly all core routers on the Internet. This surge of updates surpasses the computational capacity of affected routers, crippling their ability to make routing decisions.

In this paper we show how an adversary can attack multiple BGP sessions simultaneously and measure the impact these session failures have on the control plane of the Internet. We directly simulate the BGP activity resulting from this attack and compute the impact those messages have on router processing loads. Through simulations we show that botnets on the order of $250,000$ *nodes can increase processing delays from orders of microseconds to orders of hours.*

We also propose and validate a defense against CXPST. Through simulation we demonstrate that current defenses are insufficient to stop CXPST. We propose an alternative, low cost, defense that is successful against CXPST, even if only the top 10% of Autonomous Systems by degree deploy it. Additionally, we consider more long term defenses that stop not only CXPST, but similar attacks as well.

1 Introduction

The Internet can be divided into two distinct parts; the *data plane*, which forwards packets to their destination, and the *control plane*, which determines the path to any given destination. The control plane is designed to route around connectivity outages, resulting in the Internet's robustness to localized failure. This durability comes with a cost however: "local" events can have nearly global impact on the control plane. An excess of such control plane events can disrupt even core Internet routers. This disruption can lead to network instability, resulting in a loss of connectivity and data. There are several historical examples of such incidents stemming from rare events, such as router misconfiguration, hardware failure, and as side-effects of a fast-propagating worm.

In this work, we introduce the Coordinated Cross Plane Session Termination, or CXPST, attack, a new form of distributed denial of service (DDoS) attack that attempts to exploit the global scope of BGP updates to induce control plane instability on the Internet as a whole. In order to artificially create control plane instability, CXPST applies Zhang et al.'s [74] work on disrupting BGP sessions between routers. Zhang et al. described how an unprivileged adversary in control of a botnet can exploit the fact that the control plane and data plane use the same physical medium; from here on we will refer to this as the ZMW attack. This fate-sharing allows an adversary to convince a BGP speaker that one of its BGP sessions has failed. CXPST computes centrality measures of the network topology and uses this information to intelligently select a collection of BGP sessions to disrupt using the ZMW attack. This results in waves of control plane instability which, because of the choice of links, are broadcast globally. By exerting influence over the location and times of failures, CXPST generates enough updates to overwhelm the computational capacity of routers, crippling the Internet's control plane.

Unlike Coremelt [62], another Internet-scale DDoS attack, CXPST does not directly attack *all* links on the Internet. Instead, CXPST will only attack a small subset of links, using the properties of these links to amplify the at-

tack. This reduces the bandwidth required to successfully launch CXPST compared to Coremelt. We show through simulation that botnets on the order of $250,000$ members can cause severe disruption to the Internet control plane, even under conservative estimates of adversarial bandwidth and router over-provisioning. The resources to launch this attack are now widely available, due to the explosion in the number of end-user machines that have been compromised by a centrally controlled virus or worm. These botnets provide access to massive amounts of distributed computing power and aggregate bandwidth of several terabits per second [61, 36].

In this work we demonstrate an adversary's ability to successfully attack specific BGP sessions. In fact, our adversary running CXPST was able to disrupt more than 98% of targeted BGP sessions. We then measure the control plane instability seen in core routers resulting from BGP's natural reaction to these failed sessions. Lastly, we examine the impact of this instability by looking at the time it takes routers to make decisions. We show that core routers will experience processing delays on the order of minutes rather then microseconds after a few minutes of an adversary launching CXPST. In fact, after 20 minutes of attack, core routers experience delays of more then 100 minutes.

We also consider defenses against CXPST. We demonstrate that currently-deployed mechanisms to combat control plane instability would be ineffective against CXPST. Instead of attempting to stop BGP from broadcasting updates globally, we focus our defenses on preventing an adversary from disrupting BGP sessions. We present a short term, easily implementable solution that successfully prevents CXPST, even when only partially deployed. We also discuss long term defenses, redesigning routers or fundamentally altering the way control traffic is exchanged.

The contributions of this paper are as follows. First, we demonstrate how to extend the ZMW attack to effectively target a system of routers. We demonstrate through simulation that an attacker can attack multiple BGP sessions across the Internet simultaneously. We quantify the effect that the failure of these BGP sessions has on the Internet as a whole in terms of both BGP update messages and processing times. We examine currently existing defenses, discovering that they have little impact on our attack. Lastly, we propose both short term and long term defenses that will be successful in stopping our attack.

The remainder of the paper is organized as follows. First we provide relevant background information on BGP, the nature of control plane instability, and the ZMW attack. We then discuss CXPST itself in Section 3. Specifically we cover how an adversary selects links to attack and manages bots during the course of the attack. In Section 4 we then describe the simulator we used to experiment with CXPST, and present the results of those simulations. We demonstrate in Section 5 how deployed defenses fail to reduce the impact of CXPST. We also present a short term solution that stops CXPST, even when only partially deployed. We move on to discuss related works including both similar attacks and denial of service defenses in Section 6. We wrap up with Section 7, a discussion of why denial of service defenses do not affect CXPST and what would be required of long term defenses.

2 Background

2.1 Inter-Domain Routing and BGP

The Internet is composed of multiple networks called autonomous systems (ASes), which relay traffic to each other on behalf of their customers. ASes are diverse, with a wide range of sizes and numbers of connections to other ASes. Some ASes have very high degrees of connectivity; these ASes are considered *core* ASes. Other ASes have very low degrees of connectivity, sitting at the outskirts of the Internet; these are *fringe* ASes. Fringe ASes require the assistance of core ASes in order to route traffic. These core ASes, which agree to forward traffic to and from other ASes, are termed *transit* ASes. Routers must collectively determine what paths, or what series of ASes, packets have to travel through to reach their destination. To this end, routers exchange messages advertising their ability to reach networks via a routing protocol.

The Border Gateway Protocol (BGP) [32] is the *de facto* standard routing protocol spoken by routers connecting different ASes. BGP is a path vector routing algorithm, allowing routers to maintain a table of AS paths to every destination. BGP also uses policies to preferentially use certain AS paths in favor of others. For simplicity, we will refer to these routers as border routers, since they are located at connection points between their home AS and another AS. Each border router has its own private routing table, and therefore its own different view of the network. When one router in an AS changes its routing table due to events such as link failures, it recomputes its routing table, removes the failed link, and informs its neighboring ASes of the change via a BGP update message. This change might trigger the same series of events in other border routers.

The BGP specification [32] defines route selection and also enumerates a number of constraints on sessions between two BGP speakers. For example, the standard defines how speakers should determine if a peer is no longer functional, how to keep BGP sessions alive during periods of inactivity, and how to handle errors. As per the standard, a router will consider a BGP peering session as failed if there is a failure of the underlying TCP stream, if the router receives an error or malformed message from the peer, if the

peer explicitly closes the connection, or if too much time passes between incoming BGP messages.[1]

2.2 BGP Stability and Network Performance

In normal BGP operation the network converges to a stable state. However, local changes such as cable cuts, router hardware failure, or changes in local BGP policy can result in routes having to be withdrawn, leading to routing table recalculation and re-advertisements to other routers. The catch is, these advertisements can lead to the same activities on the routers that receive them. This includes advertisements to yet more routers who will repeat this process, possibly causing the update to be propagated globally. This behavior demonstrates a key fact: in BGP small local changes are often seen globally.

Instability in the control plane can reduce the performance of the data plane [60, 66, 23]. When a router is shut down, paths that pass through that router will no longer function, and new routes must be found. Functioning routers will continue forwarding traffic towards the now non-existent router until they complete the process of finding a new route. All traffic directed toward the powered down router will be lost, resulting in large volumes of dropped packets. This is just one example of how instability can result in large disruptions of the data plane.

When a set of routes oscillates rapidly between being available and unavailable it is termed *route flapping*. Route flapping can be the result of several flaws in the network, including misconfiguration, faulty router hardware, and link failures. It is a problem because of the sheer number of control plane messages generated, and the resulting routing table re-computations that routers must perform. Data plane performance is only restored after affected routers complete the processing of BGP messages. In the case of large amounts of instability, route re-computation increases the load on a router's CPU dramatically, potentially exceeding its capacity. This increased load translates into a longer turnaround time for processing decisions, which in turn extends the duration of the data plane disruption. During route flapping, routes need to be recalculated as quickly as possible, but the fact that so many routes need to be recalculated slows that computation.

Some functionality exists currently that attempts to mitigate the damage route flapping does to both the data and control planes. Minimum Route Advertisement Intervals (MRAI) [32] prevent a series of rapid advertisements of route changes for the same network. While MRAIs do not help the data plane recover directly, they do reduce the load on a router's CPU. BGP Graceful Restart [33] provides a grace period where two connected routers allow their data planes to continue functioning even while there is an issue on their control plane.[2] This attempts to mitigate issues where two routers need to recover from a simple error. Lastly there is Route Flap Damping [31], which directly aims to combat route flapping by suppressing (ignoring) routes that exhibit flapping behavior. The initial work to route around failures still needs to be done, but additional work is not done as the link oscillates between functional and non-functional states.

2.3 Attacks on BGP Routers

Given the importance of routers and routing protocols, it is unsurprising that there exists a large body of literature exploring their weaknesses. Of particular interest to this work is a paper by Zhang, Mao, and Wang [74] that looks at using brief targeted data plane congestion to trick a pair of routers into disconnecting from each other. In their attack, an unprivileged adversary indirectly interacts with the control plane via the data plane. This is possible because the data plane and the control plane are co-located. Because of this co-location, congestion from data plane traffic can cause the loss of control plane traffic. There are several places inside a router where control plane traffic and data plane traffic contend for resources, including buffer space and bandwidth. When resources are scarce, control traffic and data traffic must share these limited resources.

The BGP protocol (see Section 2.1) uses hold timers as one way to detect a failed session. Routers keep track of the last time they received control plane data from a BGP peer, and, if this time exceeds the hold timer, the session is torn down. If enough consecutive control plane packets are lost, the hold timer of a BGP session will expire and the session will fail. In essence an adversary can use a flood of data to digitally "cut the link" between two routers. When the BGP session fails, all routes discovered via that session will have to be withdrawn and new routes recalculated on both sides of the "failed" link. Zhang et al. demonstrated in both hardware and software routers the ability to successfully implement this attack.

3 The CXPST Attack

In this section we present CXPST, an attack against the Internet's control plane. In CXPST, an adversary in control of a botnet selectively disrupts BGP sessions in an effort to artificially generate a large number of BGP updates. This surge of updates overwhelms the computational capacity of routers, preventing them from efficiently making routing decisions.

[1] Note that this is a protocol-level timer, distinct from the TCP keepalive timer.

[2] The alternative is to immediately withdraw routes learned from the failed router and advertise new routes.

3.1 Attacker Model

There have been many instances of adversaries causing control plane instability by intentionally misconfiguring routers under their control [11, 8, 52, 14]. These attackers were able to to interact with the control plane directly using their privileged status as BGP speakers. Attacks at this level can be typically prevented with the use of BGPSEC or similar technologies [37, 65, 64].

In this work we instead consider an unprivileged adversary who does not control any BGP speakers, and consequently can only create data plane traffic. Lacking the ability to directly generate control plane messages, these adversaries instead need to force non-colluding routers to generate control plane events. We specifically consider an adversary who controls a botnet of reasonable size. This attacker is capable of generating network traffic from compromised hosts distributed across the Internet. Adversaries in control of compromised BGP speakers would be capable of generating some of the phenomena used to drive CXPST, but would be unable to do so at arbitrary locations in the network.

3.2 CXPST Conceptually

In order to create control plane instability, our attacker will apply the ZMW attack [74]. As discussed in Section 2.3, ZMW uses data traffic to trick a pair of routers into disconnecting from each other. This results in a set of route withdrawals, recalculations, and advertisements. Interestingly, the control plane disruption generated is not limited to the one set of withdrawals and advertisements. Since the targeted link is no longer used by routes after the BGP session fails, no traffic will utilize the link. This allows the two attacked routers to communicate with each other once more, as the link will no longer be congested with attack traffic. The targeted routers will, after a small amount of time, re-establish their BGP session. This will result in further BGP updates as the routes that were just withdrawn are re-advertised. Bot traffic will once again shift to the targeted link as the previous routes become utilized once more, and the attack resumes without any intervention from the attacker. The targeted BGP session will again be destroyed and the cycle repeats itself, forcing the targeted links to oscillate between "up" and "down" states. In essence, CXPST induces targeted route flapping.

While the two routers attacked will be most impacted, routers not directly attacked will be affected as well. As mentioned in Section 2.2, BGP updates that result from local changes tend to be broadcast on a global scale. By creating a series of localized failures that have near global impact, CXPST has the potential to overwhelm the computational capacity of a large set of routers on the Internet.

There are three key tasks that CXPST needs to accomplish in order to function. First, the correct BGP sessions must be selected for attack. These BGP sessions must be selected to maximize control plane instability when they fail. If an insufficient number of BGP updates are generated, then routers will not be computationally exhausted, and the attack will not succeed. Second, the attacker needs to direct the traffic of his botnet onto the targeted links. While Zhang et al touch on this in their work, they do not deal with the difficulties in managing attack traffic on a dynamic network. For example, congestion on links used to approach targeted links must be minimized. Link failures on the way to the target will prevent attack traffic from reaching its destination, possibly preventing the termination of the targeted BGP session. Lastly, since CXPST is essentially route flapping, the attacker must find a way to minimize the impact of existing mechanisms that attempt to mitigate the effects of route flapping.

3.3 Selecting Targets

Maximizing control plane disruption is equivalent to maximizing the number of BGP update messages that are generated as a result of link failures. Centrality measures from graph theory provide a good starting point for building a heuristic to govern target selection. Our method of selection uses a slightly modified version of edge betweenness as a metric. Normally edge betweenness is defined as:

$$C_B(e) = \sum_{s \neq t \in V} \frac{\sigma_{st}(e)}{\sigma_{st}} \tag{1}$$

where σ_{st} is the number of shortest paths between nodes s and t, and $\sigma_{st}(e)$ is the number of those paths that contain the edge e. BGP does not always use the shortest path between two ASes however. Because of this we use a modified definition of edge betweenness:

$$C_B(e) = \sum_{s \neq t \in V} path_{st}(e) \tag{2}$$

where $path_{st}(e)$ is the number of BGP paths between IP blocks in s and t that use link e. Since each of these routes must be individually withdrawn, recomputed, and readvertised this will provide an approximation of the number of BGP messages generated if the link were to fail. Consequently, target links are ranked in order of their "BGP Betweenness".

Another reason to use BGP betweenness is that our attacker possesses the resources to measure it. As stated in Section 3.1 our attacker controls a botnet distributed across the Internet, this provides him with a large number of distinct vantage points. Prior to the attack, bots can perform

traceroutes from themselves to a large set of nodes in separate networks and report the results. By aggregating the results an attacker can generate a rough measure of the BGP betweenness of links. Each time we see an edge in our aggregated traceroute data set, it represents an individual route that crosses a given link. This is because each traceroute originates from a distinct source and travels to a distinct destination.

Equal cost multi-path routing, or ECMP, presents an additional issue for CXPST. It requires markedly more resources to congest a link when it is part of a set of load balanced links then when it is a stand alone link. In order to avoid this, when traces are being gathered, multiple traces need to be taken. These traces can be compared in an effort to detect ECMP. Any links that are possibly using it are removed from the set of potential targets. Recent studies [6] have shown that load balancing, while prevalent inside ASes, is not widely used between ASes. This is a best case scenario for our attacker. Load balancing inside ASes removes potential bottlenecks for attack traffic, and since CXPST only attacks links between ASes, few targets will be excluded.

3.4 Attack Traffic Management

At first glance, selecting which bots will attack a given link appears straightforward. As discussed by Zhang et al. [74], an attacker could simply use all bots that can find some destination such that the path to the destination crosses the targeted link. This method suffers from two main weaknesses. First, this strategy fails to take into account the fact that network topology is dynamic. This issue is especially important in the case of CXPST as the attack forcibly changes network topology in multiple places. Second, there is the possibility that we will saturate bandwidth capacity on the way to the target link. This can result in the unintentional termination of BGP sessions, cutting off our path to the target.

3.4.1 Dealing With Changing Topology

CXPST actively changes network topology. The attacker must select which bots will attempt to attack a given link with this in mind. Instead of simply checking that a given path contains the target link, the attacker must ensure that the path does not contain other links that are being targeted as well. By doing this, when links targeted by CXPST fail, attack traffic will not be re-routed.

Attack traffic can still be re-routed because of the unintended disruption of a non-targeted link. In order to counter this, an attacker should send more attack traffic toward a targeted link then is needed to congest it. This "safety net" will allow some amount of attack traffic to be diverted because of network dynamics without relaxing pressure on targeted links.

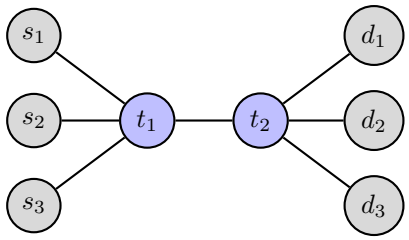

Figure 1: An illustration of attack traffic aggregating. $s_1 \ldots s_3$ are source ASes, $d_1 \ldots d_3$ are destination ASes, and t_1 and t_2 are targeted routers.

3.4.2 Fixing the Flow Issue

Our attacker will typically have more bots able to attack a given link than needed. Care must be taken when selecting a subset of these bots to attack the link. In order to minimize the amount of congestion prior to reaching the targeted link, the attacker should keep the attack traffic dispersed until it reaches the target. When the attack traffic reaches the targeted link the attack flows will be aggregated together, causing congestion on that link. After the intersection point traffic takes different paths toward its final destinations, dispersing in an effort to not congest downstream links. This is shown in Figure 1, where the link between t_1 and t_2 is the targeted link. Traffic approaches from a variety of sources, heading to a variety of destinations. Traffic levels on links before and after the target are not substantial. For example the link between s_1 and t_1 or the link between t_2 and d_2, are manageable, but the aggregation of flows across the t_1 to t_2 link creates congestion.

CXPST uses a straight-forward algorithm to automate attacker assignment. Prior to allocating resources, our attacker builds two flow networks based on the traceroutes used to select targets. In one network, bots are treated as sources and target links are treated as sinks. In the other, target links are treated as sources and destination networks are treated as sinks. The attacker can either guess the bandwidth of links involved or actively measure their capacity. When selecting destinations for attack traffic, the attacker runs a max flow algorithm on the first flow network, establishing which bots will be used to attack each targeted link. Then the second flow network is then analyzed to determine which destination networks attackers should address their traffic to. Where possible bots will attempt to send attack traffic to IP address of other bots in the botnet as described by Sunder and Perrig in Coremelt [62]. In this way, traffic sent by the attacker is "wanted" and not reported by end hosts.

3.5 Thwarting Defenses

As was mentioned in Section 2.2 there are some mechanisms that exist to reduce the effects of route flapping. Since CXPST is artificially induced route flapping, these defenses might impede it. These defenses though, were designed to deal with random network events, not an adaptive adversary. Two of the defenses, BGP Graceful Restart and Minimum Route Advertisement Intervals, require no changes. Route Damping on the other hand requires some minimal changes to CXPST's behavior. During the course of the attack the bots will need to remove links that get damped from their target set. Bots notice that links are being damped when the paths used to reach their targets do not re-appear within a time window. New target links are then chosen from the list of available targets. We will demonstrate CXPST's ability to function in the presence of these defenses in Section 5.1.

4 Simulation

There are a large number of questions to be asked of CXPST. Will real world bots be in a position to send traffic over a given link? Will bots over-saturate the edges of the network before reaching their target? How many BGP updates would CXPST be able to generate? Would the rate of these updates be sustainable over the duration of the attack? What would the impact of these updates be on routers?

In order to answer these questions we built a discrete event driven simulator modeling the dynamics of routers on the Internet. Given the level of complexity found in the system that we were attempting to model, this presented a challenge. Many diverse agents needed to be represented including: ASes, routing polices, the routers themselves, the physical links that connect these routers, and the botnet used by our attacker.

4.1 Simulator Design

In this section we discuss some of the design choices made in our simulator. The Internet is a complex system, and simulating it requires trade offs between simulation fidelity and efficiency. We discuss some of the simplifying assumptions we made on topology used in our simulator. Additionally, we define the bandwidth model used by links and the distribution of bots in our simulated botnet. Further details of the simulation, including the source code and configuration files are available online [55].

4.1.1 Network Topology

The internal topology of an AS is usually a closely held secret. Since many of the questions we would like to answer are dependent on network topology, this is an issue. Papers exist that attempt to infer internal network topology, for example RocketFuel [59]. However, even if we had a perfect view of Internet topology, efficiency concerns would prevent the full topology from being used for simulation. Given these facts, we elected to use a simpler view of Internet topology which allowed for accurate simulation.

We started building our simulator's topology by examining the wealth of data on the AS-level topology of the Internet made available from CAIDA [15]. Simulator scaling again ruled out using the complete AS topology. The fact that CXPST targets transit providers served as a guide in selecting a sub-graph of the full AS level graph. Using inferred AS relationships from January 2010, we built a set of ASes containing all ASes that provide service to other provider ASes, i.e. all the ASes who had at least one customer that itself had customers. A graph was then generated containing these ASes and any edge that existed between ASes in the subset. The result was a connected graph with 1829 ASes and nearly $13,000$ edges.

As mentioned previously, each AS is a diverse network in and of itself. Since we are only interested in the behavior of edge routers speaking BGP, we can largely ignore internal routing dynamics. Route reflectors are the one exception to this rule. When a BGP update is received from a different AS, the receiving router hands the update to a route reflector, which broadcasts the update to all BGP speakers in an AS. Each of these BGP speakers will process the update independently. This means that each edge router in an AS will deal with a BGP update regardless of which actual router in the AS first received it. This allows us to model the behavior of edge routers in an AS by maintaining a lone "representative" router for each AS.

While we recognize that the AS level topology does not represent the actual physical topology, we make a key assertion about their relationship: if there is an edge in the AS level topology, there must be at least one link on the physical topology. In reality there are three possible scenarios for an inter-AS connection. First, there might indeed only be one link. Second, there might be more then one link, but only one is actively used for traffic, or at least traffic originating for a given area. Third, there are multiple links and they are all actively used.

Since CXPST only attacks individual links (compared to other DDoS attacks, for example Coremelt [62], which target *all* links in an AS) we elect to represent edges in our topology by a single link. This is accurate in the first two scenarios previously mentioned (when a single links serves a geographical area bots can be selected from just that area in order to target it). In the third case, multiple active links, our assumption would be inaccurate, but as stated in Section 3.3, our attack actively avoids load balanced links. The fact that we don't need to simulate attacking these links, coupled with the previously mentioned fact that these links

are uncommon [6], means that our simplification is acceptable.

The bandwidth model for links in our simulator is meant to be as disadvantageous to the attacker as possible. Link capacities are based on the degrees of the connected ASes. Since we are concerned about the ability to fill core AS links we use OC-768 size links, the largest link size currently in the SONET standard, for those links. In the same spirit we connect all fringe ASes, where the majority of the attacker's resources reside, with OC-3 links. It is important to mention that while the aggregate bandwidth between two ASes may be much higher than a single OC-768 link, we are only concerned with attacking *single inter-AS links*, meaning that having to attack an OC-768 link is truly a worst case scenario for an attacker.

4.1.2 The Botnet

Along with topology, bot placement also impacts simulation results. Recent papers on botnet enumeration have given us some insight into the distribution of bots throughout the Internet, allowing us to use a real bot distribution in our simulator. We used the data set for the Waledac botnet [56] to build our model of bot distributions. IP addresses of infected machines were mapped to their parent ASes using the GeoIP database [47], providing a rough count of infections per AS. We then uniformly scaled these numbers up or down to achieve the botnet size desired. To ensure a proper lower bound for attacker bandwidth, bots were given a basic ADSL connections with an upload capacity capped at 1.0 Mbit/sec [1]. Bots were only given the ability to send network traffic and perform traceroutes. They were *not* given any additional information about the network, such as link capacities or AS relationships.

4.2 Simulation Methodology

Our event driven simulator allows us to view the results of a botnet executing CXPST. At the beginning of a simulation, routers are allowed to connect to their BGP peers and reach a stable network state. Simulated routers run BGP using policies guided by inferred AS relationships [15] and no valley routing policies [28]. They have simulated computational capacity in keeping with benchmarking studies [70]. After the network has reached a stable state, bots are allowed to interact with the network. Bots only have the ability to run traceroutes and to send network traffic. The ability of bots to send traffic is limited by the bandwidth of links carrying the traffic. All routers in the simulation are vulnerable to the ZMW attack, meaning that accidental disruption of BGP sessions in the simulation is possible. This may lead to traffic redirection away from targeted links.

The impact of CXPST needs to be evaluated in three places. First, we must answer the question of how bot placement and bandwidth bottlenecks affect the ability to attack specific links. We can compare the number of targeted and un-targeted BGP sessions that are disrupted during the course of the attack. This will give us a grasp of the feasibility of attacking specific BGP sessions.

Next, we examine the effect of these disrupted BGP sessions. Apart from topology changes generated by CXPST, the topology of our simulator is stable during the course of the simulation. This means that any updates seen are a direct result of the attack. Our simulator logs the arrival of BGP messages to routers, giving us a record of the number of BGP updates generated by CXPST. We can compare the number of update messages generated to normal loads providing us with a measure of CXPST's success.

Lastly, we would like some idea of the impact any dramatic increases in BGP update rates would have on the routers themselves. One measurable effect is the increase in the time between when a router receives a BGP update message and when it is finally processed. If the time to process an update becomes large, the data plane suffers dramatically, as local outages are not reacted to and traffic is sent to dead links. Using the logs of BGP update message arrivals and a benchmarking study by Wu et al. [70] we can build an estimate of the time to process BGP updates during an attack.

4.3 Simulation Results

CXPST was simulated with botnets of 64, 125, 250, and 500 thousand nodes. We describe the results of these simulations in this section. In general the majority of our testing focused on the 250 thousand node botnet scenario. Diminishing returns from increasing botnet size drove this decision.

4.3.1 Success in Disrupting BGP Sessions

We can examine our ability to successfully disrupt only targeted BGP sessions by placing them into buckets. We chose three different descriptors for links: targeted links, last mile links, and transit links. Any link selected for disruption by CXPST is considered a targeted link. Last mile links are un-targeted links that connect fringe ASes to the rest of the network. Any link that does not fit the other two categories is considered a transit link. Our attacker's goal is to maximize the number of targeted links that fail while minimizing the number of failures in the other two categories. As mentioned in Section 3.4.1, CXPST sends more attack traffic to a link then is needed. This "safety net" allows for the disruption of some attack traffic without degrading the attack. In our simulation the "safety net" was an extra 30% over the estimated required traffic.

The results of a $250,000$-node attacker can be seen in Figure 2. Our simulated attacker successfully disrupts more

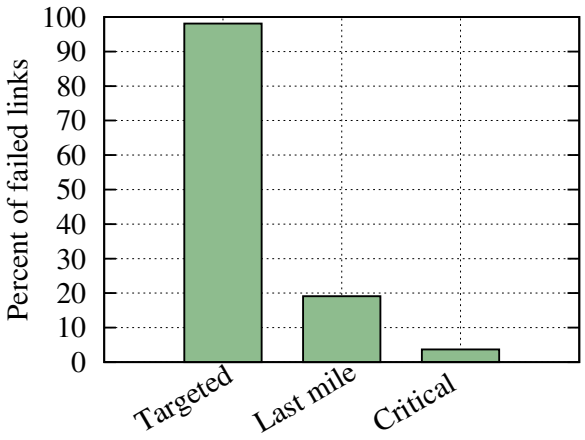

Figure 2: Percentage of BGP sessions for various types of links that failed when a botnet of 250 thousand bots launched CXPST. Note that a 30% "safety buffer" was used, so that up to roughly 30% of last mile links could fail without impeding CXPST.

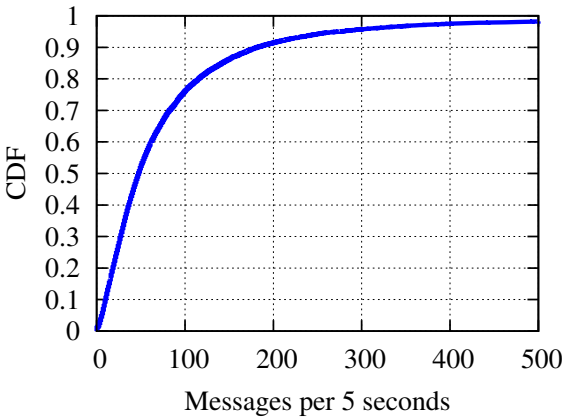

Figure 3: A CDF of normal BGP update loads for a RouteViews router during January 2010. Message load is measured as the number of BGP update messages that arrive in a per 5-second interval.

then 98% of targeted links during the course of the attack. The attacker only disrupts roughly 19% of last mile links at some point in the attack, far less then the 30% that is tolerable with the safety net. Most importantly, less then 4% of transit links are disrupted during the attack. This demonstrates the ability of CXPST to surgically disrupt BGP sessions.

4.3.2 BGP Update Generation

Next we will show the number of BGP updates that are the direct result of CXPST. We were most concerned with the impact on *core routers* in our topology, the top 10% of ASes by degree. Emphasis was placed on core routers because of the potential impact on the rest of the network. These ASes are utilized by the majority of other ASes as transit providers, and instability in these routers would be felt across the Internet.

Using simulation logs, we gathered information on the number of BGP updates routers receive during 5 second windows of time. We turned to the RouteViews data set [54] to get an idea of baseline router load. In excess of 23,000 network operators voluntarily start BGP sessions with RouteViews routers in order to validate their network configuration from an outside vantage point. RouteViews routers keep a log of the real time arrival of BGP update messages from these sessions. We used logs from January 2010, the same month as our AS relationship data, to build a view BGP update load. Figure 3 shows a CDF of the number of messages a RouteViews routers see per 5 seconds window. It is important to note that RouteViews routers have an inordinate number of BGP sessions relative to edge routers in transit ASes, meaning that this number of updates is more then likely an overestimation of the number of messages seen by a BGP speaker, and consequently will result in an *underestimation* of our attack's effectiveness.

As was mentioned in Section 2.2, large bursts of updates have a significant impact on the performance of the Internet. Simulations show that CXPST successfully creates BGP update message bursts throughout the duration of the attack. For example, during normal operation (see Figure 3), the 90th percentile load is 182 messages per 5 seconds. During CXPST the 90th percentile load is dramatically increased for the targeted routers, a CDF of their 90th percentile loads is shown in Figure 4(c). In the case of the $250,000$-node attacker, more than half of core routers are at or above a four order of magnitude increase in load. These bursts of updates are not a few isolated incidents. At the 75th percentile of update load, shown Figure 4(b), we continue to see the same dramatic increases in processing load.

Moreover, these spikes are not the only effect of CXPST, an increase in BGP update rate is felt throughout the attack. Figure 4(a) shows the increase in the median load of routers during the attack. In the case of the $250,000$-node botnet, the median load on nearly half of the core routers increased by a factor of 800 or more. Even using the $125,000$-node botnet results in 50% of routers' median loads increased by a factor of 400 or more. This increased median load shows that routers will not have a chance to recover from the previous bursts of updates.

To give some specific examples, Figure 5 plots distributions of message loads for several large ISPs during the simulated attack by $250,000$ bots. The distribution of load under normal conditions from Figure 3 is included as a point of reference.

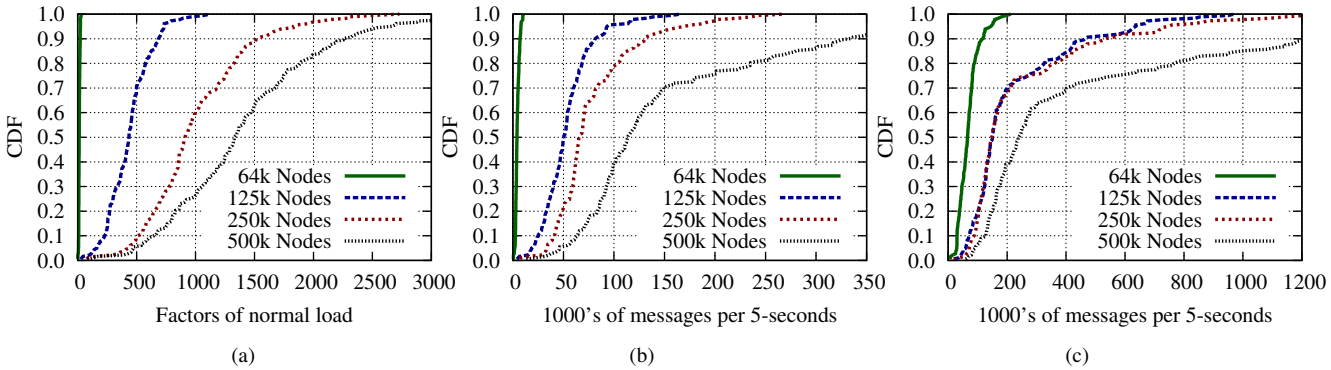

Figure 4: Median router load of targeted routers under attack as a factor of normal load (a); and 75th percentile (b) and 90th percentile (c) of message loads experienced by routers under attack, measured in BGP updates seen in 5-second windows.

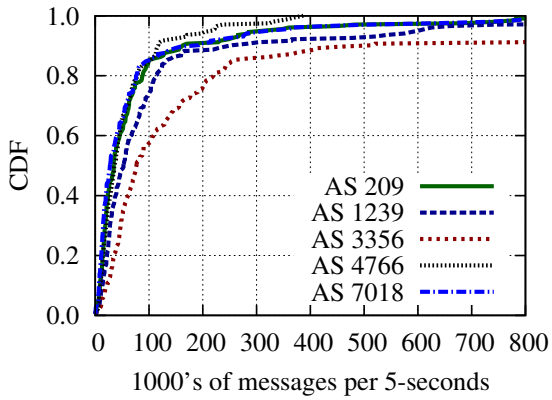

Figure 5: Update messages received during 5-second windows for a collection of specific AS under attack by 250, 000 bots.

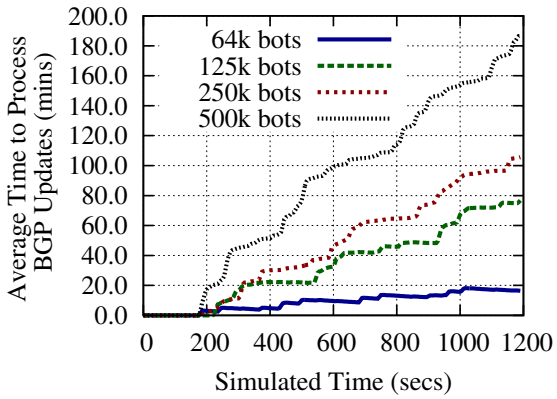

Figure 6: The average time to process a BGP update for core ASes under attack by botnets of various sizes. Attack traffic starts at time 0, the first link failures occur at time 180.

4.3.3 Time to Process Updates

The end results of CXPST can be seen by examining the time required to process a BGP update message. Routers process BGP update messages at a roughly constant rate. If the rate they are received at surpasses the rate of computation, messages will need to be buffered, and processing delays will occur. Using performance figures from a router benchmarking study [70] we computed the delay between when core routers received BGP updates and when they finally finished processing those updates while under attack.

We term the average delay between when a BGP update arrives and when it completes being processed the time-to-process or TTP.[3] The TTP for core ASes under attack by various sizes of botnets is graphed in Figure 6. CXPST successfully triggers the first BGP session failures 180 seconds into the attack. From this point onward the average TTP for updates arriving to core ASes increases dramatically. For example, in the case of a 250, 000 node attacker, after 10 minutes of attack the backlog of updates is large enough to delay processing for roughly 45 minutes. Once 20 minutes of attack time have passed the wait has increased by an additional hour, to just over 100 minutes. The reason for this constant increase in TTP was discussed in Section 2.2. Routers under this amount of computational load are resource exhausted, and can only recover if they are receiving update messages at a low rate. However, updates are nearly constantly arriving as a result of CXPST. This means that the affected routers are never given a chance to recover.

Anecdotal evidence suggests that routers placed in resource constrained states behave unstably [17]. It is not outside the realm of possibility that, when confronted with update queues thousands of messages long and processing delays measured in minutes rather then microseconds, that routers will exhibit undefined behavior. This undefined behavior adds a new dynamic to the system. We leave study of this for future work.

[3] This is also known as makespan.

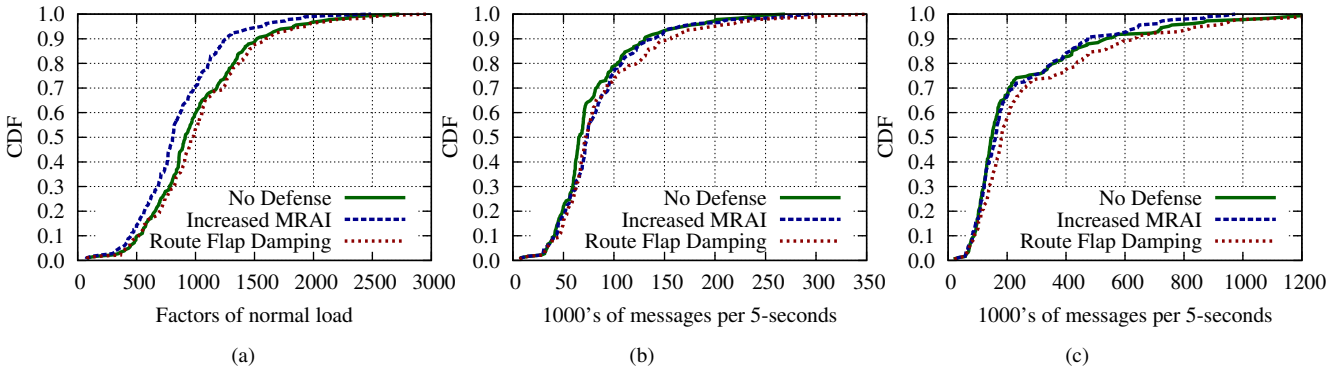

Figure 7: Median router load of targeted routers using defensive measures as a factor of normal load ((a)), 75th percentile ((b)), and 90th percentile ((c)) of message loads experienced by routers using defensive measures, while under attack by $250,000$ bots, measured in BGP updates seen in 5-second windows.

5 Toward Defenses

Given the potential consequences of an adversary carrying out CXPST, building a defense against it is of paramount importance. Since CXPST is route flapping on a grand scale, mechanisms that mitigate the damage done by route flapping might prove successful. We will examine these technologies and demonstrate that they do not have an effect on CXPST. We then focus on stopping CXPST before it has an opportunity to generate update messages rather then trying to change BGP's tendency to broadcast updates globally. We do this by proposing a simple configuration based solution to prevent Zhang et al.'s attack. Our solution has the advantage of being easily deployable and effective, even if only partially deployed.

5.1 Deployed Defensive Measures

As discussed in Section 2.2 there are a handful of currently deployed mechanisms to reduce the number of updates generated by route flapping. We ran a set of simulations using a $250,000$-node attacker in an effort to evaluate the effect of these defenses on CXPST.

In our experiments, BGP Graceful Restart [33] did not have a significant effect on the behavior of the network. BGP graceful restart is meant to provide a grace period to the data plane when a BGP session fails between two routers. Because our attack traffic travels on the data plane, it benefits from this grace period, allowing CXPST to continue stressing the link until the grace period expires. When this happens the resulting situation is the same as the one that occurs when BGP Graceful Restart is not used.

The other two defensive measures had nearly as limited an effect. A comparison of CXPST, CXPST run in a system with increased minimum router advertisement intervals, and CXPST run in a system with globally deployed route flap damping can be seen in Figure 7. As can be seen in these graphs, the defenses, as predicted in section 3.5, do not have significant impact on CXPST.

5.2 Stopping Session Failure

Instead of attempting to limit the scale and number of updates that result from CXPST, our proposed defense focuses on stopping CXPST before it can generate updates. Our defense against CXPST is simple, remove the mechanism that allows Zhang et al.'s attack to function. Sadly, accomplishing this is easier said then done. Creating a differentiated service class for control plane traffic, if done correctly, could solve this issue. The issue with this is that existing routers are incapable of correctly providing this "perfect service" class. We discuss this more in Section 7.4.

One simplistic way to stop the ZMW attack is disabling hold timer functionality in routers, something easily achievable by setting the timer to an exceedingly large value. By doing this, BGP sessions will not be terminated by high amounts of data plane traffic. This can be achieved with a simple change to configuration files, making its cost nonexistent, but it is unclear if modern network monitoring is nimble enough to correctly assume the responsibilities of hold timers. Nevertheless this solution is illustrative of any mechanism that protects BGP session failure from data traffic.

It is unlikely that any solution to the ZMW would be globally deployed. For example, not all network operators possess the same level of network monitoring, and most will be unwilling to remove hold timers. In order to test if our defense is incrementally deployable, we simulated a $250,000$-node botnet running CXPST against a network that had fractional deployment of our solution. We selected the largest ASes by degree to implement our solution. The results of these tests can be seen in Figure 8 and Figure 9.

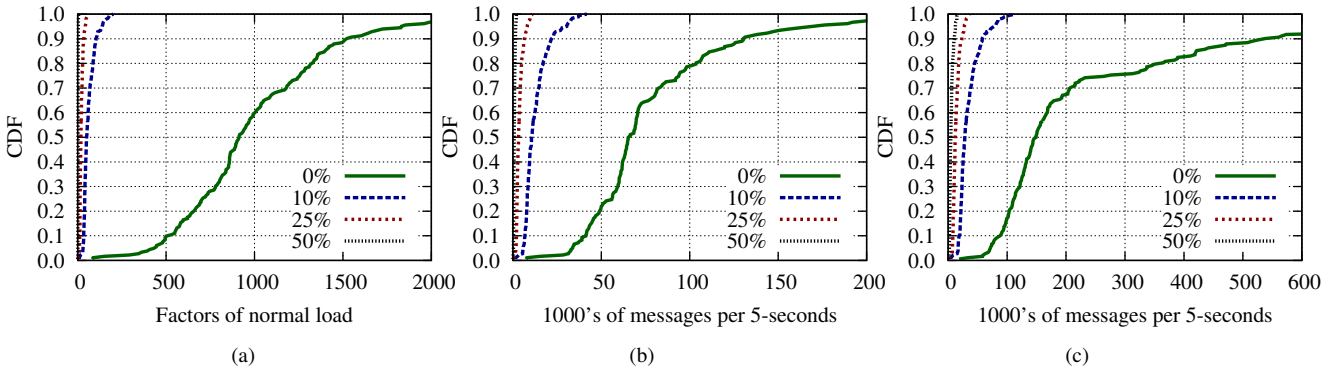

Figure 8: Median router load of targeted routers with the removal of hold timers by some routers as a factor of normal load ((a)), 75th percentile ((b)), and 90th percentile ((c)) of message loads experienced by routers with the removal of hold timers, while under attack by 250, 000 bots, measured in BGP updates seen in 5-second windows.

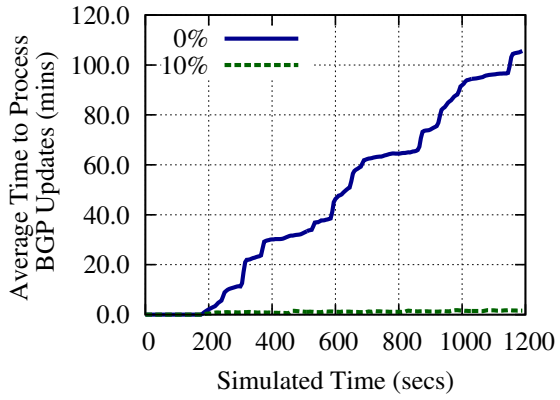

Figure 9: The average time to process a BGP update for core ASes when hold timers are removed for 10% of routers. The delay with no defense deployed is provided as a reference. Attack traffic from the 250, 000 bots starts at time 0, the first link failures occur at time 180.

We discovered that if as few as 10% of the ASes implemented our solution, it would dramatically reduce the impact of CXPST. In fact, our simulations suggest that a 50% deployment would be sufficient to stop CXPST completely.

6 Related Work

6.1 Known Attacks on BGP

With a different goal than CXPST, Bellovin and Gansner [11] propose an algorithm to select links to cut in order to divert existing traffic to a desired set of nodes. Their method assumes a perfect knowledge of the current network topology, which is not easy to obtain [59]. It is presumed that the method for link cutting or forcing topological changes is done via fake routing updates reporting that a candidate link is down in order to direct the existing victim traffic through targeted links or nodes. Our attacker's goal is different. Disruption of service in the routing core is the desired outcome, not traffic diversion.

With a slightly different motivation, yet in the same vein of studying Internet vulnerabilities, Sunder and Perrig [62] introduced Coremelt. In Coremelt, every bot in a large botnet sends data to every other bot in order to cause congestion in the "core" of the Internet. The attack exploits the fact that the customers of an AS can generate more traffic then the AS can handle, called over-subscription. However, small providers that host the bots are also oversubscribed, creating the possibility of the traffic saturating the local ISPs before reaching the core. Unlike CXPST, which selects a small set of links to attack, Coremelt seeks to congest *all* links in the core of the network. Additionally, Coremelt assumes a static network topology, and fails to take into account the dynamic nature of Internet topology.

6.2 BGP Attack Prevention

DDoS prevention techniques that use packet filtering [48] such as packet marking [57, 10, 71, 50, 44, 72] and push-back techniques [73, 43, 45, 34, 4] would have little effect on legitimate traffic generated by a botnet studied in our attack. DDoS mitigation techniques that alleviate the load for given services [21] [18] are not expected to be effective since we are attacking the underlying routing mechanism, and not the services themselves directly. Securing the traffic between BGP speakers [65, 64, 37, 16] or authenticating traffic origins or paths [75, 2, 63] will not be effective against our attack since we are preventing those messages from even flowing using legitimate end node traffic.

Improving resilience by providing failover paths [38] will cause our attack traffic to follow the updates to the

failover path, and will therefore be an ineffective mitigation technique. For example, stabilizing network paths by pro-actively modeling the network [29] and restricting the set of paths an AS can select [30] would be ineffective against a dynamic resource starvation attacks. Containing the faults and avoiding global propagation [5] will not avoid local resource exhaustion. Limiting route exchanges [9] may reduce the control plane traffic and alleviate some convergence problems but will not stop our attack traffic through the advertised paths causing disruptions along the way. Moreover, in the case of forward loop creation that reflectors may cause [26], the total amount of traffic may increase, amplifying our attack.

Unlike other proposed attacks [20], we do not assume compromised routers. Thus, techniques that analyze the behavior of BGP speakers [41, 42, 25] or propose router policy changes [23, 53, 69] do not withstand our attacks. Alleviating resource exhaustion problems by improving routers [49], introducing systems of routers [3], or using software routers [51, 58, 12] do not account for the increasing power of compromised nodes and do not remove all bottlenecks on the deployed routers, including oversubscribed links, line cards and CPU, which will could the CXPST attack to proceed.

The phenomena of events in the control plane of BGP leading to loss of quality of service in the data plane is a well studied phenomenon [66, 19, 67, 40]. These studies provide interesting snapshots that expose the effects of path changes on the data plane.

7 Discussion

While we have not demonstrated the performance of our attack in real networks, data on current router CPU and memory load [22, 70] on the Internet suggests that it is likely to work in practice. In this section, we discuss the reasoning behind some of our assumptions and simulation parameters, why currently-used BGP defenses do not work to stop CXPST, why DDoS defenses are not applicable, why other deployed and or proposed defenses are unlikely to work, and, finally, how to design long-term control plane resilience for the Internet.

7.1 Route Flapping Control Measures

Since the essence of our attack is induction of route flapping on a massive scale, it might seem natural to assume that mechanisms designed to reduce the effect of route flapping would help stop our attack. While some mechanisms — such as minimum router advertisement intervals (MRAI) [32, 39, 13], BGP Graceful Restart [33], and route flap damping [31] — currently exist to deal with route flapping, we have shown in Section 5.1 that they are not effective at limiting our attack. In fact, route flap damping can exacerbate the effects of CXPST, extending the convergence time of the network [46], and may even temporarily cause damping of all routes to a set of networks, making them unreachable.

These mechanisms are not effective because they were designed to deal with transient network events and accidental misconfiguration, not an persistent and deliberate adversary. BGP Graceful Restart was designed to prevent routers from exchanging entire routing tables following a momentary failure. Flap damping is designed to prevent a BGP message flood when physical network events cause routes to oscillate between up and down states. These measures were intended to shield the data plane from link failures, and do not work when the data plane itself was the source of the failure.

7.2 Denial of Service Defenses

Unfortunately, it is unlikely that any near-term defenses or software changes (short of pro-actively tracking down and destroying botnets before they grow beyond the $100,000$-node range) would be effective in mitigating our attack. Existing DDoS defenses such as Phalanx [21] will likely perform poorly since they defend against attacks that are orthogonal to CXPST. These defenses focus on preventing an attacker from disrupting end hosts by flooding them directly with large amounts of traffic. In our attack, traffic is sent directly to colluding bots in a diverse set of networks. In essence, traffic is "wanted" by the end networks, as it is addressed to hosts inside the network [62]. Because of this fact, end networks are unlikely to flag the traffic as malicious.

7.3 Network Complexities

Recent work in increasing router scalability [7], making routers more extensible by using virtualization [24] or even deploying a centralized processing point for routing decisions [22] will be ineffective in protecting against our attack in the long run, since those solutions serve to increase the throughput of current routers without substantially changing the architecture of the routers themselves. Increasing router throughput is an arms race that puts providers at a significant disadvantage — by design, the edges of the network will contain more processing power and traffic generation potential than the core can handle.

7.4 Toward Long-Term Defenses

While our short term solution presented in Section 5.2 stops CXPST, it reduces the ability of routers to automatically react to network issues. Also, other attacks related to

Zhang et al.'s attack might not be prevented. At a high level, the long term solution is to separate the resources used for control plane traffic from those used for data plane traffic. There are a number of ways to achieve this goal, but none are immediately implementable: they all require significant redesign of either router hardware or the Internet control plane. One possible solution is to use private links and dedicated routers for control plane traffic, pushing precomputed routing tables to routers which perform traffic forwarding but do not do route computation [27].

An alternative approach involves using an elevated quality of service (QoS) level [68] for BGP messages. While QoS can be used to ensure a control packet is sent, this does not guarantee that it is received. In order to provide QoS on the incoming side of a connection, packets must be processed and placed into service classes. If packets are not processed at line speed, then the router will be forced to buffer excess packets. Once the processing buffer is full, incoming packets will be dropped until space is available in the buffer. The router can not avoid dropping control packets, since the router must first process the packet to establish if it is or is not control traffic. In some of today's high end routers, incoming packet processing is oversubscribed [35], meaning that they are incapable of making forwarding and queuing decisions at line speed. We note that implementing line speed packet decisions would involve non-trivial changes to the design of routers, as this behavior would have to be enforced in hardware. This means that the monetary costs of defense are high. Thankfully, as shown in Section 5.2, a limited deployment would be sufficient to stop CXPST.

8 Conclusion

In this paper we introduced CXPST, an attack against the Internet control plane carried out using only the data plane. We showed through simulation that a network of $250,000$ commodity nodes can cause significant disruption to the core Internet infrastructure, potentially disabling the entire network. We show that no currently deployed solution is sufficient to prevent this attack, and suggest both short-term configuration changes and long-term architectural changes required to protect the Internet from CXPST and related attacks.

Acknowledgments This work was supported by NSF grant 0917154. We thank Adrian Perrig, Shubho Sen, Ahren Studer, and Brian Weis for helpful discussions about this work.

References

[1] Network and Customer Installation Interfaces — Asymmetric Digital Subscriber Line (ADSL) Metallic Interface. Technical Report 2, American National Standards Institute, 1998.

[2] W. Aiello, J. Ioannidis, and P. McDaniel. Origin authentication in interdomain routing. In *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*, pages 165–178, New York, NY, USA, 2003. ACM.

[3] K. Argyraki, S. Baset, B. Chun, K. Fall, G. Iannaccone, A. Knies, E. Kohler, M. Manesh, S. Nedevschi, and S. Ratnasamy. Can software routers scale? In *Proceedings of the ACM workshop on Programmable routers for extensible services of tomorrow*, pages 21–26. ACM, 2008.

[4] K. J. Argyraki and D. R. Cheriton. Active internet traffic filtering: Real-time response to denial-of-service attacks. In *USENIX Annual Technical Conference*, pages 135–148. USENIX, 2005.

[5] A. Arora and H. Zhang. LSRP: local stabilization in shortest path routing. *IEEE/ACM Trans. Netw.*, 14(3):520–531, 2006.

[6] B. Augustin, T. Friedman, and R. Teixeira. Measuring load-balanced paths in the internet. In *IMC '07: Proceedings of the 7th ACM SIGCOMM conference on Internet measurement*, pages 149–160, New York, NY, USA, 2007. ACM.

[7] H. Ballani, P. Francis, T. Cao, and J. Wang. Making routers last longer with viaggre. In *NSDI'09: Proceedings of the 6th USENIX symposium on Networked systems design and implementation*, pages 453–466, Berkeley, CA, USA, 2009. USENIX Association.

[8] H. Ballani, P. Francis, and X. Zhang. A study of prefix hijacking and interception in the Internet. *ACM SIGCOMM Computer Communication Review*, 37(4):276, 2007.

[9] T. Bates, R. Chandra, and E. Chen. BGP route reflection an alternative to full mesh iBGP, 2000.

[10] A. Belenky and N. Ansari. On deterministic packet marking. *Comput. Netw.*, 51(10):2677–2700, 2007.

[11] S. Bellovin and E. Gansner. Using Link Cuts to Attack Internet Routing. 2004.

[12] A. Bianco, J. Finochietto, G. Galante, M. Mellia, and F. Neri. Open-source PC-based software routers: A viable approach to high-performance packet switching. *Lecture notes in computer science*, pages 353–366, 2005.

[13] A. Bremler-Barr, Y. Afek, and S. Schwarz. Improved BGP convergence via ghost flushing. In *IEEE INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 2, 2003.

[14] K. Butler, T. Farley, P. McDaniel, and J. Rexford. A survey of bgp security issues and solutions. *Proceedings of the IEEE*, 98(1):100 –122, jan. 2010.

[15] CAIDA. AS Relationships Dataset. http://www.caida.org/data/active/as-relationships/, March 2009.

[16] H. Chan, D. Dash, A. Perrig, and H. Zhang. Modeling adoptability of secure bgp protocol. In *SIGCOMM '06: Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 279–290, New York, NY, USA, 2006. ACM.

[17] D.-F. Chang, R. Govindan, and J. Heidemann. An empirical study of router response to large BGP routing table load. In *IMW '02: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 203–208, New York, NY, USA, 2002. ACM.

[18] J. Chou, B. Lin, S. Sen, and O. Spatscheck. Proactive surge protection: A defense mechanism for bandwidth-based attacks. In P. C. van Oorschot, editor, *USENIX Security Symposium*, pages 123–138. USENIX Association, 2008.

[19] J. Cowie, A. T. Ogielski, B. Premore, and Y. Yuan. Global routing instabilities during Code Red II and Nimda worm propagation. Technical Report, Renesys Corporation. Hanover, New Hampshire, USA., 2001.

[20] W. Deng, P. Zhu, X. Lu, and B. Plattner. On evaluating BGP routing stress attack. *Journal of Communications*, 5(1):13–22, January 2010.

[21] C. Dixon, T. E. Anderson, and A. Krishnamurthy. Phalanx: Withstanding multimillion-node botnets. In J. Crowcroft and M. Dahlin, editors, *NSDI*, pages 45–58. USENIX Association, 2008.

[22] M. Dobrescu, N. Egi, K. Argyraki, B.-G. Chun, K. Fall, G. Iannaccone, A. Knies, M. Manesh, and S. Ratnasamy. Routebricks: exploiting parallelism to scale software routers. In *SOSP '09: Proceedings of the ACM SIGOPS 22nd symposium on Operating systems principles*, pages 15–28, New York, NY, USA, 2009. ACM.

[23] D. Dolev, S. Jamin, O. Mokryn, and Y. Shavitt. Internet resiliency to attacks and failures under BGP policy routing. *Comput. Netw.*, 50(16):3183–3196, 2006.

[24] N. Egi, A. Greenhalgh, M. Handley, M. Hoerdt, F. Huici, and L. Mathy. Towards high performance virtual routers on commodity hardware. In *CoNEXT '08: Proceedings of the 2008 ACM CoNEXT Conference*, pages 1–12, New York, NY, USA, 2008. ACM.

[25] K. El-Arini and K. Killourhy. Bayesian detection of router configuration anomalies. In *MineNet '05: Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data*, pages 221–222, New York, NY, USA, 2005. ACM.

[26] N. Feamster and H. Balakrishnan. Towards a logic for wide-area Internet routing. *ACM SIGCOMM Computer Communication Review*, 33(4):289–300, 2003.

[27] N. Feamster, H. Balakrishnan, J. Rexford, A. Shaikh, and J. van der Merwe. The case for separating routing from routers. In *FDNA '04: Proceedings of the ACM SIGCOMM workshop on Future directions in network architecture*, pages 5–12, New York, NY, USA, 2004. ACM.

[28] N. Feamster, J. C. Borkenhagen, and J. Rexford. Guidelines for interdomain traffic engineering. *Computer Communication Review*, 33(5):19–30, 2003.

[29] N. Feamster, J. Winick, and J. Rexford. A model of BGP routing for network engineering. *SIGMETRICS Perform. Eval. Rev.*, 32(1):331–342, 2004.

[30] L. Gao and J. Rexford. Stable internet routing without global coordination. In *SIGMETRICS*, pages 307–317, 2000.

[31] N. W. Group. BGP route flap damping. http://tools.ietf.org/html/rfc2439, January 2006.

[32] N. W. Group. RFC4271 - A Border Gateway Protocol 4 (BGP-4). http://tools.ietf.org/html/rfc4271, January 2006.

[33] N. W. Group. RFC4724 — Graceful Restart Mechanism for BGP. http://www.rfc-editor.org/rfc/rfc4724.txt, January 2007.

[34] J. Ioannidis and S. M. Bellovin. Implementing pushback: Router-based defense against DDoS attacks. In *NDSS*. The Internet Society, 2002.

[35] Juniper. 10-Port 10GBE Oversubscribed Ethernet PIC for T-Seres Core Routers. http://www.juniper.net/us/en/local/pdf/datasheets/1000301-en.pdf, May 2009.

[36] B. B. Kang, E. Chan-Tin, C. P. Lee, J. Tyra, H. J. Kang, C. Nunnery, Z. Wadler, G. Sinclair, N. Hopper, D. Dagon, and Y. Kim. Towards complete node enumeration in a peer-to-peer botnet. In W. Li, W. Susilo, U. K. Tupakula, R. Safavi-Naini, and V. Varadharajan, editors, *ASIACCS*, pages 23–34. ACM, 2009.

[37] S. T. Kent, C. Lynn, J. Mikkelson, and K. Seo. Secure border gateway protocol (s-bgp) - real world performance and deployment issues. In *NDSS*. The Internet Society, 2000.

[38] N. Kushman, S. Kandula, D. Katabi, and B. Maggs. R-BGP: Staying connected in a connected world. In *Proc. NSDI*, pages 341–354, 2007.

[39] C. Labovitz, A. Ahuja, R. Wattenhofer, and S. Venkatachary. The impact of Internet policy and topology on delayed routingconvergence. In *IEEE INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings*, volume 1, 2001.

[40] M. Lad, X. Zhao, B. Zhang, D. Massey, and L. Zhang. Analysis of BGP update surge during Slammer worm attack. In S. R. Das and S. K. Das, editors, *IWDC*, volume 2918 of *Lecture Notes in Computer Science*, pages 66–79. Springer, 2003.

[41] F. Le, S. Lee, T. Wong, H. S. Kim, and D. Newcomb. Minerals: using data mining to detect router misconfigurations. In *MineNet '06: Proceedings of the 2006 SIGCOMM workshop on Mining network data*, pages 293–298, New York, NY, USA, 2006. ACM.

[42] F. Le, S. Lee, T. Wong, H. S. Kim, and D. Newcomb. Detecting network-wide and router-specific misconfigurations through data mining. *IEEE/ACM Trans. Netw.*, 17(1):66–79, 2009.

[43] X. Liu, X. Yang, and Y. Lu. StopIt: Mitigating DoS flooding attacks from multi-million botnets. Technical report, UC Irvine, 2008. Technical Report 08-05.

[44] M. Ma. Tabu marking scheme for ip traceback. In *IPDPS '05: Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05) - Workshop 17*, page 292.2, Washington, DC, USA, 2005. IEEE Computer Society.

[45] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker. Controlling high bandwidth aggregates in the network. *SIGCOMM Comput. Commun. Rev.*, 32(3):62–73, 2002.

[46] Z. M. Mao, R. Govindan, G. Varghese, and R. H. Katz. Route flap damping exacerbates internet routing convergence. *SIGCOMM Comput. Commun. Rev.*, 32(4):221–233, 2002.

[47] Maxmind. Geo IP to ASN dataset. http://geolite.maxmind.com, 2009. December 3, 2009.

[48] J. Mirkovic and P. Reiher. A taxonomy of ddos attack and ddos defense mechanisms. *SIGCOMM Comput. Commun. Rev.*, 34(2):39–53, 2004.

[49] J. Mudigonda, H. M. Vin, and S. W. Keckler. Reconciling performance and programmability in networking systems. *SIGCOMM Comput. Commun. Rev.*, 37(4):73–84, 2007.

[50] M. Muthuprasanna and G. Manimaran. Distributed divide-and-conquer techniques for effective ddos attack defenses. In *ICDCS '08: Proceedings of the 2008 The 28th International Conference on Distributed Computing Systems*, pages 93–102, Washington, DC, USA, 2008. IEEE Computer Society.

[51] L. Peterson, S. Karlin, and K. Li. OS support for general-purpose routers. In *Hot Topics in Operating Systems, 1999. Proceedings of the Seventh Workshop on*, pages 38–43, 1999.

[52] A. C. Popescu, B. J. Premore, and T. Underwood. The anatomy of a leak: AS9121. Technical report, Renesys Corp., May 2005.

[53] S. Y. Qiu. *Toward reliable, verifiable, and policy-compliant inter-domain routing*. PhD thesis, Johns Hopkins University, Baltimore, MD, USA, 2007.

[54] RouteViews. RouteViews Dataset. http://www.routeviews.org/.

[55] M. Schuchard. Stormcaller Simulator. http://www.cs.umn.edu/ schuch/projects/stormcaller.html.

[56] G. Sinclair, C. Nunnery, and B. B. Kang. The waledac protocol: The how and why. In *In proceeding the 4th IEEE International Conference on Malicious and Unwanted Software (MALWARE)*, pages 69–77. IEEE Computer Society, October 2009.

[57] V. A. Siris and I. Stavrakis. Provider-based deterministic packet marking against distributed dos attacks. *J. Netw. Comput. Appl.*, 30(3):858–876, 2007.

[58] T. Spalink, S. Karlin, L. Peterson, and Y. Gottlieb. Building a robust software-based router using network processors. In *Proceedings of the eighteenth ACM symposium on Operating systems principles*, page 229. ACM, 2001.

[59] N. Spring, R. Mahajan, D. Wetherall, and T. Anderson. Measuring ISP topologies with Rocketfuel. *IEEE/ACM Transactions on networking*, 12(1):2–16, 2004.

[60] K. Sriram, D. Montgomery, O. Borchert, O. Kim, and D. R. Kuhn. Study of BGP peering session attacks and their impacts on routing performance. *IEEE Journal on Selected Areas in Communications*, 24(10):1901–1915, 2006.

[61] B. Stone-Gross, M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. A. Kemmerer, C. Kruegel, and G. Vigna. Your botnet is my botnet: analysis of a botnet takeover. In E. Al-Shaer, S. Jha, and A. D. Keromytis, editors, *ACM Conference on Computer and Communications Security*, pages 635–647. ACM, 2009.

[62] A. Studer and A. Perrig. The coremelt attack. In *Proceedings of the 14th European Symposium on Research in Computer Security (ESORICS 2009)*, Sept. 2009.

[63] L. Subramanian, V. Roth, I. Stoica, S. Shenker, and R. H. Katz. Listen and whisper: security mechanisms for bgp. In *NSDI'04: Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation*, pages 10–10, Berkeley, CA, USA, 2004. USENIX Association.

[64] P. C. van Oorschot, T. Wan, and E. Kranakis. On interdomain routing security and pretty secure BGP (psBGP). *ACM Trans. Inf. Syst. Secur.*, 10(3), 2007.

[65] T. Wan, E. Kranakis, and P. C. van Oorschot. Pretty secure bgp, psBGP. In *NDSS*. The Internet Society, 2005.

[66] F. Wang, Z. M. Mao, J. Wang, L. Gao, and R. Bush. A measurement study on the impact of routing events on end-to-end internet path performance. *SIGCOMM Comput. Commun. Rev.*, 36(4):375–386, 2006.

[67] L. Wang, X. Zhao, D. Pei, R. Bush, D. Massey, A. Mankin, S. F. Wu, and L. Zhang. Observation and analysis of BGP behavior under stress. In *Internet Measurement Workshop*, pages 183–195. ACM, 2002.

[68] D. Wrege and J. Liebeherr. A near-optimal packet scheduler for QoS networks. In *Proceedings IEEE INFOCOM'97. Sixteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 2, 1997.

[69] J. Wu. *Toward a robust internet interdomain routing*. PhD thesis, University of Michigan, Ann Arbor, MI, USA, 2009.

[70] Q. Wu, Y. Liao, T. Wolf, and L. Gao. Benchmarking BGP routers. In *IEEE 10th International Symposium on Workload Characterization, 2007. IISWC 2007*, pages 79–88, 2007.

[71] Y. Xiang and W. Zhou. Protecting information infrastructure from ddos attacks by madf. *Int. J. High Perform. Comput. Netw.*, 4(5/6):357–367, 2006.

[72] Y. Xiang, W. Zhou, and M. Guo. Flexible deterministic packet marking: An ip traceback system to find the real source of attacks. *IEEE Trans. Parallel Distrib. Syst.*, 20(4):567–580, 2009.

[73] D. K. Y. Yau, J. C. S. Lui, F. Liang, and Y. Yam. Defending against distributed denial-of-service attacks with max-min fair server-centric router throttles. *IEEE/ACM Trans. Netw.*, 13(1):29–42, 2005.

[74] Y. Zhang, Z. M. Mao, and J. Wang. Low-rate tcp-targeted dos attack disrupts internet routing. In *NDSS*. The Internet Society, 2007.

[75] M. Zhao, S. W. Smith, and D. M. Nicol. Aggregated path authentication for efficient bgp security. In *CCS '05: Proceedings of the 12th ACM conference on Computer and communications security*, pages 128–138, New York, NY, USA, 2005. ACM.