2019 IEEE Symposium on Security and Privacy

On the Feasibility of Rerouting-based DDoS Defenses

Muoi Tran*, Min Suk Kang*, Hsu-Chun Hsiao†, Wei-Hsuan Chiang†, Shu-Po Tung†, Yu-Su Wang†
*National University of Singapore, {*muoitran, kangms*}*@comp.nus.edu.sg*
†National Taiwan University, {*hchsiao, b04902077, b04902003, b04902014*}*@csie.ntu.edu.tw*

***Abstract*—Large botnet-based flooding attacks have recently demonstrated unprecedented damage. However, the best-known end-to-end availability guarantees against flooding attacks require costly global-scale coordination among autonomous systems (ASes). A recent proposal called *routing around congestion* (or RAC) attempts to offer strong end-to-end availability to a selected critical flow by dynamically rerouting it to an uncongested detour path *without* requiring any inter-AS coordination.**

This paper presents an in-depth analysis of the (in)feasibility of the RAC defense and points out that its rerouting approach, though intriguing, cannot possibly solve the challenging flooding problem. An effective RAC solution should find an inter-domain detour path for its critical flow with the two following desired properties: (1) it guarantees the establishment of an arbitrary detour path of its choice, and (2) it isolates the established detour path from non-critical flows so that the path is used exclusively for its critical flow. However, we show a fundamental *trade-off* between the two desired properties, and as a result, only one of them can be achieved but not both. Worse yet, we show that failing to achieve either of the two properties makes the RAC defense not just ineffective but nearly unusable. When the newly established detour path is not isolated, a new adaptive adversary can detect it in real time and immediately congest the path, defeating the goals of the RAC defense. Conversely, when the establishment of an arbitrary detour path is not guaranteed, more than 80% of critical flows we test have only a small number (e.g., three or less) of detour paths that can actually be established and disjoint from each other, which significantly restricts the available options for the reliable RAC operation.

The first lesson of this study is that BGP-based rerouting solutions in the current inter-domain infrastructure seem to be impractical due to implicit assumptions (e.g., the invisibility of poisoning messages) that are unattainable in BGP's current practice. Second, we learn that the analysis of protocol specifications alone is insufficient for the feasibility study of any new defense proposal and, thus, additional rigorous security analysis and various network evaluations, including real-world testing, are required. Finally, our findings in this paper agree well with the conclusion of the major literature about end-to-end guarantees; that is, strong end-to-end availability should be a security feature of the Internet routing by design, not an ad hoc feature obtained via exploiting current routing protocols.

I. Introduction

Botnet-driven Distributed Denial-of-Service (DDoS) attacks have been plaguing critical Internet services, flooding end hosts and services with volumetric attack traffic. A more sophisticated type of DDoS attack, called *transit-link flooding*, which targets the Internet's core connectivity infrastructure, has been recently discussed in academia [53], [32] and quickly moved to real-world incidents [18], [27]. Transit-link flooding attacks utilize botnet to generate low-rate flows between pairs of bots [53] or toward public services [32] such that all of these flows cross a given set of network links, degrading the connectivity for all services using these network links.

The best-known solution that offers strong service guarantees to a selected critical flow (e.g., a connection for mission-critical infrastructure or premium users) under flooding attacks is *bandwidth isolation* mechanisms [30], [17]. They guarantee high bandwidth availability regardless of attack sizes (e.g., botnet traffic volume) by allocating dedicated end-to-end bandwidth for critical flows. However, all bandwidth isolation proposals require *global coordination* between all the autonomous systems (ASes) on the end-to-end path for bandwidth reservation and filtering. Considering the current highly competitive inter-domain transit market, where no global-scale ISP collaboration exists, deploying such bandwidth isolation solutions seems challenging.

Recently, the "*routing around congestion*" (or RAC) defense [51] has been proposed as an alternative solution against server and transit-link flooding attacks. The RAC defense offers *path isolation* for any critical inbound flow of a victim network by dynamically creating a detour path *exclusively* for the critical flow. The beauty of the RAC defense is in its immediate deployability in the current Internet *without* any modification of the Internet infrastructure because it only relies on a well-known BGP inbound route-control mechanism [33] that requires *no* coordination between ASes.

In this paper, we perform an in-depth analysis of the (in)feasibility of the RAC defense based on actual Internet topology, business relationship, and public routing data. Through our analysis, we point out that the rerouting approach, though intriguing, cannot possibly solve the challenging flooding problem. We show a fundamental *trade-off* between two highly desirable properties of the RAC defense. An effective RAC defense should (1) *guarantee* the establishment of a detour path of its choice so that it can reroute the selected critical flow when experiencing congestion, and (2) *isolate* the obtained detour path from non-critical flows so that it is used only for the selected critical flow. The discovered trade-off between the two properties, unfortunately, shows that in the current Internet, the RAC defense can achieve either the isolated detour paths or the guaranteed establishment of detour paths but not both. Achieving the instant, highly available detour paths via the RAC defense, therefore, appears to be infeasible.

Worse yet, we show that failing to achieve either of the two properties makes the RAC defense not only less effective but nearly *unusable* in practice. On one hand, we show that the lack of complete path isolation significantly limits the bandwidth availability of the detour path because many non-critical flows end up sharing it with the critical flow. Perhaps more importantly, the lack of path isolation also enables a new adaptive attack, called a *detour-learning attack*, that accurately identifies every detour path establishment in real time. After learning the newly established detour path, the

DOI 10.1109/SP.2019.00055

adaptive adversary can immediately congest the new detour path by flooding one of its transit links. Unfortunately, the RAC defense cannot react to such adaptive flooding attacks quickly enough due to the undesirable long delays (e.g., 85 seconds even with a small size network of 1,000 ASes) in switching to another detour path.

On the other hand, we show that when the establishment of an arbitrary detour path is not guaranteed, it is extremely challenging to operate the RAC defense reliably because the majority (e.g., 80%) of critical flows we test have only small numbers (e.g., three or less) of disjoint detour paths that can actually be established. Note that the fact that a detour path can be established does not mean that it would have enough bandwidth to serve the critical flow; thus, the more the establishable disjoint detour paths are available, the more reliable RAC operation can be achieved.

The main goal of our paper is to draw some practical conclusions about the very challenging server/link flooding problem. The analysis of the RAC proposal has led us to learn some useful lessons. A crucial methodological lesson is that, when analyzing the feasibility of a rerouting-based DDoS defense proposal, it is not enough to evaluate the functional feasibility of the proposal based on the protocol specifications. Even if the specification [49] allows BGP messages with excessively long AS path within the theoretical limit (i.e., 2,034), the actual network operation communities (e.g., NANOG) publicly condemn the practice of broadcasting unnecessarily long (e.g., AS path longer than 75) BGP messages and strongly suggest filtering them [29], [38]. It is also well-known that certain Cisco routers, by default, filter BGP messages with AS path longer than 254 [46]. Therefore, the analysis of BGP-routing-based DDoS proposals must be founded on thorough feasibility checks with real-world feasibility tests, acceptance by network operation communities, and various large-scale simulations.

Our work reconfirms the previous conclusion of multiple independent projects [17], [31], [30], [26]; that is, defending against transit-link flooding attacks requires path or bandwidth isolation that is achievable only through large-scale coordination among many ASes. We hope that our study will renew the discussion of new, clean-slate Internet architectures that enable such inter-domain coordination for highly available Internet services.

Organization. We review the routing around congestion (RAC) defense along with the summary of transit-link flooding attacks in Section II. We present two desired goals of the RAC defense, namely, path isolation and guaranteed detour establishment in Section III, and subsequently highlight the fundamental trade-off between these two in Section IV. In Section V, we highlight the consequence of lack of path isolation with a new metric called path leakage, and, in Section VI, we demonstrate a new adaptive attack that can exploit even a small amount of path leakage to defeat the RAC defense. In Section VII, we demonstrate how hard it is to find a detour path for the majority of critical flows when the detour-path establishment is not guaranteed. We also investigate the required effort to make the RAC defense possible in the current Internet in Section VIII. Finally, we conclude our paper in Section IX.

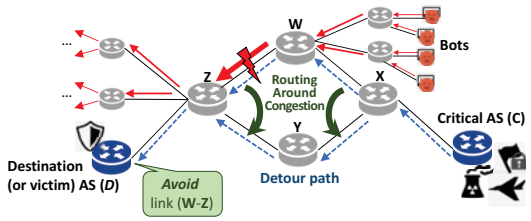


Figure 1: A RAC defense example against a botnet-based transit-link flooding attack.

II. Background and Related Work

DDoS defenses have been extensively studied over the past two decades [40], [58], [59], [45], [23], [35], [39]; refer to the following articles [44], [60] for more comprehensive surveys.

A. Transit-Link Flooding Attacks

In last few years, transit-link flooding DDoS attacks have been proposed and demonstrated substantial damage to the core of the Internet (e.g., backbone links in large ISPs or inter-ISP links) [53], [32]. Transit-link flooding attacks have been further studied to achieve more efficient botnet resource usage [48] or distributed botnet coordination [34].

Figure 1 shows how a transit-link flooding attack creates a large number of attack flows and congests a targeted transit link. The adversary controls her bots to send traffic through the link (W–Z), making all other flows crossing the targeted link experience congestion. The critical flow (see the blue dashed lines in Figure 1) from the critical AS C (i.e., the source of the critical traffic) to the destination AS D also traverses the congested link. This is in sharp contrast to traditional server-flooding attacks that aim to choke the resources (e.g., computation, memory, or access link bandwidth) of the end target (e.g., the destination AS D in Figure 1).

Transit-link flooding attacks have two specific characteristics that make them exceedingly effective at scale while rendering traditional defense mechanisms irrelevant. First, they attack targets *indirectly*. As the locus of attack is different from the targeted end servers, they cannot be easily detected by intrusion-detection systems and firewalls at the end servers. Second, these attacks use protocol-conforming traffic flows that are *indistinguishable* from legitimate flows, thereby causing high collateral damage when flows are simply dropped to relieve congestion.

End-to-end bandwidth guarantee. A line of research that aims to achieve strong availability guarantees against transit-link flooding attacks have offered the *bandwidth isolation* mechanism. These proposals isolate attack traffic by the end-to-end bandwidth reservation and enforcement, thereby providing guaranteed bandwidth for critical flows; e.g., STRIDE [30], SIBRA [17]. A critical flow can be protected by SIBRA and its share of guaranteed bandwidth does not diminish even when the number of bots outside the critical and destination ASes increases.

Other partial solutions. Several defense mechanisms have been proposed to partially mitigate the link-flooding problem. SPIFFY [31] is a single-domain flow-testing solution

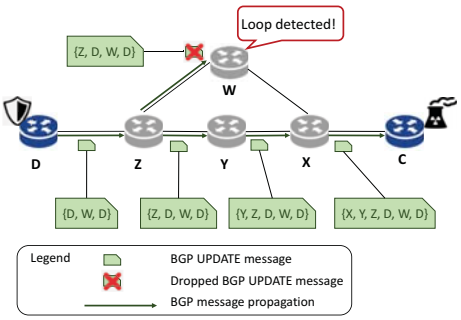


Figure 2: An example of how AS D uses BGP poisoning to poison AS W and establishes a detour path $\{C, X, Y, Z, D\}$.

that deters rational link-flooding adversaries. NetHide [43] can be applied to some ASes that wish to hide its internal network topology to arbitrary adversaries, thus rendering attack reconnaissance difficult. CoDef [36] suggests an inter-domain collaboration protocol to test potential link-flooding flows. LinkScope [57] attempts to detect link-flooding attacks quickly with its new inter-domain protocols. RADAR [61] shows that software-defined networking (SDN) can be used to rate limit link-flooding attacks.

B. Routing Around Congestion (RAC) Defense

The RAC defense was recently proposed as a new approach that offers to protect services from server and transit-link flooding attacks [51]. At a high level, the RAC defense dynamically searches for a detour path that does not include the congested links. For example, in Figure 1, RAC reroutes the critical flow into the detour path (C, X, Y, Z, D) to avoid the congested link W–Z.

Notably, the RAC defense enables an individual AS (e.g., a destination AS) to control the routes for its inbound traffic, which has been considered difficult *without any coordination* with its upstream ASes. The technical underpinning of this rerouting capability is the *BGP route poisoning* mechanism [33] that makes the incoming traffic to avoid any particular AS in the upstream.

Figure 2 illustrates an example in which the destination AS D uses BGP poisoning to poison AS W. In particular, AS D broadcasts a BGP UPDATE message including the AS-path $\{D, W, D\}$. Note that to ensure this poisoned message is accepted by the RPKI-based origin verification [37], the RAC defender also adds its own AS number (e.g., D) to the end of the AS-path included in the message. When AS W receives the BGP poisoning UPDATE message, it finds its own AS number in the message and then ignores the message because otherwise a routing loop can be created [49]. Thus, AS C would send the critical traffic through the detour path (C, X, Y, Z, D).

The RAC defense [51] utilizes the BGP poisoning capability to proactively *avoid* one or more upstream ASes so that the destination AS D can establish an arbitrary detour path for critical flows from C to D. To enforce the critical flows from C to D to follow the newly established detour path, not the old default path, the RAC deployer (or D) uses a more specific destination prefix (i.e., longer prefix) for the detour path announcement, which is also known as a hole punching technique [51].

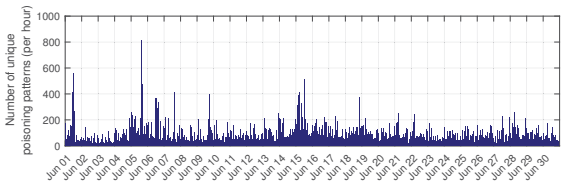


Figure 3: Number of unique BGP poisoning messages per hour from June 1, 2018 to June 30, 2018.

Table I: List of Top-10 ASes that had generated most poisoning patterns in June 2018.

ASN	AS name	No. Unique Poisoning Patterns	Avg. No. Avoided ASes
54994	Quantil Networks Inc.	66	2.20
47065	USC / UFMG PEERING Research Testbed	42	1.10
14061	DigitalOcean, LLC	31	2.32
28349	TVC Tupa Ltda.	25	7.64
15133	EdgeCast Net, Inc. Verizon Dig Med Serv	23	2.09
43996	Booking.com BV	19	2.26
11338	SKY SERVIOS DE BANDA LARGA LTDA	13	2.15
25933	Vogel Solues em Telecom e Informtica S/A	12	3.50
204893	Pawel Zamaro	11	3.36
11123	Ultimate Internet Access, Inc	11	2.91

Practicality of BGP poisoning. Before we analyze the feasibility of the RAC proposal, we examine whether the BGP poisoning is feasible and actually used in practice. To the authors' best knowledge, no such measurement studies have been done yet.

Figure 3 shows a simple measurement study during a one-month period (June 1, 2018 to June 30, 2018) from the RIPE dataset [12]. BGP poisoning messages are constantly generated and broadcast in the current Internet. We also notice that the poisoning messages are continuously generated without a clear diurnal pattern, which implies that the BGP route poisoning is a globally practiced BGP operation trick.

Table I shows the top-10 ASes that had generated most unique BGP poisoning patterns (or the unique sets of ASes to be poisoned or avoided) in June 2018. First, we emphasize that the BGP poisoning is not an unusual behavior created by a small number of illegitimate network operators; it is rather a widely adopted network operation practice. One data center network, Quantil Networks Inc., had announced 66 unique poisoning message patterns in a month with on average 2.20 avoided ASes, showing that some commercial networks frequently utilize BGP poisoning on daily basis. The second AS from the list is a research testbed network, USC / UFMG PEERING Research Testbed [10], and it had generated 42 unique poisoning patterns.

Figure 4 shows the histogram of the number of avoided ASes in the poisoning messages observed in June 2018. The majority of poisoning message patterns have a small number (e.g., 1–3) of avoided ASes as suggested by the LIFEGUARD [33] study. Yet, there also exist some rare cases with a long list of poisoned ASes, up to 15, in our observation.

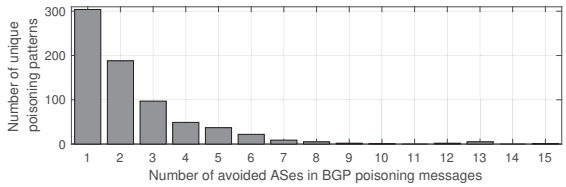


Figure 4: Number of avoided ASes in the BGP poisoning messages observed in June 2018.

Inferring the purposes of the actual poisoning messages and measuring their effectiveness are beyond the scope of our paper; however, a large number of poisoning messages and unique, diverse patterns show that the BGP poisoning is a widely practiced routing tool in the current Internet.

III. Two Desired Properties of the RAC Defense

The crux of the RAC defense [51] is to establish a detour path for a critical flow when flooding attacks congest the current path. Since the RAC deployer (e.g., the victim destination AS D in Figure 1 or Figure 2) is under flooding attacks at the moment, it wishes that a detour path to be established in real-time (e.g., within a couple of minutes at worst) before significant damage occurs, and that a detour path is used exclusively for the selected critical flows but not for other non-critical flow. Here we summarize the two desired properties:

Property 1: *Path isolation.* The RAC deployer wants to have a detour path that is *exclusively* used by the selected critical flow, or *isolated* from other non-critical flows.

Property 2: *Guaranteed detour establishment.* The RAC deployer wants its BGP message for a specific detour path of its choice is accepted by the upstream ASes and thus establishes the detour path with a guarantee.

In the subsequent sections, we describe an undesirable trade-off between the two properties of the RAC defense (§IV), and we show the consequences of the lack of **Property 1** (§V and §VI) and the lack of **Property 2** (§VII).

IV. Trade-off between Path Isolation and Guaranteed Detour Establishment

In this section, we show why it is hard to achieve the two desired properties of the RAC defense — namely, the path isolation (**Property 1**) and guaranteed detour establishment (**Property 2**) — at the same time in the current Internet.

We first analyze the requirements of achieving an isolated detour path through our in-depth analysis of potential detour paths and their BGP poisoning patterns in the current Internet topology (§IV-A). We also investigate the conditions for achieving a highly confident detour path establishment via a longitudinal study of BGP UPDATE messages (§IV-B). Then, we finally show why the two desired properties cannot be achieved simultaneously due to their two contradictory conditions (§IV-C).

A. Requirements for Detour Path Isolation

To achieve the path isolation, the RAC defense should create a detour path *exclusively* for a critical flow of choice. It requires the BGP UPDATE messages to traverse only the ASes on the detour path. If the message arrives at an AS that is *not* on the detour path, that AS may accept the message and then start sending non-critical flows through the detour path. The RAC paper [51] proposes that all the neighbors of the ASes on the detour path must be poisoned to guarantee the exclusive usage of the detour path.

How many ASes does RAC need to poison? As we reviewed in Section II-B, the basic idea of BGP poisoning has been studied in-depth [33], [51] and our analysis of actual BGP update datasets also confirms its non-negligible usage in practice. However, poisoning a large number (e.g., tens, hundreds, or even thousands) of ASes in a single BGP UPDATE message has never been studied. For example, LIFEGUARD [33] tests with the poisoning messages containing only a single AS and the RAC paper [51] does not discuss how many ASes should be poisoned. Therefore, in this section, we analyze this requirement of creating an isolated detour path, i.e., poisoning all neighbors of the ASes on it.

We use the Chaos simulator [4], an open-source BGP simulator that has been used to evaluate the RAC proposal [51], to simulate the network topology and BGP propagation among ASes in the network. Our simulation starts with the Chaos simulator taking the inferred AS relationship from CAIDA [3] as the input to build the network topology among about 60 thousand ASes. In the initialization phase, each AS broadcasts a BGP UPDATE message containing its AS number to its neighbors. The messages are propagated to other ASes in the network, allowing them to calculate the default routes to each other. The AS relationship from CAIDA [3] that we use in our analysis describes the AS level connectivity based on relationships between ASes: provider, customer, peer or sibling.

To determine a packet forwarding path, we assume that an AS applies the following widely adopted BGP policies in order [25]: (1) The AS prefers customer links over peer links and peer links over provider links. This rule comes from the fact that the ASes are most interested in maximizing their revenues in determining a forwarding path [24], [25]; (2) The AS prefers the shortest AS-path length route; and (3) If multiple best paths exist, the AS uses the AS numbers to break the tie. Particularly, the first policy guarantees that the created AS-level routes are *economically viable* because it ensures that all the ASes on the routes are guaranteed positive revenues.

Thereafter, we randomly choose 1,000 pairs of ASes to be the Critical AS (i.e., the source of the critical traffic) and the Destination AS (i.e., the defender deploying RAC), or C–D in short. With each C–D pair, we enumerate all possible detour paths (i.e., available BGP routes from C to D) and count the number of ASes to be poisoned for each detour path. Then we select one detour path per C–D pair and show the distribution of the number of ASes that need to be poisoned for the 1,000 selected detours in Figure 5. Note that when there is more than one detour path per C–D pair exists, we choose among them the one with the minimum number of neighboring ASes to evaluate the lowest possible number of ASes to be poisoned

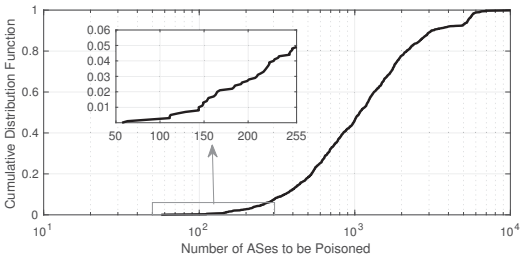


Figure 5: Number of ASes to be poisoned for the selected 1,000 detour paths.

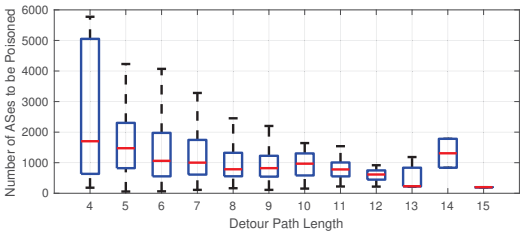


Figure 6: Relationship between the detour path length and the number of ASes to be poisoned.

of the RAC defense.[1]

Figure 5 shows that the number of ASes to be poisoned is tremendous: the maximum number of ASes to be poisoned is 10,846; the majority of cases have *more than a thousand* of ASes to be poisoned; and only less than 5% of cases have less than 255 ASes to be poisoned.

Reasons behind the large number of ASes to be poisoned. Let us investigate why the isolated detour paths require large numbers (e.g., few hundreds to thousands) of ASes to be poisoned. We first look at the relationship between the length of a detour path and the number of ASes to be poisoned. The box-and-whisker plots in Figure 6 show the distribution of detour path length where each of them contains 2 vertical dash lines representing the first and the fourth quartile of the data set and the ends of the whisker describe the minimum and maximum value. The red band inside the blue box separates the second and the third quartile and represents the median of the data set. Figure 6 shows a counter-intuitive relationship; i.e., the number of ASes to be poisoned tends to *decrease* as the detour path length *increases*. For example, the median value decreases gradually from approximately 2,000 ASes with the 4-hop detour paths to only about 200 ASes with 13-hop detour paths.

To better understand this counter-intuitive result, we further analyze the characteristics of ASes on the detour paths in different detour path lengths. In Figure 7, we calculate the average number of Tier-1 and Tier-2 ASes that appear in each detour path and show the results in each set of detour paths grouped by their length. We categorize the ASes on the selected detour paths based on their tier, following the widely

[1]In practice, a RAC defender may use some other factors (e.g., number of hops, geographic distance) for choosing one detour path.

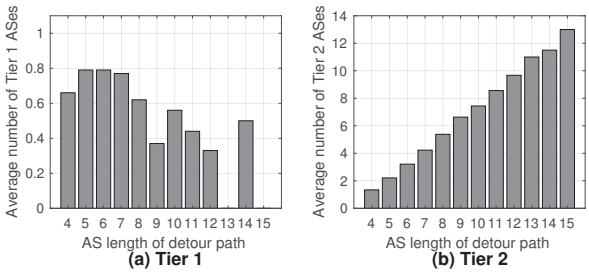


Figure 7: Average number of Tier-1 and Tier-2 ASes in different detour path length group.

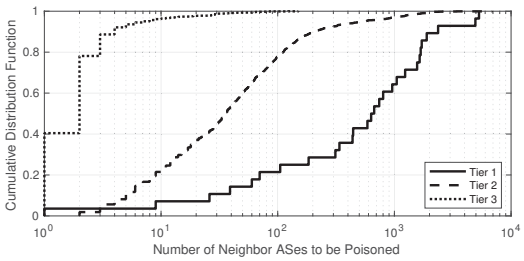


Figure 8: Distribution of the number of neighbor ASes to be poisoned for the 1,000 selected detour paths, classified by their AS type.

accepted classification [56].[2] Figure 7(a) shows that the detour paths with shorter length (e.g., 4-7) are more likely to include a Tier-1 AS, indicated by the average number of Tier-1 ASes is around $0.7-0.8$ per detour path when the detours are $4-7$ hops but suddenly drops to around 0.4 per detour path when the detour is longer than 8 hops. On the contrary, Figure 7(b) shows that Tier-2 ASes are the majority type in the detour paths and their relative proportion is increasing as the AS path length.

Figure 6 and Figure 7(a) indicate that the number of Tier-1 ASes on detour paths has a *significant impact* on the total number of ASes to be poisoned. We confirm this finding by showing the distribution of the number of neighbors of the ASes on the detour path classified by their tiers in Figure 8. Each line represents a group of classified ASes and its CDF of number of neighboring ASes. Figure 8 shows that 80% Tier-1 ASes have relationship with more than 100 ASes and 40% Tier-1 ASes have more than 1,000 neighbors. Thus, including one or more Tier-1 ASes on the detour path usually causes the RAC defense to poison hundreds to thousands of neighbors of these ASes in order to prevent any of them from receiving poisoned messages. We present an additional supporting measurement data in Appendix A, where we show the vast majority of all possible detour paths include at least one or more Tier-1 ASes.

Maximum AS path length allowed by *specification*. The BGP-4 specification (RFC 4271 [49]) defines the maximum packet size limit of 4,096 bytes for a single BGP UPDATE

[2]Tier-1 AS has no provider and can send traffic to all other ASes without paying for traffic transit or peering, Tier-2 AS purchases traffic transit from at least one provider AS and connects to one or more Tier-3 ASes and Tier-3 AS has no customer AS.

message. Considering the header and necessary fields, one BGP UPDATE message may include up to 2,034 ASes in its AS-PATH field. By exploiting these maximum 2,034 AS number fields, one can poison all the ASes that need to be avoided for isolated detour path for the majority (e.g., 80%) of the tested 1,000 C–D pairs; see Figure 5.

B. Requirements for Guaranteed Detour Path Establishment

To guarantee that a detour path will be established, the RAC deployer (i.e., the destination AS) must ensure that its BGP poisoning messages are propagated by *all* the ASes on the detour path from D to C (e.g., D, Z, Y, X, C in Figure 2) without getting filtered out on the way. Although the BGP specification supports excessively long AS path in BGP UPDATE messages (e.g., up to 2,034), we have reasonable doubt that in practice some ASes would filter UPDATE messages with such long AS paths; see the complaints in NANOG community regarding long AS paths [29], [38], the best current practice of BGP operations [21], and Cisco routers' default filtering based on AS path length [46].

In this section, we investigate how the ASes in the current Internet handle excessively long AS paths in BGP UPDATE messages. We describe two approaches to this investigation: *active* and *passive* measurements of the BGP UPDATE message filtering behaviors.

Active measurement. A large-scale active measurement is an *ideal measurement* on how the AS path length affects the acceptance of BGP poisoning messages in the current Internet. This would require a large number of ASes to send *probe* BGP UPDATE messages with different AS path length (e.g., 10s, 100s, 1000s, or longer). Then, we would monitor if these BGP messages are propagated to various geographically distributed vantage points without getting dropped on the way. This ideal experiment would provide a highly accurate estimation of actual BGP message treatment based on the AS path length. However, to the authors' best knowledge, such a large-scale, collaborative BGP testbed does not exist, unfortunately.

We also have tried a small-scale active measurement at two different networks where we can announce customized BGP messages; however, we have learned that even small-scale active tests are nearly impossible in the current Internet. Our first testing network, PEERING Research Testbed [10], does not allow BGP UPDATE messages with AS path length longer than 10. Our second testing network is one of our academic institutions and it also refuses to experiment with BGP UPDATE messages with AS path length longer than 30. In both cases, we have found two common reasons for refusing our experiment requests: (1) abnormally long AS path length cannot even be configured in their BGP routers; and, perhaps more interestingly, (2) the two institutions have explicitly expressed their concern that any abnormal BGP messages may crash their routers and their upstream routers. Our failed attempts for active measurements nevertheless strengthen our doubt that BGP messages with excessively long AS path would be filtered out by many ASes.

Passive measurement. We instead perform a passive measurement study of the public BGP UPDATE database, particularly RIPE BGP repositories [12], collected during the six-month period from January 1 to June 30, 2018. This dataset contains

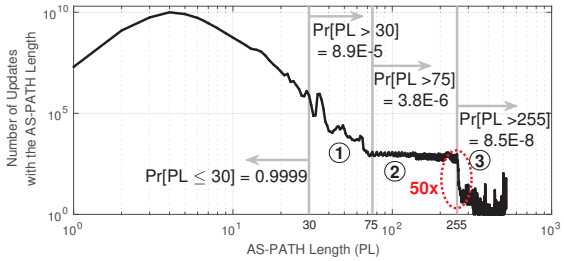


Figure 9: AS path length distribution of 37 billion BGP UPDATE messages collected during a six-month period from Jan 1 to June 30, 2018. At around the AS path length of 255, a sharp (~ 50 times) decrease of the occurrence of UPDATE messages is observed. $\Pr[\cdot]$ denotes the empirical probability distribution of the path lengths.

the UPDATE messages sent from all origin ASes in the Internet to 364 globally distributed vantage points (i.e., the peering members of the 24 RIPE collectors), offering a great view of the UPDATE messages that have been successfully delivered across the globe. Note that, however, we do not conduct controlled experiments (e.g., testing different AS path lengths from selected ASes) in this purely passive measurement study and thus it lacks the *ground truth* of all the generated BGP UPDATE messages during the observation period.

In our longitudinal analysis of 37 billion BGP UPDATE messages, we observe that the main contribution to the excessively long (e.g., ≥ 30) BGP messages is the well-known *AS prepending*; i.e., the origin AS number is repeated multiple times when the UPDATE message is first crafted. The following example shows a typical AS prepending pattern.

- **Prefix**: 2a0b:3c47:cabb::/48
- **AS-PATH**: {12307, 57118, 196621, 15576, 174, 136620, 204816, 137875, 137875, 137875, 137875, 137875, 137875, 137875, 137875, 137875, 137875, ··· (137875 skipped 500 times), ··· 137875, 137875, 137875, 137875}

In this example, the origin AS 137875 repeats its own AS number more than 500 times before sending the UPDATE message to its neighboring ASes. Such pointless long AS paths in the UPDATE messages have been criticized by network operator communities for being ignorant and causing computation and space costs to the ASes that have to process them [29], [38].

Moreover, we also find a multifaceted evidence that the majority of the ASes in the current Internet *do* filter UPDATE messages with extremely long AS paths at a specific AS path length. Figure 9 shows the AS path length distribution of all the 37 billion BGP UPDATE messages. We plot the number of UPDATE messages for each AS path length (see both X-axis and Y-axis are in log scale) from 1 to 520 (the longest AS path observed during the period). We focus only on the rare, abnormal cases (which amount to only 0.01%) where path length is ≥ 30.

Abnormal cases (path length ≥ 30). Only less than 0.01% of UPDATE messages have AS paths longer than 30 and how these abnormal messages are handled provides useful insights as to how the ASes filter such long AS-path messages.

From the distribution of abnormal UPDATE messages found in Figure 9, we have three observations:

① *Reduction of longer messages in the range [30,75).*
② *No reduction of messages in the range [75,255).*
③ *Sudden reduction of messages at around path length of 255.*

Note that we cannot directly measure how ASes filter long AS-path messages because we do not control or even know how many BGP messages with specific AS-path length have been generated in this passive measurement study. Yet, from the above observations we conjecture the following highly feasible BGP message filtering practices based on the AS-path length:

❶ *Some ASes filter messages with path length in [30,75).*
❷ *No ASes filter messages with path length in [75,255).*
❸ ***Majority of ASes filter messages with path length*** ≥ 255***.***

We confirm that the conjectured filtering practices are well aligned with the several independent, anecdotal evidence of BGP message filtering practices:

(1) The BGP's best current practice suggested by the IETF, equipment vendors (e.g., Cisco), and network operator communities (e.g., NANOG) encourages BGP UPDATE inbound filtering based on the AS path length of 40 [42], 50 [55], and 75 [22]. The ad hoc implementation of these best common practices appears to be well aligned with our observation ① and the conjecture ❶.
(2) We have not found any anecdotal evidence that ASes may filter BGP UPDATE messages with AS path length of 75 or longer and shorter than 255. This explains the observation ②, that is, no discernible reduction of messages in this range, and the conjecture ❷ well.
(3) We have found that Cisco routers with up-to-date IOS operating systems are configured by default to drop UPDATE messages with AS path length equal to or longer than 255 [46]. This has been implemented as a remedy to the Cisco IOS bug (bug#: CSCdr54230), which makes the BGP routers misbehave when processing UPDATE messages with AS path longer than or equal to 255 [47]. Considering Cisco's strong dominance in the network equipment markets in the last few decades, it is understandable that the majority of BGP messages longer than 255 are highly likely to be filtered by any Cisco routers on their propagation paths (that is, ③ and ❸).

Filtering practices in ISPs. From our personal conversations with two ISPs, we have confirmed that BGP UPDATE message filtering is indeed used in practice. We heard from one large Swiss ISP that they filter on AS path length > 40. We also heard from a large ISP in Taiwan that they implement an even stronger white-list filtering, which filters arbitrary BGP poisoning messages. This anecdotal evidence is well aligned with the above passive measurements and our conjecture on the filtering practice.

C. Contradictory Requirements of Two Defense Properties

We conclude our findings that the two highly desired defense properties — path isolation and guaranteed detour establishment — *cannot* be achieved at the same time because their requirements contradict each other. To achieve path isolation for more than 95% of detours, RAC defender has to include more than 255 ASes in its BGP UPDATE messages (§IV-A). In contrast, the BGP UPDATE message must contain the AS path shorter than 255 so that it is not filtered out and the detour path establishment will be guaranteed (§IV-B).

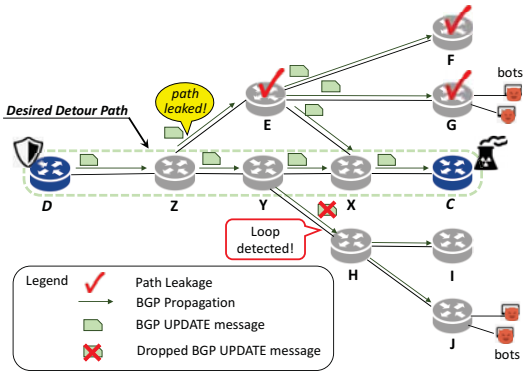


Figure 10: Path leakage of a message poisoning AS H sent by a destination AS D for the detour path $\{C, X, Y, Z, D\}$.

In the following sections from Section V to VII, we investigate the consequences of the trade-off.

V. Non-Isolated Detour Path and Path Leakage

Strong isolation of detour paths is a highly desired defense property; yet, it cannot be achieved if the RAC defender prefers the guaranteed detour path establishment. In this section, we investigate the problems when the established detour path is *not* isolated. Particularly, we introduce a metric called path leakage, whose definition follows below, to measure the negative consequence of having non-isolated detour paths. Then, we optimize the RAC defense to minimize the path leakage. In the following Section VI, we use the measured path leakage and discuss how new attacks against the RAC defense can be effective even with small amount of path leakage.

A. Metric to Evaluate Non-Isolated Detour Paths

The information about the bandwidth capacity of the inter-domain links between ASes is known to be inaccessible for public [51]. Hence, it is also difficult to estimate the bandwidth availability of a detour path.

Alternatively, we use the number of poisoned ASes that are not on the detour paths to evaluate the bandwidth availability of a detour path. This comes from the observation that both the critical flow and some other non-critical flows are rerouted to a non-isolated detour path and the detour path may carry increasing number non-critical flows as the number of non-critical ASes receiving the BGP poisoning message grows. More formally, we introduce the notion of *path leakage* and use it as a metric to quantify the bandwidth availability of a detour path in our work.

Definition 1 (Path leakage)**.** An AS is said to have (or observe) the *path leakage* of a BGP UPDATE message U when the AS is not on the detour path but receives U. The *amount* of path

leakage of U is measured by the total number of ASes that have the path leakage of U.

Figure 10 illustrates the path leakage of a message sent by a destination AS D for the detour path $\{C, X, Y, Z, D\}$. D constructs a poisoned UPDATE message to poison AS H and broadcasts it to the network. When the BGP message is sent to the ASes that are not on the detour path and is accepted by them, the path is said to be *leaked*. In Figure 10, the BGP poisoning message is accepted by the ASes that are not on the detour path (e.g., E, F and G).

B. Minimization of Path Leakage Due to Non-isolated Detours

In this section, we further improve the RAC defense to *minimize* the amount of path leakage a BGP poisoning message can have. We first define the problem of establishing a non-isolated detour path with the minimum amount of path leakage for a critical AS and destination AS (or C–D) pair and then find a sub-optimal solution we can calculate in practice.

We aim to establish a detour path with a BGP poisoning UPDATE message that minimizes the amount of path leakage.[3] We formally define an optimization problem as follows.

Non-isolated detour establishment problem [P1]. Let us consider that a destination AS D creates an UPDATE message U and sends it through the detour path $\mathcal{R} = \{C, R_1, R_2, \cdots, R_n, D\}$. Our optimization goal of this problem is to find the set of ASes $\mathcal{P}$ among all ASes in the Internet that minimizes the amount of path leakage of U such that $|\mathcal{P}| \leq B$, where B denotes the maximum allowed size of $\mathcal{P}$.

We show that this optimization problem **[P1]** can be modeled to a variant of the min-cut problem, that is the general version of the Minimum-Size Bounded-Capacity Cut (MinSBCC) problem [28], see Appendix B for details. The general MinSBCC problem is shown to be NP-complete and has a known approximation [28]; yet, it is a bi-criteria algorithm and thus it has no guarantee that the solution satisfies the condition of the bounded number of ASes to be poisoned.

Simplified Heuristic Solutions. Since the problem of minimizing path leakage seems to be hard and the known approximation is inappropriate, we (1) simplify the problem formulation of **[P1]**, and (2) perform a greedy algorithm on the simplified problem to find a sub-optimal solution.

First, we can simplify **[P1]** by poisoning *only* the neighboring ASes of the ASes on the detour path. That is, instead of choosing $\mathcal{P}$ (i.e., the set of ASes we poison) among all the ASes in the Internet, we choose them from the neighboring ASes of the ASes on the detour path. Our intuition is that stopping the spread of path leakage at earlier location (i.e., closer to the detour path) would result in a better solution of **[P1]**, i.e., smaller amount of path leakage.

Second, we implement a *greedy algorithm* to solve the simplified **[P1]**. Let us first call $\mathcal{Q}$ the set of ASes that we can poison (i.e., the set of all neighboring ASes of the ASes on the detour path). To remove path leakage at any AS, all of the ASes in $\mathcal{Q}$ that can deliver U to that AS must be poisoned. We call this set of ASes that needs to be poisoned for a given AS_i the *isolating set* $S_i \subset \mathcal{Q}$. Our greedy algorithm aims to maximize the number of ASes that are not on the detour path while the union of their isolating sets is still within the bound B. In each round, AS_i with the smallest $|S_i|$ is picked, the selected S_i is added to $\mathcal{P}$, and all other S_j $(i \neq j)$ are re-calculated. The greedy algorithm finally outputs $\mathcal{P}$, the set of ASes that we poison. We describe our greedy algorithm in details in Appendix C.

[3]Complete elimination of path leakage guarantees the path isolation, which has been shown to be hard if the RAC defender prefers the guaranteed detour path establishment.

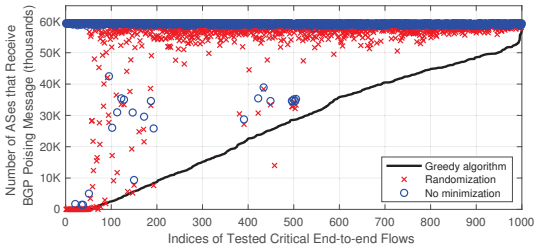


Figure 11: Amount of path leakage in three experiments: no minimization, randomization and greedy algorithm.

Measuring the path leakage with the greedy algorithm. To evaluate our greedy algorithm, we use the Chaos simulator [4] and the enumerated detour paths between 1,000 randomly selected C-D pairs to run several experiments. We set the bound B to be 255; see Section IV-B. Since ASes prepend their own AS numbers to the BGP UPDATE message when they forward it, the poisoned message must reserve some empty spaces for the ASes on the detour paths. Therefore, the number of ASes to be poisoned is at most $B - |R| - 2$, where the destination AS already takes two slots.

To show the effectiveness of our greedy algorithm, we construct three different BGP poisoned UPDATE messages and test how they result in different amount of path leakage.

1. *No minimization.* We poison only one AS AS_L to avoid the congested link on the default path, and append the destination AS D to the BGP UPDATE message: $\mathsf{U}_{no\_minimization} = \{D, AS_L, D\}$.
2. *Randomization.* We also randomly choose $(252-|R|)$ ASes among all neighboring ASes to be poisoned in the BGP UPDATE message: $\mathsf{U}_{random} = \{D, AS_L, AS_1, AS_2, \cdots, AS_{252-|R|}, D\}$.
3. *Greedy algorithm.* We include the $(252-|R|)$ ASes chosen from our greedy algorithm in the BGP poisoned message: $\mathsf{U}_{greedy} = \{D, AS_L, AS_1, AS_2, \cdots, AS_{252-|R|}, D\}$.

Figure 11 shows the amount of path leakage when three BGP poisoning messages are sent out. Without any path leakage minimization attempt (i.e., no minimization), the BGP poisoning message ends up being delivered to almost all $60K$ ASes in the network. When we avoid randomly a set of $(252 - |R|)$ neighboring ASes, the amount of path leakage only slightly reduces in a majority of cases in comparison with the no minimization experiment. Compared to these two methods, our greedy algorithm reduces the number of path leakage significantly; however, the amount of path leakage is still enormous. For example, in 80% of cases, more than 10K of ASes receive the BGP poisoning message.

VI. Adaptive Link Flooding Attack versus Adaptive RAC Defense

In this section, we show that as a result of path leakage, the RAC defense becomes vulnerable to a new adaptive attack, which we call the *detour-learning* attack (§VI-A). In this attack, the adversary exploits the visibility of the path leakage in the network to learn the establishment of a new detour path in *real-time*. Once the new detour path of the critical flow is known to the adversary, she can immediately congest it and denies the goal RAC defense (i.e., rerouting critical flows to an uncongested path).

Moreover, we show that the RAC defense is not able to react against this detour-learning attack (e.g., switching from a non-isolated detour path to another) due to the long delay caused by the well-known *path hunting* problem when it withdraws an established detour path (§VI-B).

A. Detour-Learning Attack Based on Path Leakage

Adversary model. We first define our adversary model, i.e., the goals and capabilities of the detour-learning attack as follows:

1. *Attack goals.* The adversary aims to detect a new detour path immediately (e.g., within a few seconds) once the detour path is established. Then, the adversary picks one link on the detected detour path and floods it to continuously congest the critical flow.
2. *Attack capabilities.* Similar to today's typical DDoS attackers, the adversary controls a botnet to congest the main link target (i.e., a link on the default path of the critical flow) as well as the new link target (i.e., a link on the detected detour path). We assume that the adversary does *not* have any routing capability (e.g., add/modify/remove any inter-domain routes). Thus, the adversary cannot learn any route changes (e.g., detour path establishment) directly from routing table changes in the BGP routers. Moreover, we assume that the adversary has no bots in the critical ASes.[4]
3. *Knowledge of the existence of RAC defense.* We assume that the adversary knows the existence of the RAC defense at a destination network D and its critical AS C via analyzing the BGP messages recorded by the public BGP datasets (e.g., RIPE [12], RouteView [14]); see Appendix D for our analysis on this assumption. The recorded RAC operations are *old* (e.g., 10 minutes or more) and thus not directly useful for learning the current detour path in real-time, but can be used to detect the existence of the RAC defense and infer its critical AS, which is persistent regardless of the changing detour paths.

Learning real-time detour path changes. We show that the adversary can easily detect in real-time that a new detour path has been established by proactively measuring path from her botnet. The accurate target-link selection for the continuous link-flooding attack, however, is non-trivial, as we will point out later. Finally, we propose a link selection algorithm that outputs the target-link accurately and efficiently.

[4]If an adversary controls bots in the critical ASes, learning new detour paths is reduced to a trivial forward route measurement task.

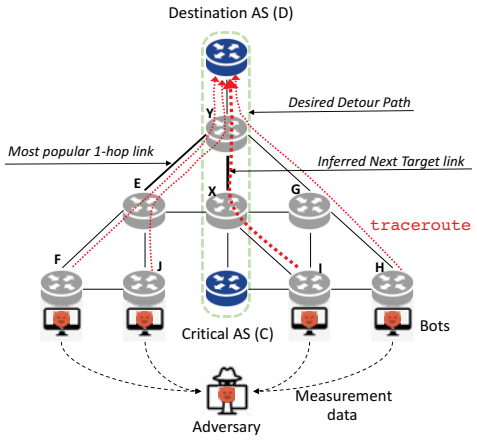


Figure 12: Detecting the detour path establishment in real time.

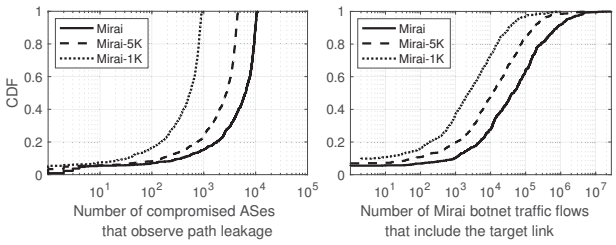
The adversary periodically (e.g., every second) measures the routes (e.g., via `traceroute`) from the ASes that contain her bots (or *compromised* ASes) to the destination AS. When a poisoning BGP UPDATE message arrives at a compromised AS, the adversary does not directly see the message. Yet, the adversary can detect the sudden changes in proactive route measurements from the bots. Note that `traceroute` measurement is highly effective for end-to-end route investigation because the majority of transit ISPs do allow `traceroute` measurements [20].

Ideally, the adversary may want to infer the *full* detour path from C to D. Then, with the full detour path, the adversary picks any link on the detour path to be the next target-link. However, each AS-level path measured by a bot may provide only some *partial views* of the detour path but no single path measurement gives the full detour path.[5] Achieving the accurate next target-link on the detour path thus turns out to be non-trivial and easily error prone. Figure 12 illustrates the challenges of the choice of the correct new target link.[6] As a naïve approach, the adversary may pick the most popular link from all compromised-to-destination route measurements to be the next target link (e.g., link (E-Y)). However, this does not guarantee the correct choice of the next target-link because each measurement has a different partial view of the detour path and thus the most popular link may happen to be the one that is not on the detour path. In Figure 12, there are more compromised-to-destination routes crossing the link (E-Y) than the link (Y-X) because AS E connects with more compromised ASes.

We propose a simple *distance-based target-link selection* algorithm to choose the link that is on the detour path. Our algorithm prefers the compromised-to-destination measurement that is made *closer* to the critical AS. We use the BGP path length between the compromised ASes and critical AS (see the BGP routing policy used in Section IV-A) as the distance metric. Our intuition is that the ASes in the close proximity may share similar AS paths to a destination AS. For example,

[5]Unless an adversary has bots in the critical AS C.

[6]Notice that the adversary would not pick the links that directly connect to the destination AS (e.g., link (Y-D)) because such attack becomes a traditional, direct server flooding attack that can be immediately detected and mitigated via traditional server-based solutions (e.g., scrubbing and filtering).



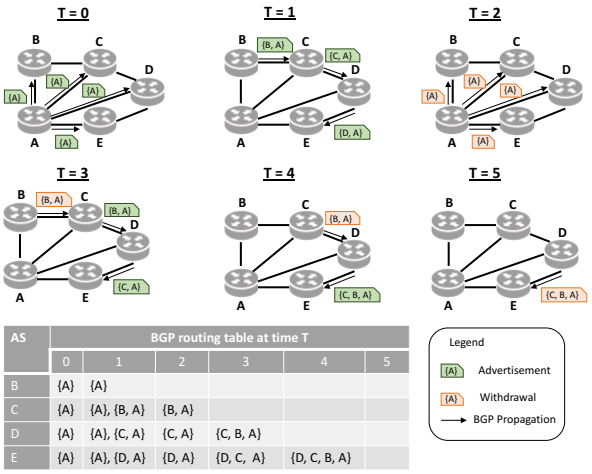
(a) (b)

Figure 13: Effectiveness of the detour-learning attack. (a) Distribution of the number of compromised ASes that observe path leakage. (b) Distribution of the number of default paths from a compromised AS to any other AS that contain the target link.

in Figure 12, I is closer to C than F, J and H in terms of BGP path length and its compromised-to-destination route shares more common links with the detour path (see the thicker `traceroute` measurement in red dashed line from I).

Evaluation of the detour-learning attack. We now evaluate the effectiveness of the detour-learning link flooding attack against the RAC defense. In our evaluation, we consider the Mirai botnet [15] as a real DDoS botnet to model the ASes that host compromised bots. The Mirai botnet is distributed in 11K ASes [8], representing a large-size DDoS botnet. To model small to mid-size botnets, we artificially create two more botnets by random sampling Mirai botnets: *Mirai-1K* and *Mirai-5K* botnet models have 1K and 5K compromised ASes, respectively.

1. *Detection of detour path establishment*. We evaluate the detection of the detour establishment by counting the number of compromised ASes in three botnet sets that receive the BGP poisoning message. We assume the RAC defender uses the greedy algorithm as proposed in Section V-B to minimize the path leakage.
   Figure 13a shows the distribution of the results in 1,000 cases. As we see from the Figure 13a, even with the botnet set including only 1,000 and 5,000 compromised ASes, the adversary is able to observe the path leakage in 96.0% and 99.0% of the cases, respectively. When the adversary gains control of the full Mirai botnet, she always sees the RAC operation in real-time, as there is at least one compromised AS receiving the BGP poisoning message.
2. *Accuracy of target-link selection algorithm.* We show that our distance-based target-link selection algorithm chooses the next target-link on the detour path with high accuracy. The target-link selection is said to be successful when there exists some compromised ASes observing the BGP poisoning message (hence detecting the detour path change) and the selected link is indeed on the detour path. The algorithm is evaluated with three mentioned botnet dataset, i.e., Mirai, Mirai-5K and Mirai-1K.
   The success rate of selecting the correct next target-link are 94.1%, 86.4% and 79.2% with the Mirai, Mirai-5K, and Mirai-1K, respectively. Even the adversary with 1K compromised ASes can find the correct next target-link accurately in the majority of cases.
3. *Link-flooding attack using botnet dataset.* Additionally, we would like to confirm if the correctly selected next target-links from the experiments with the three Mirai botnet sets (941, 864 and 792 links from Mirai, Mirai-5K, Mirai-1K sets, respectively) can be indeed targeted and flooded by link-flooding attacks with the Mirai botnet traffic.
   Figure 13b shows that in Mirai, Mirai-5K, Mirai-1K sets, there are 94.69%, 93.17% and 90.4% of the selected target-links are included in one or more Mirai botnet flows, respectively. Moreover, even when the adversary controls only 1,000 botnets, the vast majority ($> 83\%$) of the next link targets would have more than hundred (and easily several thousand) AS-level attack flows. Thus, the adversary can easily generate large numbers of legitimate-looking attack flows while congesting the next link target.



AS	BGP routing table at time T					
	0	1	2	3	4	5
B	{A}	{A}				
C	{A}	{A}, {B, A}	{B, A}			
D	{A}	{A}, {C, A}	{C, A}	{C, B, A}		
E	{A}	{A}, {D, A}	{D, A}	{D, C, A}	{D, C, B, A}	

Figure 14: An example of how *path hunting* occurs with BGP message withdrawal. Assume that each BGP message propagates one hop per unit time, the withdrawal message takes 4 units (T=2,3,4,5) to be converged while the advertisement takes only 2 units (T=0,1).

B. Slow Reaction of RAC Defense against Adaptive Attacks

To react against adaptive transit-link flooding attacks, the RAC defense may change the detour path nearly instantly whenever it observes congestion. However, as we show in the following evaluation, the RAC mechanism inflicts a long waiting time to switch to a new detour path (e.g., 85 seconds in even small topology size of 1,000 ASes). The main reason is that, before the RAC deployer can establish a new detour path, it is required to *undo* any changes made by its old BGP poisoning messages with the hole-punching prefixes. However, because the BGP poisoning message is leaked to multiple non-critical ASes, withdrawal messages in RAC suffer from the BGP *path hunting* problem [1], [19] and thus their convergences are significantly delayed. The chasm between such a slow reaction of the route changes and the fast-moving attacks makes the RAC defense nearly unusable in practice.

Path hunting and slow convergence. Path hunting is a well-known problem to BGP, which refers to a phenomenon where the withdrawal of a prefix causes other ASes to keep exploring for a route to reach that prefix until knowing all routes are invalid. In BGP, an AS selects the best path among

all paths advertised by its neighbors, which are kept in its routing table [7]. A new best path is re-computed and propagated whenever the routing table is changed. When the best path for a prefix is withdrawn, the AS will select and propagate the second best path found on its current routing table; however, this second best path might also be invalid (in the sense that it also depends on the withdrawn prefix) but yet to be withdrawn, because the withdrawal message takes longer to traverse the second best path. Once the second best path is again withdrawn, the AS has to explore the third best path and so on.

Figure 14 illustrates an example of how the path hunting occurs when AS A withdraws its prefix at $T = 2$. Consider AS C at $T = 3$, because the best path $\{A\}$ is withdrawn, AS C propagates its second best path $\{B, A\}$ to AS D, not before it knows the path $\{B, A\}$ is also withdrawn by AS B. This process is iterated with all other ASes until all possible paths from B, C, D, E to A are withdrawn, which consumes 4 time units in total (e.g., $T = 2, 3, 4, 5$), assuming messages propagate one hop per time unit. This convergence is significantly delayed, compared to the advertisement takes only 2 time units (e.g., $T = 0, 1$).

Because a non-isolated detour path leaks BGP poisoning messages to almost every AS in the Internet (see Section V-B), there exist many possible paths from each AS to the destination AS. Therefore, the withdrawal of a poisoning message will inevitably encounter the path hunting problem, causing an enormous number of update and withdrawal messages to be propagated until the network is converged. Worse yet, path hunting has a prolonged effect on RAC due to its use of hole punching and BGP poisoning. That is, because BGP prefers messages with hole-punching prefixes and shorter AS-path lengths, an AS would keep hunting for paths that are advertised by old RAC poisoning messages. Therefore, before a new RAC poisoning message can take effect, the RAC defender must completely and explicitly eradicate its old RAC poisoned paths from every routing table of all ASes that have path leakage.

Evaluation of path hunting in RAC. To investigate the convergence time when path hunting occurs in realistic settings, we use SSFNet [13] and add BGP poisoning and hole punching capabilities to it, instead of using the Chaos simulator because SSFNet can faithfully simulates BGP message propagation and processing based on RFCs. Such a fine-grained simulation is resource-consuming and thus prevents SSFNet from scaling to an Internet-size topology. To synthesize smaller-size realistic network topology graphs, we use a topology generation tool called BRITE [2] and adopt the Barabási-Albert model to generate a random scale-free network topology [16], which is similar to the structure of the Internet [9]. We fix the number of directional edges per node to be four according to CAIDA AS relationships statistics [3].

In each simulation run, a randomly selected destination AS will send a BGP poisoning message and then withdraw it. We measure the convergence time of those BGP advertisement and withdrawal messages, which is defined as the time elapsed since the message is fired until all routing tables reach a stable state. Besides, we also consider other factors that may influence BGP convergence, such as Route Flap Damping (RFD) which is designed to suppress excessive path changes in a short time [54] and Minimum Route Advertisement Interval (MRAI) which defines the time that an AS should wait before sending an update for the same prefix [49]. We consider the recommended practice where RFD is disabled [41] and MRAI is set to a small value, 0.5 seconds [25], [5], [11].[8]

[7]More specifically, a BGP speaker keeps its neighbors' advertisements in the RIB-in tables in its Route Information Base (RIB).

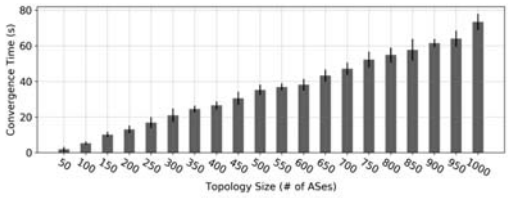


Figure 15: Convergence time of withdrawal messages increases significantly with the topology size.

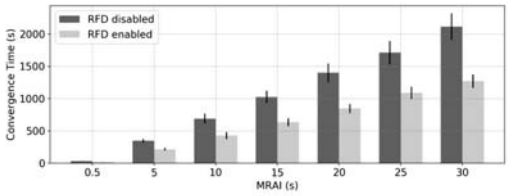


Figure 16: Convergence delay vs. MRAI value when RFD is enabled or disabled.

Figure 15 shows the convergence time of withdrawal messages on topology graphs ranging from 50 to 1,000 ASes. Each data point is computed over 10 randomly generated topology graphs. We see that the convergence time grows up to about 85 seconds on the 1000-AS topology. On the other hand, the convergence time of update messages are within 1.2–2.2s with a mean of 1.6s and a standard deviation of 0.2s, regardless of the topology size. These results confirm that RAC severely suffers from the path hunting problem, and an 85-second delay to establish a detour path is too slow to react to the detour-learning adversary. Moreover, because the 85-second delay is measured on a topology 60x smaller than the current Internet, we speculate that reaction time of RAC could be much worse in practice.

We also examine the convergence time of withdrawal messages under various MRAI and RFD configurations. In Figure 16, each data point is computed over 20 randomly generated topology graphs, each of which has 500 ASes. The simulation results confirm that the path hunting problem persists regardless of the MRAI and RFD configurations: the minimum convergence time when MRAI = $0.5s$ and RFD is enabled is 21s. Although withdrawal messages converge faster when RFD is enabled than disabled because RFD is designed to prevent frequent path changes, enabling RFD will exacerbate convergence delay of subsequent BGP poisoning messages [41].

[8]We acknowledge that setting MRAI to zero can evaluate the optimal case for the RAC defense, but simulation with MRAI = 0 is not supported in SSFNet, and thus we set MRAI to a small value to ensure our simulation completes within a reasonable timeout.

Table II: The measured and estimated number of ASes in the sets that either allow or drop BGP UPDATE messages based on their length.

	$\mathcal{A}_{[30,255)}$	$\mathcal{A}_{\geq 255}$	$(\mathcal{A}_{[30,255)} \setminus \mathcal{A}_{\geq 255})$ $(= \mathcal{D}_{\geq 255})$
Cardinality	654	156	**498**

VII. Best-Effort Detour Path Establishment For Detour Path Isolation

We have discussed, in Section V and Section VI, the consequences when the RAC deployer gives up achieving the path isolation property for the sake of the guaranteed detour path establishment. We now consider that the RAC deployer is aware of all the consequences of having non-isolated paths and thus chooses to achieve the path isolation property at the cost of the guaranteed detour path establishment. In the other words, the RAC deployer may attempt to establish an isolated path, but its establishment is *not* guaranteed because the upstream ASes may filter long BGP poisoning UPDATE messages (e.g., with an AS path length ≥ 255), which is often necessary for the path isolation property.

Although the detour path establishment is *not* guaranteed for each BGP poisoning message, the RAC deployer may make its best effort to find a possible detour path after multiple trials and errors. For example, it may try to establish different detour paths repeatedly until it finds one detour path that can be established. In this section, we ask *whether a RAC deployer can find any such detour path*, whose intermediate ASes all allow long (e.g., ≥ 255) BGP messages.

A. Identifying ASes that Would Filter Long BGP Messages

Our aim is to identify the set of ASes that would *drop* BGP messages with AS path length ≥ 255 but allow messages with AS path length < 255, which we denote as $\mathcal{D}_{\geq 255}$. Knowing the set $\mathcal{D}_{\geq 255}$ is crucial for our analysis because if a detour path contains one or more ASes in $\mathcal{D}_{\geq 255}$, it would be filtered out by them and thus the detour path cannot be formed. However, there is no direct way to measure $\mathcal{D}_{\geq 255}$ because only the BGP messages that have been allowed (and thus not dropped) are recorded in the public BGP dataset (e.g., RIPE [12], RouteView [14]). Thus, instead, we rely on an indirect way to estimate the ASes in $\mathcal{D}_{\geq 255}$.

Recall from Section IV-B, we have observed a fraction of abnormal BGP messages with an AS path length longer than 30 on the Internet. We can identify the ASes that accept and propagate such long messages by analyzing public datasets (e.g., RIPE [12]). Let us denote the set of ASes that *allow* BGP messages with an AS path length longer than 255 as $\mathcal{A}_{\geq 255}$, and denote the set of ASes that allow BGP messages with an AS path length in the range [30,255) as $\mathcal{A}_{[30,255)}$. We revisit the six-month period RIPE measurement data (see Section IV-B) to calculate the ASes in sets $\mathcal{A}_{\geq 255}$ and $\mathcal{A}_{[30,255)}$ and show their cardinalities in Table II.

We are interested in studying the set differences between $\mathcal{A}_{\geq 255}$ and $\mathcal{A}_{[30,255)}$, or $(\mathcal{A}_{[30,255)} \setminus \mathcal{A}_{\geq 255})$, which includes nearly 500 ASes. These ASes allow AS path length in the range [30, 255) but do *not* appear to accept any AS path length ≥ 255. There are two possible explanations: (1) these ASes also accept BGP messages with AS-path ≥ 255 but they do not receive messages with such long AS-path during the six-month period; and (2) these ASes drop such long BGP messages.

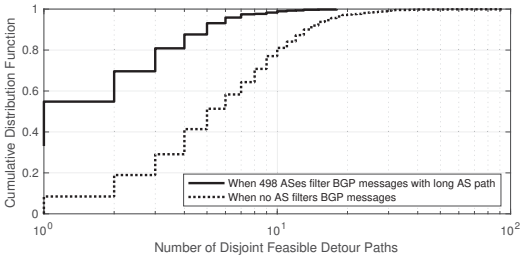


Figure 17: Number of disjoint feasible detour paths.

To evaluate how likely the first explanation holds true, we analyze some highly influential ASes in the current Internet (i.e., top-ranked ASes in the CAIDA AS rank[9]) and evaluate whether they would allow long BGP UPDATE messages with AS path length longer than 255 (i.e., belong to $\mathcal{A}_{\geq 255}$) or appear in the $(\mathcal{A}_{[30,255)} \setminus \mathcal{A}_{\geq 255})$ set. We find that some highly influential Tier-1 and Tier-2 ASes (e.g., Cogent, NTT, TATA) seem to allow long BGP messages whereas some others (e.g., Level 3, Telia, Singtel, AT&T) appear to not propagate any BGP messages with AS-path longer than 255, see Table III in Appendix E for more details. Given the dataset is collected in a six-month period, it is *highly unlikely* that such influential ASes do not receive any AS path ≥ 255 during this six-month observation period.

On the other hand, the second explanation is well-aligned with some conjectures we have mentioned in Section IV-B, e.g., it is advisable that the ISPs to *filter* out BGP messages with unnecessarily long AS path length. Moreover, Cisco routers by default are configured to drop messages with an AS path longer than 255 [46], [47]. Hence, we assume that the difference between $\mathcal{A}_{\geq 255}$ and $\mathcal{A}_{[30,255)}$ is because some ASes filter BGP messages when their AS path ≥ 255. That is,

$$\mathcal{D}_{\geq 255} = (\mathcal{A}_{[30,255)} \setminus \mathcal{A}_{\geq 255}). \quad (1)$$

B. Limited Feasible Detour Paths

Based on the set $\mathcal{D}_{\geq 255}$ inferred from Equation 1, we now evaluate all the enumerated detour paths for the selected 1,000 C–D pairs. Each pair may have more than one possible detour path and we evaluate if each detour path contains any AS in the set $\mathcal{D}_{\geq 255}$ (i.e., AS that would filter BGP messages with an AS path length ≥ 255). After removing such would-be infeasible detours (because BGP messages would be filtered), the percentage of the remaining feasible detour paths is less than 1% of total detour paths for the 80% of the tested cases.

Furthermore, we measure the number of *disjoint* feasible detour paths for each C–D pair. Note that when there exist more disjoint feasible detour paths, it is harder to congest the critical flows because a transit-link flooding attacker must flood at least one link per disjoint path simultaneously.[10] Figure 17

[9] http://as-rank.caida.org/

[10] Note that isolated detour paths have no path leakage and thus the detour-learning attacks are ineffective.

presents the distribution of the number of disjoint feasible detour paths when ASes in the set $\mathcal{D}_{\geq 255}$ filter BGP messages with long AS path (see the solid line). We see that in the vast majority of cases (e.g., 80%), an adversary only needs to flood at most three transit links to congest all the feasible detour paths. Worse, the RAC deployer has zero or only one disjoint feasible detour path in 55% of the cases, making it unusable. This strictly limits the RAC deployer's diversity to choose different detour paths. We can see that the filtering at the ASes in $\mathcal{D}_{\geq 255}$ significantly reduces the disjoint feasible detours by comparing the above result with the one where no AS filters BGP messages (see the dotted line).

Note that higher diversity of feasible detour paths is necessary for the reliable operation of the RAC defense because any established detour path will eventually become public knowledge found in the public BGP dataset (e.g., RIPE [12], RouteViews [14]) roughly 10-15 minutes after its first announcement and thus the RAC deployer should switch to another detour path frequently. Then, the RAC deployer would choose a new detour path from a small set (e.g., 3 or less for the majority of cases) of the disjoint feasible detour paths. With such a small set of disjoint feasible detour paths, the detour path change patterns of the RAC deployer are, unfortunately, extremely limited.

VIII. How to Make the RAC Defense Possible

We have shown that the RAC defense is impractical in the current Internet due to the incompatible BGP protocol and its practice. A natural question that arises is whether we can change the BGP protocol and its practice to make the RAC defense possible, and, if possible, what the cost of the changes would be.

From what we have observed, we can make the RAC defense possible by (1) allowing the BGP protocol to include much longer AS paths than its current maximum 2,034 [52] because the current maximum is not enough for many RAC detours (see Section IV-A) and (2) forcing all the BGP routers to accept BGP UPDATE messages *regardless* of their AS path length. However, these changes are not trivial. Making changes to the standardized BGP protocol and achieving large-scale deployment of the changes may require several years of effort. Moreover, transit ASes may not have clear incentives to accept the new changes after all because they do not directly benefit from the RAC defense.

Worse yet, when the Internet is forced to support it, the RAC's rerouting capability can be easily misused for malicious purposes. For example, a destination AS with large incoming traffic volume can control the traffic volume of its upstream transit ASes arbitrarily and make their operation unreliable. Route manipulation can also be launched by transit ASes for eavesdropping of critical flows. Since the main scope of this paper is the evaluation of the feasibility of the RAC defense but not its potential misuses, we leave more detailed analysis of these misuses for future work.

IX. Conclusion

Our in-depth study on a recent, intriguing rerouting-based defense proposal [51] against server/link-flooding attacks shows that a rerouting-based DDoS defense is fundamentally challenging because: (1) the current inter-domain routing protocol (i.e, BGP) is not expressive enough to support fine-grained control over inbound traffic, either directly or indirectly; and (2) one may attempt to abuse some routing features that are not intended for rerouting (e.g., loop detection) to achieve dynamic detouring of selected critical flows, but such an ad hoc approach only makes an unreliable solution due to the inconsistency among protocol specifications, implementations, and community's best current practice.

Our analysis has led us to learn several important *methodological lessons*:

1. Any new defense proposal against flooding attacks designed for *immediate deployment* must consider real-world constraints imposed by implementations, ISP operations, and legal consequences [50], [7], [6], not just the basic feasibility analysis based on protocol specifications;
2. It is of utmost importance to conduct a *rigorous security analysis* on any DDoS defense proposal. Realistic adversary capabilities should be assumed (e.g., knowing the defense strategies of the targeted networks; see Section VI) and the desired defense properties should be defined explicitly (e.g., the *invisibility* of detour paths; see Section VI-A); and
3. New proposals interacting with large systems and protocols, such as BGP, must demonstrate a *comprehensive evaluation* with complementing evaluation tools (e.g., SSFNet [13]) along with real experiments.

Our study on the (in)feasibility of the RAC defense confirms the previous conclusion of the major literature on this topic — strong bandwidth and path isolation against DDoS attacks must be considered as a *main security feature by design*. We hope that our findings will renew our quest to moving towards new DDoS-resilient Internet architectures.

Acknowledgments

We thank the anonymous reviewers of this paper and our shepherd Zhiyun Qian for their helpful feedback. This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its Corporate Laboratory@University Scheme, National University of Singapore, Singapore Telecommunications Ltd, and the Ministry of Science and Technology of Taiwan under grant MOST 107-2636-E-002-005-.

References

[1] "BGP Stability Improvements draft-li-bgp-stability-01." [Online]. Available: https://tools.ietf.org/html/draft-li-bgp-stability-01#section-1.3.1

[2] "BRITE." [Online]. Available: http://www.cs.bu.edu/brite

[3] "CAIDA Inferred AS Relationships Dataset." [Online]. Available: http://www.caida.org/data/as-relationships/

[4] "Chaos: BGP-4 Simulator." [Online]. Available: https://github.com/VolSec/chaos

[5] "Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide, Release 6.x." [Online]. Available: https://www.cisco.com/c/en/us/td/docs/switches/datacenter/nexus9000/sw/6-x/unicast/configuration/guide/l3_cli_nxos/l3_bgp.html

[6] "DDoS Mitigation Firm Has History of Hijacks." [Online]. Available: https://krebsonsecurity.com/tag/bgp-hijacking/

[7] "[nanog] Advice in dealing with BGP prefix hijacking." [Online]. Available: https://www.nanog.org/mailinglist/mailarchives/old_archive/2008-09/msg01006.html

[8] "Netlab360's Mirai Scanner." [Online]. Available: http://data.netlab.360.com/mirai-scanner/

[9] "Network Science: The Barabási-Albert model." [Online]. Available: http://barabasi.com/f/622.pdf

[10] "PEERING: The BGP Testbed." [Online]. Available: https://peering.usc.edu/

[11] "Quagga Routing Software - News: Quagga 1.2.0 has been released." [Online]. Available: https://savannah.nongnu.org/forum/forum.php?forum_id=8799

[12] "RIPE Network Coordination Centre." [Online]. Available: https://www.ripe.net/

[13] "SSFNet." [Online]. Available: http://ssfnet.org

[14] "University of Oregon Route Views Project." [Online]. Available: http://www.routeviews.org/routeviews/

[15] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J. A. Halderman, L. Invernizzi, M. Kallitsis *et al.*, "Understanding the mirai botnet," in *USENIX Security*, 2017.

[16] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *science*, 1999.

[17] C. Basescu, R. M. Reischuk, P. Szalachowski, A. Perrig, Y. Zhang, H.-C. Hsiao, A. Kubota, and J. Urakawa, "SIBRA: Scalable internet bandwidth reservation architecture," *Proc. NDSS*, 2016.

[18] P. Bright, "Can a DDoS break the Internet? Sure... just not all of it," *Ars Technica*, 2013.

[19] J. Chandrashekar, Z. Duan, Z.-L. Zhang, and J. Krasky, "Limiting path exploration in BGP," in *Proc. IEEE INFOCOM*, 2005.

[20] B. Donnet, M. Luckie, P. Mérindol, and J.-J. Pansiot, "Revealing MPLS tunnels obscured from traceroute," *ACM SIGCOMM CCR*, 2012.

[21] J. Durand, I. Pepelnjak, and G. Doering, "BGP operations and security," in *IETF RFC 7454 - Best Current Practice*, 2015.

[22] D. Dy, "Cisco Community: long bgp asn prepending," 2009. [Online]. Available: https://community.cisco.com/t5/routing/long-bgp-asn-prepending/td-p/1301441

[23] S. K. Fayaz, Y. Tobioka, V. Sekar, and M. Bailey, "Bohatei: Flexible and Elastic DDoS Defense," in *Proc. USENIX Security*, 2015.

[24] L. Gao, "On inferring autonomous system relationships in the Internet," *IEEE/ACM TON*, 2001.

[25] P. Gill, M. Schapira, and S. Goldberg, "A survey of interdomain routing policies," *ACM SIGCOMM CCR*, 2013.

[26] V. D. Gligor, "Guaranteeing access in spite of distributed service-flooding attacks," in *International Workshop on Security Protocols*, 2003.

[27] D. Goodin, "How extorted e-mail provider got back online after crippling DDoS attack," *Ars Technica*, 2015.

[28] A. Hayrapetyan, D. Kempe, M. Pál, and Z. Svitkina, "Unbalanced graph cuts," in *European Symposium on Algorithms*, 2005.

[29] W. Herrin, "NANOG-mailing-list: Long BGP AS paths," 2017. [Online]. Available: https://mailman.nanog.org/pipermail/nanog/2017-September/092536.html

[30] H.-C. Hsiao, T. H.-J. Kim, S. Yoo, X. Zhang, S. B. Lee, V. Gligor, and A. Perrig, "STRIDE: sanctuary trail–refuge from internet DDoS entrapment," in *Proc. ACM Asia CCS*, 2013.

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

[32] M. S. Kang, S. B. Lee, and V. D. Gligor, "The Crossfire Attack," in *Proc. IEEE S&P*, 2013.

[33] E. Katz-Bassett, C. Scott, D. R. Choffnes, Í. Cunha, V. Valancius, N. Feamster, H. V. Madhyastha, T. Anderson, and A. Krishnamurthy, "LIFEGUARD: Practical Repair of Persistent Route Failures," in *Proc. ACM SIGCOMM*, 2012.

[34] Y.-M. Ke, C.-W. Chen, H.-C. Hsiao, A. Perrig, and V. Sekar, "CICADAS: congesting the internet with coordinated and decentralized pulsating attacks," in *Proc. ACM Asia CCS*, 2016.

[35] M. Kührer, T. Hupperich, C. Rossow, and T. Holz, "Exit from Hell? Reducing the Impact of Amplification DDoS Attacks," in *Proc. USENIX Security*, 2014.

[36] S. B. Lee, M. S. Kang, and V. D. Gligor, "CoDef: collaborative defense against large-scale link-flooding attacks," in *Proc. CoNEXT*, 2013.

[37] M. Lepinski and S. Kent, "An infrastructure to support secure internet routing," Tech. Rep., 2012.

[38] M. Liotta, "NANOG-mailing-list: anyone else seeing very long AS paths?" 2009. [Online]. Available: https://mailman.nanog.org/pipermail/nanog/2009-February/007941.html

[39] Z. Liu, H. Jin, Y.-C. Hu, and M. Bailey, "Middlepolice: Toward enforcing destination-defined policies in the middle of the internet," in *Proc. ACM SIGSAC CCS*, 2016.

[40] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling high bandwidth aggregates in the network," *ACM SIGCOMM CCR*, 2002.

[41] Z. M. Mao, R. Govindan, G. Varghese, and R. H. Katz, "Route flap damping exacerbates internet routing convergence," in *ACM SIGCOMM CCR*, 2002.

[42] D. Massameno, "Secret CEF Attributes Part 6, The BGP Connection," 2014. [Online]. Available: https://packetpushers.net/secretcef6-bgp/

[43] R. Meier, P. Tsankov, V. Lenders, L. Vanbever, and M. Vechev, "NetHide: Secure and Practical Network Topology Obfuscation," in *Proc. USENIX Security*, 2018.

[44] J. Mirkovic and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," *ACM SIGCOMM CCR*, 2004.

[45] B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. Maggs, and Y.-C. Hu, "Portcullis: protecting connection setup from Denial-of-Capability attacks," *Proc. ACM SIGCOMM*, 2007.

[46] I. Pepelnjak, "Limit the maximum BGP AS-path length," 2009. [Online]. Available: http://wiki.nil.com/Limit_the_maximum_BGP_AS-path_length

[47] Pepelnjak, Ivan, "Oversized AS paths: Cisco IOS bug details," 2009. [Online]. Available: https://blog.ipspace.net/2009/02/oversized-as-paths-cisco-ios-bug.html

[48] R. Rasti, M. Murthy, N. Weaver, and V. Paxson, "Temporal lensing and its application in pulsing denial-of-service attacks," in *Proc. IEEE S&P*, 2015.

[49] Y. Rekhter, T. Li, and S. Hares, "A border gateway protocol 4 (BGP-4)," in *IETF RFC 4271*, 2005.

[50] F. Sanchez and Z. Duan, "Region-based bgp announcement filtering for improved bgp security," in *Proc. ACM Asia CCS*, 2010.

[51] J. M. Smith and M. Schuchard, "Routing Around Congestion: Defeating DDoS Attacks and Adverse Network Conditions via Reactive BGP Routing," in *Proc. IEEE S&P*, 2018.

[52] P. Smith, "BGP Best Practices," in *RIPE NCC Regional Meeting*, 2006.

[53] A. Studer and A. Perrig, "The coremelt attack," in *ESORICS*, 2009.

[54] C. Villamizar, R. Chandra, and R. Govindan, "BGP route flap damping," Tech. Rep., 1998.

[55] E. Vyncke and S. Hogg, "IPv6 Internet Security for Your Network," 2009. [Online]. Available: http://www.ciscopress.com/articles/article.asp?p=1312796&seqNum=3

[56] M. Winther, "Tier 1 ISPs: What they are and why they are important," *IDC White Paper*, 2006.

[57] L. Xue, X. Luo, E. W. Chan, and X. Zhan, "Towards Detecting Target Link Flooding Attack," in *Proc. USENIX LISA*, 2014.

[58] A. Yaar, A. Perrig, and D. Song, "SIFF: A stateless Internet flow filter to mitigate DDoS flooding attacks," in *Proc. IEEE S&P*, 2004.

[59] X. Yang, D. Wetherall, and T. Anderson, "A DoS-limiting network architecture," in *Proc. ACM SIGCOMM CCR*, 2005.

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

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

APPENDIX A
TIER-1 AS ON ALL POSSIBLE DETOUR PATHS

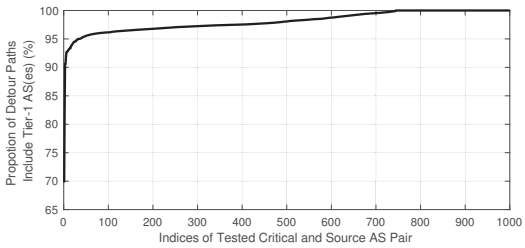


Figure 18: Number of detour paths include one or more tier-1 ASes.

We have counted the number of tier-1 ASes on the 1,000 detour paths with the least number of neighbors. We also wish to evaluate other possible detour paths between 1,000 C–D pairs as well.

Figure 18 shows the ratio of the detour paths that include at least one tier-1 AS in 1,000 cases. About 25% of the tested C–D pairs show that *all* their detour paths incur the large number of ASes to be poisoned as they include one or more tier-1 ASes. In nearly the rest of 75% of the cases, at least 95% of the all possible routes between C and D include at least one high-degree AS. To this end, we can see that the critical traffic will usually be transited at some major ASes regardless of which detour path is chosen. Hence, to achieve an isolated path for its critical flows, the RAC deployer usually has to poison hundreds to thousands of ASes.

APPENDIX B
MODELING PROBLEM [P1] TO PROBLEM MINSBCC

We first briefly review the general version of the MinSBCC problem [28] and then present the modeling from problem **[P1]** to MinSBCC. Interested readers may refer to the original paper of the MinSBCC problem [28] to see its full description and proofs of NP-Completeness.

We describe the generalization of the MinSBCC problem as follows:

The generalized MinSBCC problem. Given a graph $G = (V, E)$ where each edge $e \in E$ has a capacity c_e and each node $v \in V$ is assigned a weight w_v, a source node s, a sink node t and a capacity bound B. The objective is to find an s–t cut $(S, \overline{S}), s \in S$ within the budget B such that the total node weight $\sum_{v \in S} w_v$ is minimal.

We now model the problem **[P1]** following the MinSBCC problem. In particular, we consider the destination AS D as the source node s and the propagation of a BGP message sent from D to all other ASes forms a directed graph G. While we keep ASes on the detour path $R = \{C, R_1, R_2, \cdots, R_n, D\}$ as single nodes, we model each AS A that is not on the detour path with two nodes, A_{in} and A_{out}. All messages received by AS A are modeled as the edges going into A_{in} and all messages sent out by A are modeled as the edges going out from A_{out}. A_{in} and A_{out} have an edge with the capacity of 1 between them. All other edges are assigned to have the capacity of the infinity value so that the s–t cut will only remove the edges between A_{in} and A_{out} nodes. In other words, only the ASes not on the detour path can be poisoned.

We assign the weight for A_{in} and A_{out} nodes as 0 and 1, respectively. This means if AS A has path leakage, both nodes A_{in} and A_{out} will be included in set S and the total node weight $\sum_{v \in S} w_v$ increases by 1. If AS A does not have path leakage (e.g., because it is poisoned), only node A_{in} or no node is included in set S and the total node weight $\sum_{v \in S} w_v$ is unchanged. Also, all the nodes corresponding to the ASes on the detour path have the weight of 0 so that they will be included in S. Since the MinSBCC problem aims to minimize $\sum_{v \in S} w_v$, it is equivalent to the goal of minimizing number of path leakage in problem **[P1]**.

APPENDIX C
A GREEDY ALGORITHM TO MINIMIZE PATH LEAKAGE

Our greedy algorithm contains two phases as follows.

1. *Calculate isolating sets of all ASes.* We first enumerate all paths between each AS in the network and the destination AS and then calculate its isolating set by taking note of all ASes in $\mathcal{Q}$ that appear in the paths. We implement this process efficiently using recursion with memorization. Assuming we have k ASes in total that can have path leakage, we now have k isolating sets: $S_1, S_2, \cdots, S_k$.
2. *Choose AS to be poisoned.* We present the multi-round process of picking neighboring ASes to poison in Algorithm 1. In general, in each round, the algorithm selects the smallest isolating set, say S_i, and includes it in the final poisoning set $\mathcal{P}$ (see Line 3-10). By doing so, we have completely prevented the path leakage at AS_i. Because the chosen isolating set S_i may overlap with other isolating sets, we have to remove the ASes that are already chosen to be poisoned in those sets (see Line 15-17). Finally, the algorithm stops when we have selected the sufficient number of ASes to be poisoned (see Line 11-13).

APPENDIX D
THE COVERAGE OF PUBLIC BGP DATASETS

When the destination AS D performs the RAC defense (i.e., establishes a detour path in response to link-flooding attacks), it broadcasts one or more BGP poisoning messages. These poisoning messages are in fact recorded in the public BGP datasets(e.g., RIPE [12], RouteView [14]) approximately 10-20 minutes after they are exchanged. There currently exist 528 ASes that send the received BGP messages to the RIPE and RouteView data collection points. Our analysis shows that more than 99.95% of all possible detour paths for the selected 1,000 C–D pairs are monitored by the 528 ASes. As a consequence, any poisoning message becomes public knowledge that can be easily accessed from the two BGP datasets. From the BGP poisoning pattern (i.e., the set of ASes the RAC deployer wants to avoid), one can easily infer the intended detour path. For example, in Figure 10, the BGP UPDATE message poisons ASes E and H, and many other direct neighbor ASes of X, Y, and Z, but does not poison ASes C, X, Y, and Z. This provides a strong indication that the BGP message is designed to establish a detour path: $\{C$, X, Y, Z, $D\}$.

Algorithm 1 Choosing poisoning ASes

Require: $S = \{S_1, S_2, \cdots, S_k\}$: the set of all k isolating sets.
B: the maximum number of ASes that can be poisoned.
Ensure: $\mathcal{P}$: set of ASes we choose to poison.

```
procedure ChoosePoisoningASes
    P ← []
    while True do
        MinSet ← S[0]
        for all S_i ∈ S do                ▷ Choose the lowest-cost set.
            if |S_i| ≤ |MinSet| then
                MinSet ← S_i
            end if
        end for
        P' ← P ∪ MinSet
        if |P'| ≥ B then
            break                         ▷ Terminate when exceeding bound.
        end if
        S ← S \ [MinSet]                  ▷ Remove the chosen set.
        for all S_i ∈ S do                ▷ Update the remaining sets.
            S_i ← S_i \ MinSet
        end for
        P ← P'
    end while
    return P
end procedure
```

Appendix E
Inferred Filtering Practice at Top ASes

We present the list of top 20 rank ASes and whether they belong to set $\mathcal{A}_{\geq 255}$ or set $(\mathcal{A}_{[30,255)} \setminus \mathcal{A}_{\geq 255})$ in Table III.

Table III: Whether the CAIDA top 20 rank ASes filter BGP UPDATE messages with AS path ≥ 255. The ASes are ordered according to their ranking.

ASN	AS name	$\in \mathcal{A}_{\geq 255}$	$\in (\mathcal{A}_{[30,255)} \setminus \mathcal{A}_{\geq 255})$
3356	Level 3		✓
1299	Telia Company AB		✓
174	Cogent Comm.	✓	
2914	NTT America, Inc.	✓	
3257	GTT Comm.	✓	
6762	Telecom Italia		✓
6453	TATA Comm.	✓	
6939	Hurricane Electric	✓	
3491	PCCW Global	✓	
3549	Level 3		✓
1273	Vodafone Group	✓	
6461	Zayo Bandwidth		✓
9002	RETN Limited	✓	
209	Qwest Comm.		✓
12956	Telefonica Int	✓	
3320	Deutsche Tel.	✓	
7473	Singapore Tel.		✓
12389	PJSC Rostelecom	✓	
7018	AT&T		✓
20485	TransTeleCom		✓