IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 11 January 2023, accepted 10 March 2023, date of publication 23 March 2023, date of current version 30 March 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3261128

RESEARCH ARTICLE

On the Effectiveness of BGP Hijackers That Evade Public Route Collectors

ALEXANDROS MILOLIDAKIS[1], **TOBIAS BÜHLER**[2], **KUNYU WANG**[1], **MARCO CHIESA**[1], **LAURENT VANBEVER**[2], **AND STEFANO VISSICCHIO**[3]

[1]KTH Royal Institute of Technology, 114 28 Stockholm, Sweden
[2]ETH Zürich, 8092 Zürich, Switzerland
[3]Department of Computer Science, University College London (UCL), WC1E 6BT London, U.K.

Corresponding author: Alexandros Milolidakis (miloli@kth.se)

This work was supported in part by the Swedish Foundation for Strategic Research under Grant 64455, and in part by the KTH Digital Futures.

ABSTRACT Routing hijack attacks have plagued the Internet for decades. After many failed mitigation attempts, recent Internet-wide BGP monitoring infrastructures relying on distributed route collection systems, called route collectors, give us hope that future monitor systems can quickly detect and ultimately mitigate hijacks. In this paper, we investigate the effectiveness of public route collectors with respect to future attackers deliberately engineering longer hijacks to avoid being recorded by route collectors. Our extensive simulations (and attacks we device) show that monitor-based systems may be unable to observe many carefully crafted hijacks diverting traffic from thousands of ASes. Hijackers could predict whether their attacks would propagate to some BGP feeders (i.e., monitors) of public route collectors. Then, manipulate BGP route propagation so that the attack never reaches those monitors. This observation remains true when considering plausible future Internet topologies, with more IXP links and up to 4 times more monitors peering with route collectors. We then evaluate the feasibility of performing hijacks not observed by route collectors in the real-world. We experiment with two classifiers to predict the monitors that are dangerous to report the attack to route collectors, one based on monitor proximities (i.e., shortest path lengths) and another based on Gao-Rexford routing policies. We show that a proximity-based classifier could be sufficient for the hijacker to identify all dangerous monitors for hijacks announced to peer-to-peer neighbors. For hijacks announced to transit networks, a Gao-Rexford classifier reduces wrong inferences by $\geq$91% without introducing new misclassifications for existing dangerous monitors.

INDEX TERMS BGP, BGP hijacking, stealthy IP prefix hijacking, inter-domain routing, routing policies, route collectors, forged AS path, BGP monitoring, BGPStream.

I. INTRODUCTION

Routing hijacks keep affecting industry (including Google, Amazon, Apple, Microsoft) [1], [2], financial platforms (e.g., the Bitcoin network) [3], security authorities [4], Internet services [5], governments [6], and citizens [7]. 775 (830) suspicious BGP hijack (route leak) incidents have been documented in 2021 [8], and more than 2200 (1200) suspicious BGP hijack (route leak) incidents have been documented in 2020, a 30% hijack increase from 2019 [9].[1] This is mainly because BGP allows attackers to inject arbitrary information in the Internet routing system. By falsely claiming ownership of IP prefixes, malicious networks can divert, eavesdrop, store, and possibly modify traffic in so-called *interception attacks*.[2]

The associate editor coordinating the review of this manuscript and approving it for publication was Salekul Islam.

[1]While the references distinguish between route leaks and BGP hijacks, we consider them analogous terms, both classified under the category of Type-N attacks, which we will introduce formally in Section III-A.

[2]BGP hijacks can be used to drop traffic as well. We focus on interception attacks as they are harder to detect for the victim.

Many attempts to prevent such attacks have proven ineffective. Notably, experience shows that relying on Public key Infrastructures to prevent hijackers' BGP messages from propagating may not always work well in practice [10], [11]. Security-enhancing protocols, such as RPKI [12], are slow to deploy and, although effective against accidental BGP leaks, they are generally not effective against all possible hijacks – as we also show in this paper.

Luckily, public Internet-wide BGP monitoring infrastructures give us hope to build more effective defenses against BGP hijackers. For hijacks to intercept traffic, hijackers' BGP messages must be visible to networks transporting traffic. Hence, feeds from a battery of geographically distributed BGP monitors can be used to expose hijacks while they are happening.

For this reason, multiple of the current commercially deployed and nonprofit monitoring solutions rely on *route collectors* [13], [14], [15], [16], [17], [18], [19]), i.e., BGP speaking devices that disclose any route they receive from their peering neighbors. The Public Route Collector Infrastructure, namely RIPE's Routing Information Service (RIPE-RIS) [20] and Oregon's Routeviews project [21], consists of multiple route collectors distributed throughout the world. Network volunteers, that we call monitors, peer with these route collectors to publicly disclose their BGP best routes. Fast detection can then ideally trigger prompt mitigation, e.g., enabling victims to contact other networks' operators or propagate more specific routes mitigating hijacks ideally within a minute [16].

In this paper, we investigate the effectiveness of today's public route collectors in the face of sophisticated interception attackers that adapt the routes they announce to avoid public monitor-based devices. As we show, hijackers can indeed exploit well-known techniques, such as AS-path poisoning or AS-path prepending, to circumvent their BGP messages from propagating to route collectors. These techniques, however, tend to reduce the hijacks' impact because the attackers' routes become longer, and hence generally less preferred by networks without monitors, too. We, therefore, ask the following question: Can hijackers avoid all the public RIPE-RIS and Routeviews monitors while still diverting traffic from a significant portion of the Internet?

Past work [16] has shown in simulations that hijacks that affect more than 2% of the Internet are always visible by public route collectors [16]. While this percentage holds indeed true for naively designed attacks, unfortunately, our simulations show that many carefully designed attacks could evade traditional monitor-based systems while also attracting traffic from potentially thousands of ASes. We extend a BGP simulator (which we will publicly release) and use it to quantify the best- and worst-case attack surface. For the worst-case threat scenario, we consider an *omniscient* attacker with complete knowledge of the inter-domain routing policies (import and export) that ASes use to install and advertise routes from their routing tables. Such an attacker can *accurately predict* which attacks will propagate to the monitors and is therefore capable of modifying the attacks appropriately so that they do not reach the route collectors. For the best-case threat scenario, we consider a *realistic* attacker that lacks the above elaborate knowledge, but is still capable of *fairly estimating* which attacks will propagate to the route collectors from the routing information the monitors themselves report.

Having quantified and measured the best- and worst-case attack surface in the traditional CAIDA Internet topology [22], we additionally measure the attack surface of hijackers in plausible future Internet topologies. We consider two factors. On the one hand, we model the (hoped) growth of public BGP route collector infrastructures by disclosing routes from more participating ASes. Although additional monitors decrease the attack surface, we discover that many hijackers could respond by adapting their attacks to remain undetected even in topologies with four times more monitor ASes than today. On the other hand, we consider topologies with an increasing number of IXP peer-to-peer links, as resulting from further Internet topology flattening. As we show, flatter topologies are even *more prone* to hijacks invisible to route collectors, as more peer-to-peer paths exist that are less likely to include monitors, and therefore be reported to route collectors.

Having evaluated the feasibility of stealthy attacks in simulations, we proceed to ethically conduct real-world hijacks to evaluate the feasibility of stealthy attacks in the real Internet using the PEERING Testbed [23]. A key feature that would enable completely stealthy attacks to route collectors is the capability of the hijacker to accurately detect the monitors that will report the attack to route collectors. Therefore, we experiment with a binary *classifier* to distinguish monitors into two categories: (*i*) those that propagate the attack to the route collectors and (*ii*) those that will not. From the viewpoint of the attacker, the first category are *dangerous monitors* that the hijacker needs to avoid. The second category are *safe monitors* that the hijacker does not need to react against.

We design and compare two types of binary classifiers: one based on proximities (i.e., shortest AS-path lengths similar to the realistic hijacker in simulations) and one based on BGP policies (i.e., which override shortest path lengths). We compare the success of those classifiers on two occasions: when the hijacker exports the attack to peer-to-peer neighbors versus when the hijacker exports to transits.[3] While we could not create a stealthy attack due to the Testbed restrictions on AS-path lengths and a variety of other ethical concerns (discussed in Appendix A), the high sensitivity values of the second classifier to detect dangerous monitors, i.e., ≥ 0.95 for transits and sometimes equal to 1 for peers (see Table 8), suggest that parts of the Internet potentially exist that are vulnerable to stealthy hijacking.

Contributions:

- We are the first ones to rigorously investigate, both in simulations and in the real-world, the feasibility of hijackers to avoid public route collectors. In that regard,

[3]The PEERING Testbed has no customers.

we present the limitations of traditional public route collector infrastructures and demonstrate how hijackers could take advantage of what route collectors publicly disclose to design attacks that do not propagate to route collectors.

- Our simulations (see Section VI) show that a hijacker with complete knowledge about the routing policies of other networks can *always* engineer an attack that is not visible to public route collectors. Compared to naive (baseline) hijackers that could not affect more than 2% of the Ases, omniscient hijackers could affect up to 11.7× more, 24.5× more, and 4.1× more ASes without being observed by public route collectors in the traditional CAIDA topology and future topologies with more peer-to-peer links and more monitors (respectively). While adding more monitors reduces the attack surface, it is hard to completely eliminate all large-scale stealthy attacks from hijackers that react once they become aware of the new monitors. On the other hand, adding more peer-to-peer links benefits such hijackers. While naive hijackers see barely noticeable improvement, realistic and omniscient attackers both increase their attack space the flatter the Internet topology.
- A realistic hijacker with no knowledge about routing policies, observing the best routes that monitors disclose to public collectors, is still able to fairly engineer less-preferred forged routes that do not propagate to route collectors. Compared to baseline hijackers, which are usually visible to route collectors, realistic hijackers were completely invisible for 62% of the simulations in the traditional topology. At the same time, hijacking up to 8.1× more, 22.7× more, and 2.9× more ASes in the traditional topology and future topologies with more peer-to-peer links and more monitors (respectively). Part of the reason for the high success of realistic hijackers is due to announcement made to peer-to-peer and customer related neighbors, contrary to announcements made to transit providers which are commonly visible (i.e., see Table 5).
- Our real-world analysis (see Section VII) indicates that stealthy hijacks to the public route collectors (RIPE-RIS and Routeviews) are likely feasible in the real-Internet. While we could not design a stealthy attack due to the limitations of the PEERING testbed, *both* our simulations and real-world findings show that a classifier based on proximities could be sufficient for the hijacker to correctly identify all dangerous monitors when announcing the attack to peer-to-peer neighbors.
- For hijacks announced to transits, we show that a classifier based on Gao-Rexford routing policies can reduce misclassifications by more than 91% compared to a proximity classifier. Although a classifier based on Gao-Rexford is not sufficient to accurately identify every dangerous monitor that exists in the topology, all our hijack experiments show that such a classifier could allow the hijacker to identify the majority of dangerous monitors without misclassifying any such monitors. As Table 9 shows, the Gao-Rexford classifier introduces *no* false negatives in *any* of our experiments.
- We will publish our binary classifier output and ground truth collected from the monitors of RIPE-RIS and routeviews during the PEERING Testbed experiments.[4] We believe that our dataset would benefit future research seeking to understand the reasons behind wrong inferences in route propagation to improve inter-domain route modeling.

The continuation of this paper is organized into the following sections: Section II explains the problem with traditional route collectors that enables hijackers to design stealthy attacks. Section III explains how hijackers could manipulate their announcements to avoid route collectors. Section IV presents the simulator that we use to design stealthy attacks and its limitations. Two different hijacking configuration strategies are presented in Section V: (*i*) an omniscient hijacker and (*ii*) a realistic hijacker. We use the omniscient hijacker to understand what is feasible from the perspective of an all-knowledgeable attacker and the realistic to understand what is feasible from a realistic attacker. Section VI compares our results for the omniscient and realistic hijackers against baseline (i.e., traditional) hijackers. Three topologies are considered: (*i*) the traditional CAIDA topology, (*ii*) flatter topologies with more peer-to-peer links, and finally, topologies with more monitors. Section VII evaluates the feasibility of stealthy hijackers in the real world. While due to a variety of ethical and limitation concerns announcing a stealthy attack in the real-world was not feasible, we explain the reasons behind the misclassifications and show that a simple classifier based on proximities could be sufficient to predict the monitors that will report the attack for announcements made to peer-to-peers. Section VIII summarizes our belief about the feasibility of stealthy attacks from the results we gathered in simulations and in the real-world. Then, we present our insights for defending route collectors against stealthy attacks and explain what hijackers could do better to design stealthier attacks. Finally, Section IX presents the related work, and Section X closes with the conclusions. We believe that our results will stimulate additional research on hijack detection solutions.

II. THE PROBLEM

This section motivates our research questions and explains the reasons why many carefully designed attacks could potentially avoid public route collectors. In sum, there are two reasons why a route may not propagate to a route collector, both unrelated to hijacking:

A. MONITORS ONLY EXPORT THEIR BGP-BEST ROUTES TO ROUTE COLLECTORS

Peers (so-called monitors) establish a BGP session with route collectors to communicate the BGP routes that are currently

[4]https://github.com/AlexMiloli/stealthy-hijacks

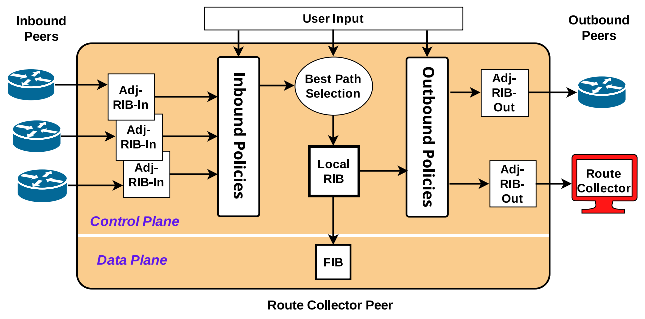


FIGURE 1. Logical model of a commercial BGP router propagating routes to the route collector.

being used to forward traffic towards possibly 100s of thousands of IP prefix destinations. Fig. 1 shows an example of how a monitor communicates the routes it learns from its inbound neighbors to the route collector. Based on the current version of BGP [24], the monitor first receives a BGP advertisement containing an AS-path and a prefix from its inbound neighbors. The monitor stores this unprocessed routing information in the *Adj-RIB-In* table. From that table, BGP's best path selection algorithm selects a *single best* AS-path per-prefix that satisfies the inbound filtering policies of the local node and stores it in the *Local RIB* table. Normally, AS-paths containing loops, i.e., routes which the local AS is already part of the AS-path, are excluded from this selection. This is the so-called loop-prevention mechanism of BGP. Finally, the prefix routes stored in the Adj-RIB-In table may propagate further to the corresponding *Adj-RIB-Out* table and outbound BGP peer dictated by the outbound filtering policies of the local node. Traditionally, monitors do not distinguish between usual outbound peers and outbound peers which are route collectors. Therefore, based on the pipeline of Fig. 1, monitors communicate only a single route per IP prefix (at a time) to the route collector, i.e., the one stored in the Local RIB (best route) even though they may know of multiple routes from their neighbors). A hijacker can take advantage of this propagating behavior, manipulating either the preference of the attack or exploiting BGP's loop-prevention mechanism by inserting bogus ASes into the AS-path (a technique known as *AS-path poisoning*), to cause the hijack to not propagate to the route collector.

B. MONITORS OBSERVE A LIMITED VIEW OF THE INTERNET [25], [26]

Zitong et al. [25] explain that only a limited number of BGP routes are visible to every route collector of the public infrastructure. In their study consisting of the 432K inter-domain links, only 17 500, i.e., 4%, of those links were visible across all monitors typically because most peer-to-peer routes propagate only within one (73%) or two AS hops (20%) away from the monitors. Therefore, malicious routing events, such as BGP hijacks, could remain hidden if they occur outside the topological visibility of the monitor. Specifically, a smart BGP hijacker plotting to hijack the prefix(es) of a target victim could first collect the current best routes that monitors report for its target to route collectors. This way, the hijacker might be able to forge a **less-preferred** hijack that the monitors would never select (as the best route) to propagate to public route collectors. These observations motivate the following research questions.

TABLE 1. Impact and visibility of BGP hijacks based on the announced prefix.

Announced Prefix	*Stealthiness (Control Plane)*	*Impact (Data Plane)*
More Specific	Low	High
Less Specific	Low	Low
Same Prefix (Type-N)	Higher with Higher N	Higher with Lower N

C. RESEARCH QUESTIONS

Can hijackers design attacks that are not preferred by monitors but still attract multiple other networks? How reliable are public route collectors in observing such hijacks? How many ASes could such hijackers affect while remaining stealthily?

To answer these research questions, we use a simulated model of the Internet where we ethically experiment with different stealthy BGP hijack strategies. From one day worth of BGP UPDATES (1st July 2021), we use BGPStream [27] to identify the peer ASes of RIPE-RIS and RouteViews collectors. We identify in total 483 peers, 367 of which actively report their routes within 5 minutes. We mark those 367 ASes as *the monitors* in our simulator that hijackers seek to avoid. We say that the hijack is *stealthy* in the simulated environment if none of the monitors propagates the attack to the route collectors.

III. HIJACK STRATEGIES

We now explain the attack strategies that hijackers could utilize to create stealthy attacks, i.e., forged AS-paths that do not propagate to route collectors. First, Section III-A briefly presents the taxonomy of BGP attacks according to the literature, focusing on the trade-offs between low (high) control-plane stealthiness and high (low) data-plane impact. Then, Section III-B explains the attack strategies that hijackers could utilize to create stealthy attacks. Finally, Section III-C presents the challenges in creating stealthy impactful attacks and our assumptions that better clarify the problem space investigated in this work.

A. ATTACK TAXONOMY

We briefly recall how attacks are taxonomized in the control plane by the way hijackers manipulate bogus BGP updates [16]. This manipulation commonly involves how the bogus prefix the hijacker announces compares to the genuine prefix announced by the victim (more specific, less specific, or same prefix) and the type of bogus AS-path the hijacker advertises (notation *Type-N* in Table 1, where $N \geq 0$ denotes the position of the hijacker on the bogus AS-path).

Announcing a more or less specific prefix than the victim would commonly cause the hijack to propagate *everywhere* in the control-plane as BGP would treat such a prefix independently. The visibility of the attack by the monitors would be high, and hence, the stealthiness low both for more specific and less specific announcements (first two rows in Table 1). On the contrary, the data-plane impact would be high for more specific attacks and low (practically none) for less specific attacks. This is because the data plane prefers more specific prefixes, causing more specific attacks to reroute all of the traffic destined to the affected prefix to the hijacker.

Announcing the same prefix as the victim would cause the hijacker to compete with the victim, hence splitting the Internet into two regions. The monitors in the *affected* region would report the forged route, while the monitors in the *unaffected* region would not observe it. In this setting, the stealthiness and the impact of the attack change according to the announced attack Type (third row in Table 1).

To illustrate this, Figs. 2 and 3 show from 2000 hijack simulations[5] the visibility of the hijacker by the monitors (Fig. 2) and the hijack impact (Fig. 3) among different attack Types, ranging from Type-1 to Type-5. In Type-1 attacks, the hijacker (H) claims to be a neighbor of the victim (V) by announcing the forged AS-path {H, V}. In Type-2 attacks, the hijacker claims to be two hops away from the victim, by announcing the forged path {H, AS1, V}, etc. As we notice from the figures, the longer the bogus AS-path (i.e., the higher the N) the stealthier the attack is in the control-plane but the less impactful the attack is in the data-plane. For example, when increasing the attack Type, the median number of monitors observing the hijack decreases from 101 to 40, 18, 10, and 6 monitors for Type-1, Type-2, Type-3, Type-4, and Type-5 attacks, respectively. Similarly, the median number of ASes affected by hijacks decreases from 18802 to 7382, 3033, 1763, and 1140 ASes for Type-1, Type-2, Type-3, Type-4, and Type-5 attacks, respectively. This trade-off between increased stealthiness but reduced impact illustrates the challenge of designing both a stealthy and an impactful attack, as we later present in Section III-C.

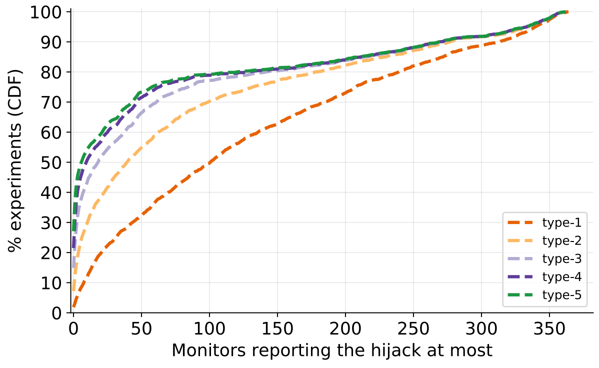


FIGURE 2. Number of monitors reporting the hijack per attack Type (2000 hijack simulations). Stealthiness increases the higher the attack Type.

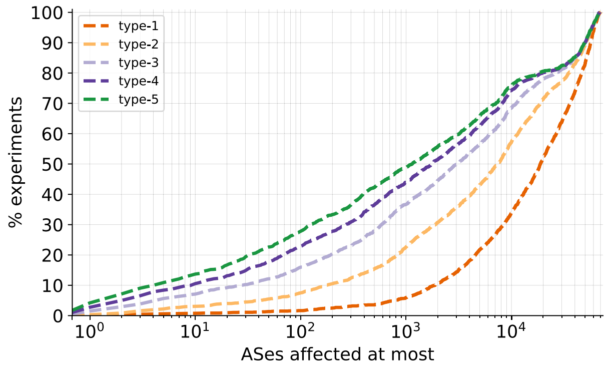


FIGURE 3. Number of hijack affected ASes per attack Type (2000 hijack simulations). Impact decreases the higher the attack Type.

B. STEALTHY ATTACK STRATEGIES

We now discuss how the hijacker could manipulate its BGP announcements to design an attack that does not propagate to route collectors.

1) ROUTE PROPAGATION

We briefly recall how routes propagate in BGP. In sum, route propagation over the Internet depends on the routing decisions of multiple BGP speaking devices. These decisions depend on how each AS operator has configured each device's local inbound filtering policies, local best AS-path selection preferences, and local outbound routing policies (see Fig. 1). Routing devices that prefer the hijacker's advertised route over the victim's advertised route would propagate the hijack further, potentially corrupting the RIB of other neighboring devices. Once the hijack corrupts the Local RIB of a monitoring device, the forged advertisement propagates to the route collector, which archives the route.[6] Although the hijacker has no control over how other ASes have configured the routing decisions of their devices, the hijacker can still influence the monitor preferences over which routes propagate to the route collectors by carefully adjusting:

- *The hijacked prefix:* As discussed in Table 1, announcing a more/less specific prefix than the victim would cause the hijack to propagate everywhere in the control plane while announcing the same prefix as the victim would cause the hijacker to compete with the victim. Naturally, same-prefix attacks are stealthier in the control-plane and harder to notice in the data-plane, as monitors (services) in the unaffected region would not observe (notice) the attack.
- *The AS-path length of the forged route:* The second step in the BGP path selection process (after local preferences) prefers routes with shorter AS-paths. Therefore, a hijacker announcing the same prefix as the victim could forge routes with longer AS-paths that are

[5]For details about the simulator see Section IV and VI.

[6]Rarely, the monitor may still decide to not propagate the hijack to the public collector. For more details, see "Assumption 5" in Section III-C.

less likely to get selected as the best routes to propagate to route collectors. Although longer AS-paths are less likely to be observed by route collectors, longer AS-paths show also a reduced data-plane impact. We face a trade-off between our desire to create a completely stealthy but still impactful attack. We define impact as the amount of hijack affected ASes.

- *The ASes included in the AS-path:* The BGP loop-prevention mechanism drops routes containing the ASN of the BGP speaker processing it. Attackers can exploit this mechanism to prevent forged routes from participating in the BGP decision process of monitors. Thus, preventing the hijack from propagating to the route collector.
- *The neighbors to which the hijack is exported:* Commonly, only a few neighbors are responsible for the hijack propagating to the monitors. Carefully forging specific per-neighbor announcements would allow the creation of shorter forged routes that could propagate further while potentially avoiding the monitors.

2) ATTACK STRATEGY

Table 1 indicates that announcing more or less specific hijacks would cause the attack to propagate throughout the control plane. We could suppress that using specific BGP community attributes [28]. However, as these attributes are not widely supported (see Section IX), we limit hijackers to only same-prefix attacks based on standard inter-domain traffic engineering operations, i.e., AS-path prepending, AS-path poisoning, and selective export of hijack announcements. To summarize, the hijacker's goal is to create hijacks that maximize the number of affected ASes while remaining hidden (stealthy) from the public route collectors. To do so, the hijacker uses its configured knowledge to forge *same-prefix* routes on a *per-neighbor* basis. In addition, the hijacker aims to maintain a route back to the victim once the hijack has been announced. This way the hijacker creates so-called interception attacks which do not lead to data-plane disruptions of the hijacked traffic. For more on how we use the above attack strategies to design stealthy attacks not visible to route collectors, see Section V.

C. THE CHALLENGE

Designing routing hijacks that affect a considerable part of the Internet while the hijack remains stealthy is not easy. Forged AS-paths need to propagate sufficiently *far* in the Internet to affect multiple ASes, yet sufficiently *close* so that they do not affect the monitors. Therefore, identifying the correct forged attack Type (i.e., the AS-path length) that would enable the above propagation is crucial for the hijacker to design as stealthy and impactful as possible attacks. For this, knowledge about which hijack Types each monitor would propagate to route collectors is crucial to identify and the focus of our attack design in Section V.

This problem of crafting impactful hijacks is challenging as various other work shows. For example, [29], [30], and [31] focus on the generation of interception attacks that maximize the number of affected networks, all showing the NP-hardness of the results. Part of the reason is that sometimes, counter-intuitively, more impactful attacks are possible if the attacker exports *longer* AS-paths or *no* AS-paths at all to some neighbors (for more such examples, see [29]). None of these works explicitly try to evade detection from route collectors. In Section V, to relax some of the complexity, we therefore develop a heuristic to generate *stealthy* interception attacks with the goal of propagating per-neighbor forged announcements as far as possible (i.e., as close to the monitors as possible), thus affecting a large number of networks while avoiding being reported by monitors to the route collector(s).

To further clarify the problem space that we consider in this work, we introduce the following five assumptions:

Assumption 1 (RPKI Is Fully Deployed): We assume that all IP prefixes are RPKI signed and all ASes perform route-origin validation (worst-case scenario for an attacker). To be RPKI compliant, a hijacker *always* places the legitimate AS at the origin of the announced AS-path in malicious routes (i.e., Type-0 hijacks are disabled in our simulator). While it may be interesting to compare how much RPKI reduces the impact of stealthy attacks between a model where RPKI validation is enabled or disabled, we leave this as potential future work.

Assumption 2 (Either Full or No A Priori Knowledge): We consider two types of attackers:

- *Full knowledge:* In the worse-case threat scenario, a hijacker knows the routing preferences (inbound, outbound, best path selection) of all the networks in the topology. We call such an attacker an *omniscient* hijacker. We use the omniscient attacker to *estimate* the maximum potential impact of a stealthy attacker (for more, see Section V-A).
- *No knowledge:* In the best-case threat scenario, a hijacker has no information about the topology and can only access the routes that BGP route collectors publicly report. We call such an attacker a *realistic* hijacker. From these data the realistic hijacker infers the ASNs of the monitors and then the routes that monitors prefer to report for the victim's prefix. By observing how its legitimately owned IP prefixes are reported at the collectors, the realistic hijacker could identify which monitors are dangerous to report the forged routes. We use this hijacker to estimate the impact of stealthy attacks in a more realistic setting (for more on the realistic hijacker in simulations and on the real world, see Sections V-B and VII, respectively).

Assumption 3 (Single Hijack Attempt): We allow only a single attempt for the hijackers to succeed in the attack. A more naive strategy would be for the hijacker to first announce the attack and then to forge the AS-path based on what the monitors report to eventually produce a stealthy attack. While this naive strategy may work at times, the hijacker is already visible to the route collectors. Instead, the hijackers we design in Section V seek to precompute the attack and then announce it over the Internet.

Assumption 4 (BMP Not Deployed): Traditionally, monitors export their best routes to route collectors. We verified this by analyzing the routes collected by the public route collector infrastructure using BGPStream [27] for the duration of the real-world experiments in Section VII). The BGP monitoring protocol (BMP [32]) allows BMP-enabled devices to disclose all routes that they know of in their Adj-RIB-In to route collectors, instead of only the best one. As far as we are aware, RIPE-RIS route collectors currently do not support BMP and while some Routeviews route collectors support it, BMP feeds from monitors remain limited. We choose not to model BMP, as we seek to understand how reliable traditional route collectors are for observing stealthy attacks. We leave BMP-related research for future work.

Assumption 5 (Monitors Propagate Their Entire Local RIB to Route Collectors): We assume the best-case scenario for the public route collector infrastructure, i.e., that monitors provide full BGP feeds to route collectors and not partial feeds for only some of the routes that they know. While past work [33] has focused on the incompleteness of certain route collector feeders (i.e., the monitors) to properly report their best routes per prefix, measuring the attack space that this incompleteness enables is not the purpose of this work. For consistency, we assume that all monitors behave in the same way and report their best routes to route collectors (best-case scenario for the route collector infrastructure). For readers interested in learning more about how incomplete the routes that feeders provide to route collectors are, we direct them to the first three columns of Table 2. As a reference, the routing table size (i.e., the local RIB size) was around 814 K in January 2020, 860 K in January 2021, and 906 K in January 2021 [34].

IV. THE SIMULATION APPROACH

To simulate how stealthy hijack attacks propagate through the Internet, we first need to model BGP to measure the reliability of public route collectors. Section IV-A explains the modeling approach that we follow in this work, while Section IV-B explains the modeling limitations.

A. MODELING BGP

In this section, we explain how we model BGP. This involves modeling the control-plane components of Fig. 1, which we explain in the following paragraphs.

1) MODELING INBOUND POLICIES

Inbound filtering policies refer to all BGP filtering operations that a router applies to routes received from its neighbors. Routes removed by this filter are no longer considered and do not further propagate on the Internet. Examples of some common filtering operations in the wild involve route origin validation (using RPKI) and filtering of prefixes announced by customer networks. In our simulator, we assume that all ASes implement RPKI validation which blocks Type-0 attacks completely. Furthermore, we presume that providers filter the prefixes announced by their stub customers ASes, defined as the ASes at the edge without any customer. We do not consider more complex inbound filters as we seek to measure the reliability of the public infrastructure as a standalone protection mechanism. How reliable the infrastructure would have been given more advanced filtering methods could be the subject of potential future work.

2) MODELING BGP BEST PATH SELECTION

To find the *best* route over multiple routes towards the same prefix, commercial routers' BGP selection process consists of multiple decision steps [35], [36] commonly summarized as: (*i*) the highest local preference attribute (a metric governed by the BGP policies). In the case of a tie, then (*ii*) the shortest AS-path length, and finally, (*iii*) the rest of the tie-breaking methods. To model BGP's best path selection, we use a widely accepted approach followed by multiple previous works [16], [29], [37] that seek to simulate route propagation on the Internet. For (*i*) we model local preferences based on the Gao-Rexford [38] conditions, as they capture the commercial nature of ASes. According to these conditions, ASes prefer customer routes over peer routes, while these are preferred over provider routes. For (*ii*) we simply perform a comparison based on the AS-path length. For (*iii*) as the previously introduced realistic and omniscient hijackers aim to forge less-preferred routes than those currently reported by monitors (and therefore not leave the hijack propagation to luck), we consider modeling the remaining tie-breaking mechanisms less relevant for this work. We assign those tie-breakers to a router at random. However, for consistency, we make sure to use the same randomization seed across all the simulation experiments (Section VI). Finally, to prevent loops, routers in our simulations exclude from the BGP's best path selection routes containing their ASN in the AS-path.

3) MODELING OUTBOUND POLICIES

Outbound policies control how and which routes a router advertises to each neighbor. We model route propagation based on the Gao-Rexford [38] conditions. Therefore, routes received from customers propagate to all neighbors, whereas routes from peers or providers propagate only to customer neighbors.

4) MODELING THE RIB TABLES

We model the Adj-RIB-In and Local RIB tables as illustrated in fig. 1, i.e., before the inbound policies and between the inbound and outbound policies, respectively. For simplicity, and without functionality loss for this work, we skip modeling the Adj-RIB-Out table. Instead, each router directly exports its best paths to the outbound neighbors selected by the previously introduced outbound policies.

5) MODELING THE INTERNET TOPOLOGY AND ROUTING POLICIES

We use a well-tested BGP simulator [39] which we modify for the purpose of our study. We rely on the Internet topology inferred by CAIDA gathered from RouteViews and RIPE-RIS route collectors [22]. As of July 2021, the topology includes 72K ASes and 509K inter-domain links between those ASes, each characterized over its Gao-Rexford relations (p2p or c2p). We model each AS as a *single* BGP speaking node with a global routing table, i.e., global Adj-RIB-in and Local RIB for all its peering neighbors. Routes propagate from inbound AS peers to outbound AS peers as depicted in Fig. 1.

6) MODELING THE FIB AND THE AFFECTED ASes

For the purposes of this work, we do not focus on modeling the data plane as hijackers always announce the same-prefix as the victim. Instead, to identify which ASes have been affected by the hijacker, we directly scan the local RIB to discover if the best path towards the victim prefix has been replaced by the forged path announced by the hijacker.

B. MODELING LIMITATIONS

Although Gao-Rexford is the most widely accepted model we currently know to capture the inter-domain routing practices of networks, we note that this model is not sufficient to accurately capture realistic path propagation in the wild [40]. Part of the reason is that routing policies are more complex than what simple customer, provider, and peer-to-peer relations can capture. Past work has criticized this model and explained the reasons why it is not sufficient [40]. This is due to per-prefix routing, filtering, and tie-breaking decisions that are not visible and hard to infer from propagated BGP messages. Although the previous work found that Gao-rexford offers an appropriate granularity to model BGP route filtering, this is not the case for accurately modeling BGP route propagation as multiple tied Gao-Rexford conforming routes may be known by each AS that are impossible to accurately tie-break in simulations. The answer on what is the right granularity to model routing policies remains open by the previous work, and, as far as we are aware, this still remains an open-ended question to this day.

We quickly examine the compliance of ASes to Gao-Rexford routing policies in the wild. Using BGPstream (dataset date 8th July 2020), we first fetch the RIB dumps of the monitor ASes as reported by the RIPE-RIS and Routeviews route collectors. Then, using the closest CAIDA AS-relation dataset (dataset date 1st July 2020), we transform each AS-path to the corresponding Gao-Rexford relation path by converting each pair of ASes to their Gao-Rexford relation. Finally, we examine whether the Gao-Rexford relation path produced complies or not with the outbound policy model used in the simulator (see ''modeling outbound policies'' in Section IV-A).

Table 2 shows for each route collector of RIPE-RIS and Routeviews the amount of monitor ASes (peers) reporting their routes to the route collector, the total number of reported routes, and the compliance of the reported routes with Gao-Rexford propagation policies. We distinguish three cases for the compliance: (*i*) the reported route complies with Gao-Rexford (valid route), (*ii*) the reported route does not comply with Gao-Rexford (invalid route), or (*iii*) the reported route compliance is unknown, as it contains an AS-pair relation not available in CAIDA's AS relation dataset.

As we observe from the collected data, the majority of the routes reported by the monitors comply with Gao-Rexford, although the % of non-complying routes may significantly vary per project and per collector. For example, for the Routeviews route collectors, the median % of non-complying routes is at 3.16%, the mean at 4.59%, while the max at 27.22%. For the RIPE-RIS route collectors, the mean % of non-complying routes is at 5.07%, the median at 6.20%, while the max at 14.79%. Those % indicate that while Gao-rexford may offer the appropriate granularity to model the export practices for the majority ASes in the Internet, it is not sufficient to fully model the routing practices of every AS for every part of the topology.

Other related work [41] has tried to infer the routing policies that ASes use in the wild by AS-path poisoning announcements to force specific ASes to switch to alternative routes. While discovering and ordering the preferences of ASes to less-preferred routes could strengthen the feasible success of hijackers deliberately seeking to remain hidden from route collectors, we do not consider reverse engineering real-world BGP decisions (a direct consequence of AS-path poisoning) ethical for this work.

Despite the limitations mentioned above, we see from the results of Table 2 and from surveys such as [42] that Gao-Rexford offers a good granularity of detail to model route filtering. Due to a lack of a more widely accepted model, we use this model for our simulations. We note here that the goal of this work is not to realistically simulate stealthy attacks and obtain accurate to the real-world results (i.e., not accurately reproduce best path selection). Rather, to understand whether or not hijackers wishing to remain stealthy could deliberately produce such attacks and, if yes, measure their potential impact. In that regard, this work takes the first steps in this direction by measuring the feasible success of such stealthy hijackers in simulations (see Section VI). Furthermore, we conclude with a real-world analysis (see Section VII) illustrating that, despite some wrong inferences, a Gao-Rexford classifier *could provide* the appropriate granularity of detail for the hijacker to correctly classify the vast majority of actual dangerous monitors without introducing new misclassifications (see the zero false negatives for Gao-Rexford violations in Table 9).

V. DESIGN OF STEALTHY HIJACK ATTACKS

We now present a novel method of how hijackers could design impactful stealthy interception attacks affecting multiple ASes while *evading* detection from public BGP route collectors. We consider both an omniscient (Section V-A)

TABLE 2. Compliance of AS-paths to Gao-Rexford export policies as observed by the Routeviews and RIPE RIS route collectors. Valid, invalid, and unknown routes represent complying, non-complying, and unknown relation paths.

Route Collector	*# Peers*	*# Reported routes*	*% Valid routes (all routes)*	*% Invalid routes*	*% Unknown routes*
Route Views Route Collectors					
rib-route-views.amsix	27	9.32 M	94.05%	2.87%	3.08%
rib-route-views.chicago	12	9.36 M	92.92%	5.47%	1.60%
rib-route-views.chile	1	0.82 M	94.01%	1.83%	4.16%
rib-route-views.eqix	20	11.55 M	94.24%	4.63%	1.13%
rib-route-views.flix	8	3.91 M	95.32%	3.09%	1.59%
rib-route-views.fortaleza	3	1.92 M	85.67%	2.94%	11.39%
rib-route-views.gorex	1	0.00 M	0.26%	0.00%	99.74%
rib-route-views.isc	7	4.43 M	96.67%	2.68%	0.64%
rib-route-views.kixp	2	0.73 M	95.79%	2.01%	2.20%
rib-route-views.linx	32	18.94 M	96.88%	2.45%	0.67%
rib-route-views.napafrica	8	3.62 M	92.74%	5.95%	1.31%
rib-route-views.nwax	3	1.58 M	97.66%	1.99%	0.35%
rib-route-views.perth	2	0.92 M	92.82%	5.78%	1.40%
rib-route-views.rio	5	2.56 M	93.20%	3.43%	3.36%
rib-route-views.saopaulo	11	5.01 M	89.50%	8.42%	2.08%
rib-route-views.sfmix	6	1.69 M	96.79%	2.67%	0.53%
rib-route-views.sg	32	12.02 M	94.42%	4.15%	1.43%
rib-route-views.soxrs	1	0.17 M	94.39%	2.32%	3.28%
rib-route-views.sydney	10	3.60 M	93.76%	3.23%	3.01%
rib-route-views.telxatl	12	6.70 M	94.39%	4.54%	1.07%
rib-route-views.wide	4	1.68 M	96.66%	2.33%	1.01%
rib-route-views2	39	7.15 M	91.90%	6.47%	1.63%
rib-route-views2.saopaulo	19	10.16 M	91.14%	2.28%	6.58%
rib-route-views3	25	12.14 M	92.22%	7.20%	0.58%
rib-route-views4	23	12.69 M	92.71%	3.60%	3.69%
rib-route-views6	23	2.62 M	64.42%	27.22%	8.37%
RIPE RIS Route Collectors					
rib-rrc00	76	52.20 M	85.29%	11.43%	3.29%
rib-rrc01	61	26.22 M	89.77%	8.10%	2.12%
rib-rrc03	63	27.12 M	87.28%	10.20%	2.52%
rib-rrc04	10	4.68 M	94.38%	4.49%	1.13%
rib-rrc05	27	7.05 M	95.49%	2.94%	1.57%
rib-rrc06	3	2.64 M	94.04%	4.26%	1.70%
rib-rrc07	19	5.22 M	95.35%	3.26%	1.39%
rib-rrc10	21	12.97 M	94.32%	4.30%	1.38%
rib-rrc11	21	8.67 M	92.57%	5.45%	1.98%
rib-rrc12	70	28.51 M	92.05%	5.07%	2.88%
rib-rrc13	17	9.18 M	93.95%	4.72%	1.34%
rib-rrc14	14	6.93 M	94.66%	4.11%	1.23%
rib-rrc15	35	18.19 M	88.92%	6.86%	4.22%
rib-rrc16	12	3.45 M	94.90%	3.80%	1.30%
rib-rrc18	8	1.68 M	88.45%	10.85%	0.70%
rib-rrc19	24	8.86 M	82.07%	14.79%	3.14%
rib-rrc20	33	25.20 M	90.89%	6.69%	2.42%
rib-rrc21	33	22.10 M	88.20%	6.85%	4.95%
rib-rrc22	13	2.92 M	85.62%	5.08%	9.29%
rib-rrc23	15	4.49 M	91.99%	3.88%	4.13%
rib-rrc24	10	5.56 M	78.26%	3.25%	18.49%

and a realistic (Section V-B) hijacker which we later compare with a baseline (i.e., traditional) hijacker in Section VI. We use the omniscient hijacker to understand the potential upper bound impact of an all-knowing stealthy attacker and the realistic hijacker to compute the potential impact of a less knowledgeable stealthy attacker based on how the hijacker would design the attack in the real world. In both cases, we make the conservative assumptions that RPKI is fully deployed.

A. THE OMNISCIENT HIJACKER

The omniscient hijacker has perfect visibility of the inter-domain routing policies (inbound filtering, best path selection, outbound filtering) of all ASes in the topology. It uses this knowledge to forge a set of per-neighbor announcements that maximize the amount of hijack-affected networks while preventing propagation of the routes to the route collectors. However, as we mention in Section III-C, maximizing the number of hijack-affected networks (even without the stealthiness requirement) is challenging. In the following, we therefore develop a heuristic to generate *stealthy* interception attacks with the goal of propagating per-neighbor announcements as far as possible (i.e., as close to the monitors as possible), hence intercepting the traffic from a large number of networks without being reported by monitors to the route collector(s).

1) A TOY EXAMPLE

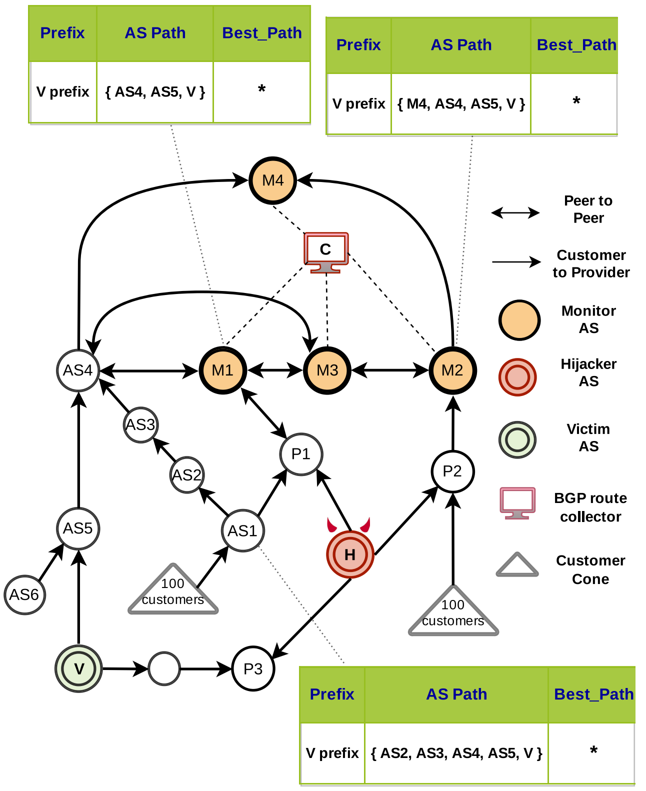
We explain the omniscient and realistic hijacker with the help of Fig. 4, which shows a toy topology with a hijacker (H), a victim (V) and four monitors (M_1 to M_4) connected to a route collector (C). The two ASes ($AS1$ and $P2$) are further connected to 100 customer ASes each. Arrows point from customers to providers or between peers (double-headed arrows). The ASes follow the Gao-Rexford [38] routing policies while monitors only export the best routes that they know to the collector. Note that for clarity reasons, we only explain the design of the attack in this small topology. However, as we show in Section VI, our heuristics are optimized and scale to sizes of the full CAIDA topology and beyond. Furthermore, although we use Gao-Rexford as a means to model routing, we note that the method that we follow in Section V-A and Section V-B to design stealthy attacks is not Gao-Rexford dependent and could be code-redesigned to work under any routing policy.

2) ATTRACTING TRAFFIC FROM AS1 AND P2

We now explain why a naive stealthy attack approach would not work well in practice by focusing on attracting traffic from the 100 nodes connected to $AS1$ and $P2$. We first note that $AS1$ and $P2$ always have a legitimate available route of length 5 towards the victim i.e., from AS2 (i.e., $\{AS2, AS3, \dots, V\}$) and M2 (i.e., $\{M2, M4, \dots, V\}$), respectively. Suppose the hijacker naively adds *all* the monitor ASNs (plus the victim V) in the malicious route to forge an attack not visible to the collector. The hijacker would need a Type-5 attack (i.e., $\{H, M1, \dots, M4, V\}$) which would result in a less-preferred route of length 7 propagating at $AS1$ (i.e., $\{P1, H, M1, \dots, M4, V\}$). This route is two hops longer than the legitimate route $AS1$ receives from $AS2$. While the attack is indeed stealthy, it would not attract any traffic from the 100 customers of $AS1$, as $AS1$ still prefers the shortest route. To make the forged route more appealing to AS1, the hijacker could instead reduce the Type of the attack by removing some monitor ASNs from the path. Although this naive strategy works at times, removing the wrong monitor's ASN would risk the forged route becoming visible to the collector. Therefore, smarter attack strategies are required to enable impactful hijacks while hiding from all the monitors, especially at the scale of today's Internet.

3) THE OMNISCIENT HIJACK STRATEGY

The design of the omniscient hijack strategy (and the realistic, see Section V-B) consists of three parts: First, (*i*) classifying the monitors between *safe* and *dangerous*. *Safe* monitors will naturally not report the attack to route collectors, whereas *dangerous* monitors would report the attack (for the reasons why monitors may not report the attack, see Section II). Second, (*ii*) designing forged neighbor-specific AS-paths that prevent dangerous monitors from either observing or reporting the attack. Third, (*iii*) announcing a stealthy interception by identifying which hijacker neighbors would maintain a path to the victim.



Prefix	AS Path	Best_Path
V prefix	{ AS4, AS5, V }	*

Prefix	AS Path	Best_Path
V prefix	{ M4, AS4, AS5, V }	*

Prefix	AS Path	Best_Path
V prefix	{ AS2, AS3, AS4, AS5, V }	*

FIGURE 4. A topology with a hijacker (H), a victim (V), and four monitors (M1-M4) peering with a route collector (C). The arrows point customers to providers or indicate peer-to-peer connections (double-headed).

4) PART 1: IDENTIFYING THE DANGEROUS MONITORS

As Fig. 2 shows, it is often the case that only *some* of the monitors would observe and report a hijack for a given prefix to the route collectors. Identifying and reacting to only these dangerous monitors is the key to achieving an impactful stealthy routing attack. For this purpose, the omniscient hijacker queries a so-called BGP Oracle.

The Oracle is an alias for a heuristic we designed that reverse engineers BGP. Given a BGP node as input (called root) an a prefix, the Oracle reverse engineers the BGP preferences and filtering policies of the root to discover which ASes in the topology could provide a *more preferred* (forged) route that will propagate and replace the best route known by the root for the given prefix in its Local-RIB. The hijacker can query the Oracle, providing the victim's prefix and a monitor as root. The Oracle will reverse engineer BGP route propagation to answer for each hijacker neighbor which hijack Types are safe or potentially dangerous to be reported by the monitor.

For example, Fig. 5 shows how the Oracle reverse engineers BGP from the root (monitor M1) to answer if this monitor is dangerous. The Oracle starts from M1 as seen in step (a). Then, based on M1's routing and filtering policies,

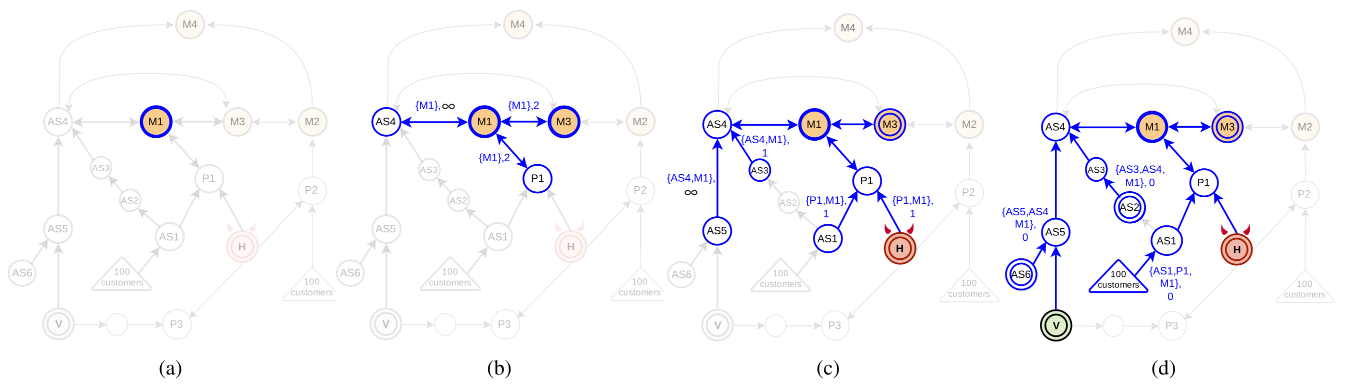


FIGURE 5. The Oracle reverse engineers BGP in four steps to discover if M1 is a dangerous monitor for the hijacker H. Low opacity nodes and links have not yet been discovered while double-circled nodes indicate terminal nodes. Edge labels show the followed path and the collected restrictions required to reverse engineer BGP.

the Oracle reverse engineers BGP identifying which of M1's neighbors could provide a more preferred forged route than the best route that M1 currently knows for the victim's prefix. To reverse engineer BGP, at least two variables need to propagate to each newly discovered neighbor, as shown in the edge labeling of step (b). The first variable is the AS path that Oracle followed from the root. It is used to break AS-path loops, as normally such routes do not exist in BGP. The second variable is an integer restricting how much further the reverse discovery is allowed to propagate from the root. It captures the first and second best path selection criteria of BGP, i.e., the local-preferences and the shortest AS-path preferences of all nodes along the path towards the route (i.e., first variable).

For example, in step (b), the Oracle discovers the ASes P1, M3, and AS4 from M1 with a restriction of 2, 2, and ∞, respectively. A restriction of 2 by M1 indicates that due to the shortest path policies of the ASes along the path to the root (in this case only M1) only Type-2 hijacks or below could propagate from P1 and M3 and corrupt the Local RIB of the root. On the other hand, a restriction of ∞ to AS4 means that any attack Type could propagate from AS4 and corrupt the local RIB of the root. In this case, this is because M1 has no other alternative paths to the victim except via AS4, meaning that any (forged) route propagating from AS4 would automatically be accepted by M1 and reported to the route collector. Other cases which could result in ∞ restrictions involve local preference policies which could cause any hijack Type to propagate to the route collector. For example, if M2 was the root, P2 would have been discovered with a ∞ restriction. This is because M2 always prefers routes received from customers rather than the best route M2 currently knows for V, over its provider M4.

Once all the root neighbors have been discovered, the reverse discovery continues in step (c), where the Oracle discovers AS3 & AS5 from AS4, AS1 & ASH from P1, and no new ASes from M3. Due to M3's routing policies, no route received by M3 can propagate to the root M1. Therefore, M1 is marked as a terminal node from which the Oracle cannot discover any more attacks. For AS5, the Oracle discovers it with a ∞ restriction for the same reason as to how AS4 was discovered from M1. Meanwhile, AS3 is discovered with a restriction of 1 as AS4 will only accept potentially Type-1 and below hijacks to propagate to M1. For AS1 and H, the Oracle discovers both with a restriction of 1. While P1 would accept any attack Type from AS1 and H, M1 would only accept Type-1 or below hijacks from AS1 and H due to M1's shorter AS-path preferences.

The reverse BGP discovery terminates once no new nodes can be discovered, illustrated in step (d). A restriction of zero indicates a terminal node, i.e., a node from which no further attack Types can be discovered as lower than Type-0 attacks do not feasible exist. If the Oracle discovered the hijacker H (in this case, it did via P1 with a Type-1 restriction – see step (c)), it informs the hijacker that exporting any attack of Type-1 or below to P1 would result in M1 reporting the hijack to the route collector.[7] If the Oracle did not discover H, then this means that the root (here M1) is always a safe monitor where no attack from the hijacker can feasibly propagate to the route collector C.

The hijacker repeats this procedure querying the Oracle for every monitor in the topology. Then, it documents the Oracle's reply, i.e., the safe and the dangerous monitors per hijacker neighbor, and the Type restrictions (summarized in Table 3). The methodology then proceeds to the next part; the design of a stealthy attack per hijacker neighbor.

5) PART 2: DESIGNING STEALTHY HIJACKS

Having documented the dangerous and safe monitors per neighbor, the omniscient hijacker now forges stealthy hijacks to route collectors.

For hijacker neighbors in which Table 3 does not contain any dangerous monitors, such as P3, H can safely announce any forged route without risking this route propagating to the route collector. For example, H announces the shortest

[7]While the M1 would report Type-0 attacks to the route collector, Type-1 attacks may still not, depending on M1 tie-breakers.

TABLE 3. Restrictions reported by the BGP Oracle for each monitor per hijacker neighbor. ✗ indicates a neighbor not discovered when reversing engineering BGP from the corresponding monitor.

Hijacker Neighbors	*Propagated restrictions*			
	M1	**M2**	**M3**	**M4**
P1	1	✗	✗	✗
P2	✗	∞	0	0
P3	✗	✗	✗	✗

available and commonly most impactful attack, i.e., the Type-1 RPKI valid route $\{H, V\}$.

For hijacker neighbors in which the table contains dangerous monitors, like P1 and P2, H must forge a higher Type route than the specified monitor restrictions or risk corrupting the local RIB of the monitors, and therefore, the attack becoming visible to the route collector. For example, to P1, H can safely announce the Type-2 RPKI valid route $\{H, V, V\}$.[8] This attack satisfies the restrictions set in the table for M1, and therefore, it will not propagate to the route collector due to the shortest AS-path preferences of M1.

For hijacker neighbors in which Table 3 contains ∞ restriction(s), such as P2, any route announced by H will propagate to the route collector. This is due to M2 routing preferences, which cause M2 to prefer more customer routes over provider routes. As no Type attacks are safe, H has to AS-path poison the forged route with the ASN of M2. H safely announces the Type-2 RPKI valid route $\{H, M2, V\}$. This forged route satisfies the restrictions set by M3 and M4 while, at the same time, it eliminates the restrictions of the monitor M2 in the table. Due to BGP's loop prevention, M2 now drops the route instead of reporting it to the route collector.

At this point, the omniscient hijacker has configured a stealthy blackhole attack for each neighbor. While this attack is stealthy to control-plane monitors, it is easily noticeable in the data-plane as it disrupts traffic communication of the victim to affected ASes. The final part explains how to convert the stealthy blackhole to a stealthy interception.

6) PART 3: DESIGNING INTERCEPTION ATTACKS

So far the omniscient hijacker has forged a discrete bogus route to announce to each neighbor that will not propagate to the route collector. By announcing these forged routes, H will blackhole the traffic destined to the victim from the hijack affected ASes. To convert the stealthy blackhole attack to a stealthy interception, the hijacker needs to: (*i*) identify which neighbors are going to maintain a route towards the victim, and (*ii*) select among them one to which to *not* export the attack. Identifying which neighbors will maintain a path is challenging, as the hijacker needs knowledge about how the forged routes will propagate before the hijack is actually announced. Recall that failure to maintain a route towards the victim is not an option as the hijacker has a single attempt to create the attack.

To solve the problem at hand, the hijacker makes the following observation: a legitimate path to the victim is guaranteed to be maintained if and only if *none* of the ASes in that path are affected by the hijack, i.e., if none of the ASes install the bogus route(s) in their Local RIB.

To identify if any AS is going to install the bogus route(s), the Omni hijacker queries the BGP Oracle. By providing each ASN on the legitimate path as input to the Oracle (i.e., root), the hijacker constructs a restriction matrix similar to Table 3 (ASes on the legitimate path now replace the monitors). After creating a new restriction matrix per candidate interception path, the matrices are compared against the stealthy attack Types designed part 2. If for any candidate interception path the Oracle does not discover the hijacker (i.e., restriction matrix empty), or (*ii*) if discovered but with a lower restriction than the planned attack Types (designed in part2), then the hijacker can safely conclude that the candidate interception path is going to be safely maintained after the attack.

In our example, the hijacker has two candidate routes towards the victim, one via P3 and one via P2. Using the Oracle, the hijacker constructs two empty restriction matrices, one for each candidate route. Because the P3 matrix is empty, the hijacker identifies that P3 will maintain its route no matter what bogus route is announced to P1 and P2. Similarly, because the P2 matrix is empty, the hijacker identifies that P2 will maintain its route no matter what bogus route is announced to P1 and P3. As P2 offers the customer cone that the hijacker wants to affect, it creates a stealthy interception attack by announcing the forged routes to all neighbors with the exception of $P3$ which provides the return path.

7) CLARIFICATIONS

We note here that unlike previous works that aim to accurately infer how routes propagate, such as [40], [43], [44], and [45] the purpose of the Oracle heuristic, as illustrated in Fig. 5, is not to accurately predict how routes will propagate, rather to accurately answer if (and under which Type conditions) a forged route will propagate to the route collector. To this regard, all the per-neighbor attacks the omniscient hijacker announces have been designed with the specific property of remaining hidden to route collectors. While how the attack propagates may change based on to which neighbors the hijacker decides to export the attack (2nd part and 3rd part of the design), the stealthy property (engineered in the 1st part) remains intact as each neighbor specific announcement has been designed with that stealthiness property in mind. As our results in Section VI show, an omniscient hijacker who has complete knowledge of the import and export policies of other ASes is capable of *always* designing a stealthy hijack to route collectors, indicating the successful predictive capabilities of the Oracle.

[8] To create the desirable attack Type, we simply prepend the forged route with the victim's AS. A hijacker aiming to forge a more credible route could instead AS-path poison the forged route with the victim's provider.

B. THE REALISTIC HIJACKER

Unlike the omniscient hijacker, the realistic hijacker has *limited* knowledge about the Internet topology, obtained from the BGP feeds that route collectors publicly report. From these feeds, the realistic hijacker can identify the monitors, record the routes monitors disclose to route collectors, and finally design an attack that is potentially not reported by monitors to the public route collectors.

Similarly to the omniscient hijacker, the design of the stealthy attack for the realistic hijacker consists of three parts: (*i*) the classification of the monitors between safe and dangerous, (*ii*) the design of neighbor-specific forged routes, and (*iii*) the identification of neighbors that will maintain a route to the victim.

1) PART 1: IDENTIFYING THE DANGEROUS MONITORS

Unlike the omniscient hijacker, the realistic hijacker cannot utilize the BGP Oracle to accurately classify the monitors. Instead, it *estimates* the dangerous and safe monitors based on the proximity of the monitor to the victim compared to the proximity of the monitor to the hijacker ($Prox_{M,P}$). The proximity captures the selection of the shortest AS-path towards a destination prefix. While, of course, ASes may select longer AS-paths towards a destination as more preferred (e.g., due to local preferences), as our simulations and real-world results show, sometimes proximity is sufficient to identify the dangerous monitors.

To compute the proximities, first, the hijacker independently announces one distinct IP prefix that it owns to each neighbor. Afterwards, the hijacker queries the public route collectors to figure out how these prefixes propagated from each neighbor to each monitor and extracts the AS-path lengths. These path lengths represent the distance at which the hijacker expects each monitor to observe the future attack once it is announced. Finally, the hijacker compares those distances with the distance at which the monitors currently observe the victim's prefix. It calculates the proximity *difference* between the distances (shown in Equation (1)) and builds the corresponding proximity matrix (shown in Table 4).

$$\begin{aligned} Prox_{M,P} = &\ len(victim\_best\_path_M) \\ &- len(hijacker\_best\_path_{M,P}) \qquad (1) \end{aligned}$$

where M represents a monitor and P represents a neighbor of the hijacker. Similarly to Table 3, a negative proximity indicates safe monitors that the hijacker expects to not observe the attack. A zero or positive proximity indicates dangerous monitors that the hijacker must avoid e.g., by designing attack Types higher than those designated in Table 4. If due to its location a monitor does not observe either the hijacker's or the victim's prefix, we say that the corresponding distance to the monitor is ∞, and thus, compute a $Prox_{M,P}$ of either $-\infty$ or ∞ (respectively, see Equation (1)).

For example, to compute the proximities to the monitors, M1-M4, the hijacker announces three distinct IP prefixes that

TABLE 4. The proximity matrix of each monitor per hijacker neighbor.

Announced	*Monitor Proximity*			
Prefix To	**M1**	**M2**	**M3**	**M4**
P1	1	$-\infty$	$-\infty$	$-\infty$
P2	$-\infty$	2	0	0
P3	$-\infty$	$-\infty$	$-\infty$	$-\infty$

it owns, one to each neighbor. For the prefix announced to $P1$, $M1$ reports the route $\{M1, P1, H\}$ to the route collector. $M1$ also reports the route $\{M1, AS4, AS5, V\}$ for the victim. The proximity difference of the monitor to the hijacker and the victim is $Prox_{M1,P1} = 4 - 3 = 1$. The hijacker learns that to avoid the monitor from $P1$, it needs to announce a Type-2 or longer attack. Similarly, the prefixes announced to $P2$ and $P3$ are not reported by M1 to the route collector. Both proximity differences are $Prox_{M1,P2} = Prox_{M1,P3} = 4 - \infty = -\infty$. The hijacker concludes that $M1$ is safe for hijacks announced to $P2$ and $P3$. Similar procedure for the rest of Table 4.

2) PART 2: DESIGNING STEALTHY HIJACKS

Similarly to how the omniscient hijacker forged AS-paths based on the restriction matrix (Table 3), the realistic hijacker forges AS-paths based on the proximity matrix (Table 4). However, as monitors may actually select longer AS-paths toward a destination (due to local preferences), the realistic hijacker may misclassify some dangerous monitors. For example, the entry $[P2, M2]$ in Table 3 contains the correct restriction under which the routes announced by $P2$ propagate to the route collector. A Type restriction of ∞ means that any attack propagates by $M2$ to the route collector. Meanwhile, the $[P2, M2]$ entry in Table 3 contains only an attack Type of value 2 misclassifying Type-3 attacks and above as safe.

To reduce the probability that the monitor reports the attack due to misclassifications such as the ones above, the realistic hijacker combines AS-path poisoning with AS-path prepending. For each neighbor, the hijacker considers attacks of increasing Type, starting with Type-1. Given an attack of Type-N, the hijacker recomputes the safe and dangerous monitors for that neighbor based on the fact that the forged route is now N hops longer. However, note that longer routes of N hops also have space for N additional AS-path poisons. The hijacker uses this observation to AS-path poison the forged route with the ASNs of the N highest proximity monitors (based on Table 4). If any monitor remains still dangerous at the end of this step, the hijacker increases the attack Type to $N+1$ and repeats the above procedure. The calculation of the forged route ends when no monitors remain dangerous, either because the proximity restriction of the monitor is satisfied (in Table 4) or because the restriction is eliminated due to poisoning of the AS-path.

For example, to compute the forged route announced to $P2$, H begins by planning a Type-1 attack to $P2$, i.e., $\{H, V\}$. This attack satisfies the proximity restrictions of monitors $M1$, $M3$, and $M4$, but $M2$ which continues to remain dangerous (see [$P2, M2$] entry in Table 4). As Type-1 attacks that are RPKI valid have no space available for the hijacker to AS-path poison, H increases the planned attack to a Type-2 attack. Based on Table 4, $M2$ continues to remain dangerous, as the hijacker believes that Type-3 or greater attacks are required to block the propagation of the hijack to the route collector. However, H now has one space available to poison the forged route. As $M2$ is the only remaining dangerous monitor for $P2$, H inserts $M2$ into the forged path (i.e., $\{H, M2, V\}$), thus eliminating the restriction $M2$ from the table. This prevents the hijack from propagating to the route collector, as $M2$ drops the route due to the BGP loop avoidance algorithm.

In the above case, the attack is successful as H achieves to attract traffic from $P2$ while evading detection. However, this may not always be the case for all attack scenarios. In fact, since the realistic hijacker does not know the routing policies of other networks, there may always be misclassifications. At this point, the realistic hijacker proceeds to the next part, the design of the stealthy interception attack.

3) PART 3: DESIGNING INTERCEPTION ATTACK

Past work [29], [46] has researched the conditions that the hijacker must meet to produce interception attacks with a high probability of success. Specifically, if (*i*) networks propagate AS-paths based on economic incentives and (*ii*) the topology contains no customer-to-provider cycles, then an interception path is always maintained if: (*i*) the interception path is from a customer, (*ii*) the interception path is from a peer-to-peer related neighbor while the hijack announcement is either to another peer-to-peer related neighbor or to a customer, or (*iii*) the interception path is from a provider while the hijack announcement is to a customer.

Using the above properties, the realistic hijacker identifies one neighbor with a valid path to the victim that will be maintained. The attacker then announces the forged routes from the previous step to all *other* selected neighbors. If no neighbor fulfills the properties, the attacker instead selects the neighbor offering the shortest path to the victim as the one that is most likely to maintain its path.

For example, the hijacker H needs to select a neighbor among $P1$, $P2$, and $P3$ whose return path is expected to be maintained after the attack. As all the neighbors of the hijacker are providers, none satisfies the conditions presented above. Instead, the hijacker selects $P3$ as it is the neighbor offering the shortest return path back to the victim, and thus the one most likely to be maintained after the attack. Finally, H announces the designed attack to all neighbors except the selected one, $P3$.

VI. SIMULATION FINDINGS

We now assess how capable sophisticated hijackers (i.e., the above omniscient and realistic hijackers) are to hide from public route collectors. Using the simulated environment of Section IV-A, we aim to answer three main questions:

- How effective are today's public route collectors in the face of sophisticated hijackers?
- How does the flattening of the Internet affect the capabilities of sophisticated hijackers?
- What are the benefits of increasing the number of monitors?

First, we describe the methodology of our evaluation with more details about the simulated environment and topology.

A. COMPARED ATTACK STRATEGIES AND MEASURED METRICS

We compare the feasible success of the omniscient and the realistic hijacker against the feasible success of a simple baseline hijacker that announces the same Type hijack to all its neighbors. The baseline hijacker provides a reference of what a traditional hijacker can achieve that does not seek to avoid route collectors. The omniscient hijacker provides an upper-bound reference to what is possible by an all-knowledgeable hijacker. While such a hijacker may not exist in the real Internet, it provides useful information on how increasing the number of monitors would benefit route collectors against stealthy attacks. Finally, the realistic hijacker provides a reference of what would be possible by a real-world hijacker.

In each simulation, the victim first announces its prefix to all its neighbors. Then, after the victim's prefix propagates, the hijacker announces the attack according to its strategy (baseline, realistic, omniscient). To compute the feasible success of the hijack, we measure the following pair of figures: (*i*) the number of monitors that report the attack to the route collectors. We say that the attack is stealthy if it is not reported by any monitor (i.e., it affects *none* of the monitors). (*ii*) the number of hijack-affected ASes that prefer the forged route of the hijacker over the valid route of the victim. As we are interested in this number only if the attack succeeds, we measure this number only if the attack is a stealthy interception. In any other hijack outcome, we mark the attack as failed with zero stealthily affected ASes.

B. SELECTION OF HIJACKERS AND VICTIMS

In each simulation, a random victim and hijacker AS is selected. We make sure that the victim is: (*i*) among the tier-2 and tier-3 networks (originating the vast majority of today's traffic), (*ii*) well connected with at least two providers/peers, and (*iii*) the victim's prefix propagates to most of the Internet (i.e., $> 95\%$ of ASes) to remove any outliers that may exist in the CAIDA topology. Furthermore, we assign the hijacker so that: (*i*) it is not among the monitor ASes – as we assume monitor reports can be trusted, (*ii*) it is connected with at least two neighbors so that intercepting is feasible, (*iii*) it is not among tier-3 ASes (i.e., ASes with no customers) – as we assume their announcements are filtered by their providers), and (*iv*) it is not a neighbor of the victim – as we assume there is a more trust-based relationship.

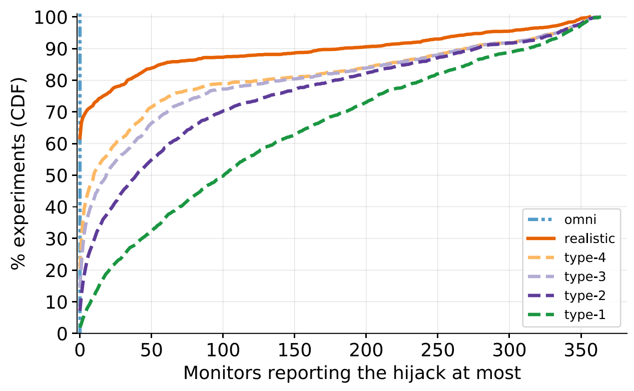


FIGURE 6. **Baseline (Type-1 to Type-4), realistic, omniscient hijackers: Number of monitors reporting the attack in the traditional CAIDA topology. Lines closer to X-axis means more visible attacks.**

In each pair of figures presented in this section, we run 2000 simulations for each line. The above victim restrictions, (*i*) - (*ii*) , limit the possible victims to 44.5 K ASes from the 72 K that exist in the topology. Similarly, the above hijacker restrictions, (*i*) - (*iii*) , limit the possible hijackers to 5.5 K ASes from the 72 K. In total, the 2000 simulations consist of 1958 unique victims and 1670 unique hijackers. We make sure that this selection of victim-hijacker pairs is identical and valid (based on the above restrictions) across all the IXP and monitor topologies. To remove noise caused by tie-breakers that may affect propagation of prefixes, we use the same randomization seed to initialize the same BGP tie-breakers across all topologies. A question that arises here is whether 2000 simulations are sufficient or not to measure the feasible success of realistic and omniscient hijackers. We also experimented with more simulations (up to 10 K) but discovered similar findings (for more, see Appendix B).

The rest of this section consists of the following subsections: (*i*) findings for the traditional CAIDA topology (see section VI-C), (*ii*) findings in flatter topologies with more IXP links (see section VI-D), (*iii*) findings in topologies containing more monitors (see section VI-E) and (*iv*) research on the hijack properties that enable stealthy attacks (see section VI-F).

C. FINDINGS—TRADITIONAL CAIDA TOPOLOGY

We now focus on the traditional (serial-2) Internet topology inferred by CAIDA [22] (July 1st 2021), which includes 72 K ASes and 509 K interdomain links. From one day worth of BGP UPDATES (1st July 2021), we have identified 367 peer ASes that actively report their routes to the public route collector infrastructure. We mark those 367 ASes as the monitors that the hijacker needs to avoid in the simulated topology.

1) VISIBILITY—BASELINE HIJACKERS

We first seek to understand how effective realistic and omniscient hijackers are in avoiding the public infrastructure. For this reason, we use the baseline hijacker as a reference, i.e., to compare realistic and omniscient attackers with what is feasible from the perspective of a naive hijacker that does not seek to avoid the monitors.

Fig. 6 shows for a certain percentage of omniscient, realistic, and baseline simulations (Y-axis), the number of monitors that *at most* report the hijacker to the route collectors (X-axis). As we observe by looking at the lines along the Y-axis, out of the 2000 simulations, 2%, 7%, 15%, and 21% of the Type-1, Type-2, Type-3, and Type-4 baseline hijackers (respectively) were not reported by any monitor. This suggests that, based on the relative position of the hijacker, the victim, and the monitors in the topology, locations exist where stealthy hijacks may be randomly feasible even by hijackers that do not seek to avoid the monitors.

2) IMPACT—BASELINE HIJACKERS

Fig. 7 complements the above picture by showing for a certain percentage of simulations (Y-axis) the number of ASes that omniscient, realistic, and baseline hijackers were *at most* able to affect while remaining hidden to the route collectors (X-axis). As we observe by looking at the lines along the Y-axis, out of the 2000 simulations, less than 1% of Type-1 baseline hijackers were able to stealthily maintain a route to the victim without the attack being observed by route collectors. This percentage increases to 4%, 9%, and 13% for baseline hijackers of Type-2, Type-3, and Type-4 (respectively). Although sometimes baseline hijackers may be lucky and succeed in stealthy interceptions, usually they can intercept only a few ASes. Looking at the tail of the CDF for Type-1 to Type-4 hijacks, we notice that such hijackers could not stealthily intercept more than 1158 ASes in the traditional topology. Past work by Sermpezis et al. [16] showed in simulations that hijacks that affect more than 2% of the Internet (i.e., 1440 ASes in our simulations) are ***always*** visible by public route collectors. Indeed, our simulations verify this finding for *baseline hijackers* in the traditional 2021 CAIDA topology.

3) VISIBILITY—REALISTIC AND OMNISCIENT HIJACKERS

Having understood what is feasible from the perspective of baseline hijackers based on visibility and impact, we now focus on what is feasible from the perspective of realistic and omniscient hijackers. Compared to baseline hijackers which are commonly visible, realistic and omniscient hijackers that dynamically adapt the attack they announce to purposely export less-preferred attacks to specific neighbors are better at hiding from route collectors. As the lines along the Y-axis in Fig. 6 show, realistic hijackers were completely invisible for 62% of the simulations, a stealthiness improvement by almost *a factor of three* compared to traditional Type-4 hijacks. Still, though, as Fig. 6 shows and as our real-world experiments in Section VII later show, realistic hijackers can still make classification mistakes that reveal them to route collectors. Recall that realistic hijackers estimate and adapt the attacks they announce based on the monitors they consider

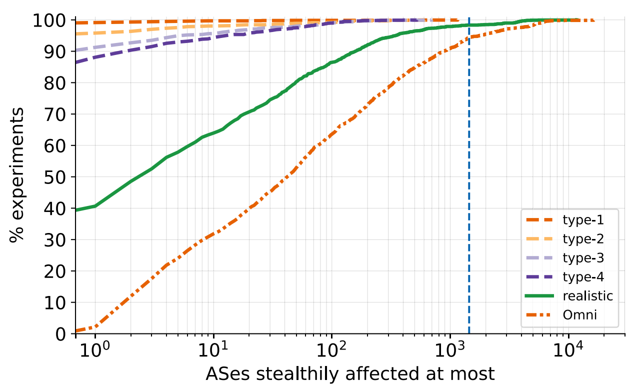


FIGURE 7. Baseline, realistic, omniscient: Number of ASes affected by stealthy interceptions in the traditional CAIDA topology. Lines closer to X-axis mean more ASes affected. The dashed vertical line illustrates 2% of the topology (i.e., 1440 ASes).

dangerous. Since such hijackers are unaware of the exact local policies that other networks use to propagate routes, incorrect inferences about the exact dangerous and safe monitors are possible. This was the case for the 38% failed simulations that we observe in Fig. 6.

On the contrary, omniscient hijackers, who have complete knowledge of the import and export policies used by other ASes to install and advertise routes, were able to ***always*** generate a stealthy hijack for all 2000 simulations (see the omniscient vertical line at $X = 0$ in Fig. 6). This finding holds also true for **all** the simulation experiments, both in the traditional topology and in topologies with more monitors or more p2p links over IXPs (see Section VI-E and Section VI-D, respectively). [9]

4) IMPACT—REALISTIC AND OMNISCIENT HIJACKERS

We now focus on the number of ASes that realistic and omniscient hijackers can stealthily affect without being observed by public route collectors. As discussed, Type-4 baseline hijackers that do not deliberately try to avoid route collectors were able to create stealthy interceptions in 13% of the simulations. At the same time, such hijackers were not able to affect more than 1158 ASes (i.e., $< 2\%$ topology). On the contrary, realistic and omniscient hijackers in Fig. 7 were able to create stealthy interceptions in almost 62% and 100% of the simulations, respectively (a change of less than 1% compared to Fig. 6). At the same time, by announcing neighbor-specific forged routes, realistic and omniscient hijackers were able to stealthily intercept more than 2% of the topology in 1.65% and 5.65% of the simulations, respectively (see the tail of the CDF). At worse, up to a maximum of 16.2%, i.e., 11.7 K ASes, and 23.5%, i.e., 17 K ASes, were stealthily affected by the two hijackers, an increase by a factor of 8.1× and 11.7× compared to baseline hijackers (respectively).

[9]Except for a single simulation where the omniscient hijacker was visible by a single monitor (for more on the number of monitors reporting the hijacker, see Table 10.

5) ANALYSIS OF EXPORTED FORGED ROUTES

In total, within the 2000 simulations, 84 K (61 K) neighbor-specific hijacks were announced by omniscient (realistic) hijackers, with 18 K (18 K) being to customers, 63 K (40.5 K) to p2p, 3 K (2.5 K) to providers (respectively).[10] Fig. 8 shows for certain percentages of forged announcements, the attack Types that were at most exported to each of hijacker's neighbors (grouped by customer, p2p, and provider relation). Compared to Type-1 to Type-4 hijackers, which *always* announce the designated attack Type, 99.9% (99.9%) of the omniscient (realistic) routes announced to customers were at most Type-1 attacks, reaching a maximum of a Type-5 (Type-4) exported attack. Similarly, 77% (84%) and 96% (98%) of the omniscient (realistic) routes to p2p neighbors were at most Type-1 and Type-4 hijacks (respectively), reaching a maximum of a Type-11 (Type-8) attack. Finally, 18% (1.5%) and 63% (47%) of the omniscient (realistic) routes to transit providers were at most Type-1 and Type-4 hijacks (respectively), reaching a maximum of a Type-15 (Type-8) exported attack. Practically, those large percentages of Type-1 routes announced to customers and p2p neighbors indicate that only ***a few*** of these neighbors are responsible for hijacks propagating to route collectors.[11]

To better understand who is responsible, Table 5 provides more details on the neighbors that caused baseline, realistic, and omniscient hijacks to become visible to the route collectors. To create this table, for each simulation, we extract the routes reported by monitors. Then mark the group(s) of hijacker neighbors (i.e., customers, peers, providers) responsible for the hijack propagating to route collectors. Each group (columns 3-5 in Table 5) is counted only once per simulation to compute the percentage responsibility over the visible simulations.

For example, baseline Type-1 hijacks were visible in 1963 of the 2000 simulations. In 0.3% of those 1963 simulations, the neighbor responsible for the hijack becoming visible was a customer. While for 47% and 99% of the simulations, the neighbor responsible was a p2p or a provider (respectively). As we observe, the responsibility from Type-1 to Type-4 attacks decreases for customer and p2p neighbors, indicating that the length of forged announcements matters to such neighbors. In contrast, the responsibility of the provider neighbors remains virtually the same, indicating that path lengths do not matter for such neighbors. Indeed, realistic hijackers that decide which monitors to avoid based on proximity (i.e., AS-path lengths) are able to create stealthy attacks for the vast majority of p2p announcements, with only 3% of the simulations remaining still dangerous due to such forged routes. Unlike realistic hijackers, omniscient hijackers that know the routing policies of other ASes were able to completely *eliminate* responsible neighbors, including hijacks announced to *provider* neighbors.

[10]Realistic hijackers may decide to not announce an attack to neighbors with too many dangerous monitors.

[11]Recall that Type-1 are the minimum allowed attacks in simulations (due to RPKI) announced only when no monitors are considered dangerous.

TABLE 5. Neighbors responsible for the hijack propagating to route collectors (grouped by Gao-Rexford relation).

Hijacker Configuration	*# Visible Simulations*	*Hijacker Neighbor Responsible (grouped by relation)*		
		Customers	***P2Ps***	***Providers***
Type-1	1963	0.3%	47%	99%
Type-4	1570	0.0%	24%	99%
Realistic	764	0.0%	3%	99%
Omniscient	0	0%	0%	0%

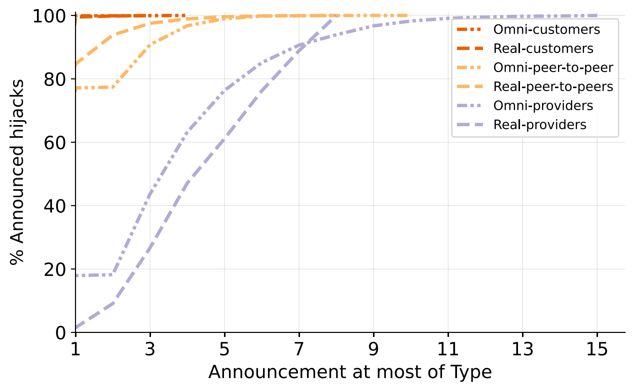


FIGURE 8. omniscient and realistic hijackers: distribution of attack Types announced to customers, peer-to-peers, and providers in the traditional CAIDA topology.

6) CONCLUSIONS

Having analyzed the capabilities of hijackers in the traditional topology of CAIDA, we reach the following conclusions:

- Globally distributed route collectors are clearly effective in observing simple hijacks. As expected, longer, and therefore, less-preferred malicious routes are stealthier to the distributed infrastructure. Although higher-Type attacks may sometimes be completely stealthy to the infrastructure (e.g., in 21% of the attacks for the Type-4 simulations), such hijackers could **not** affect while remaining stealthy more than 2% of the Internet.
- In contrast, by selectively exporting longer forged routes to specific p2p and provider neighbors, realistic hijackers were by a factor of 3× stealthier (i.e., completely stealthy in 62% of the simulations) and by a factor of 8× more impactful (i.e., up to 16.2% of the Internet affected), while exporting *shorter* than Type-4 hijacks for 95% of the announced routes. Although realistic hijackers may still make mistakes that reveal them, when visible, they are always reported by *less* monitors than baseline hijackers.
- In contrast, omniscient hijackers were completely invisible in **all** of the simulations and by a factor of 11.7× more impactful than baseline hijackers while exporting shorter than Type-4 hijacks for 91% of the announced routes.

D. FINDINGS–INCREASING NUMBER of IXP LINKS

Having evaluated the number of monitors reporting realistic and omniscient attacks and the number of ASes that such hijackers can stealthily affect in the traditional CAIDA topology, we now focus on flatter topologies studying the impact of increasing the number of peer-to-peer links.

Since the original CAIDA topology largely underestimates the number of peer-to-peer links [25], we consider topologies augmented with the CAIDA's IXP database [47], which contains 966 IXPs and their member ASes. In our simulations, we control the percentage of extra links added at these IXPs, from 0% to 100%. When considering all IXP links, the topology contains 5M links (up from the original 509K). In the absence of any information, we assume these links are peer-to-peer. Similarly to before, we consider Gao-Rexford routing policies.

1) FLATTER TOPOLOGIES ARE NOT WELL-OBSERVED BY ROUTE COLLECTORS

We now focus on how the visibility of completely stealthy hijackers changes as we increase the number of peer-to-peer interconnections over the IXPs that exist in the topology. Similarly to before, we select the same hijacker-victim pairs and repeat the baseline, realistic, and omniscient simulations in flatter Internet topologies. Fig. 9 shows for a certain percentage of simulations (Y-axis), the number of monitors that report the baseline Type-1, the baseline Type-4, and the realistic hijacker (X-axis) as we increase the number of peer-to-peer interconnections established in each IXP by 0%, 25%, 50%, 75%, and 100% of its total ASN members (respectively).[12] As the lines across the Y-axis show (for $X = 0$), increasing the number of peer-to-peer interconnections does **not** appear to affect the feasibility of hijackers succeeding a stealthy attack. The percentage of completely stealthy attacks remains virtually the same as in the original CAIDA topology (IXP0), i.e., 2% for Type-1, 21% for Type-4, and 62%for realistic hijackers. The fact that all hijacker types remain unaffected, even baselines which do not deliberately avoid the monitors, means that routes over peer-to-peer relations are not well-observed by monitors. In fact, as indicated from a previous work [25], routes over peer-to-peer links are indeed not well-observed by route collectors.

2) ALREADY VISIBLE ATTACKS ARE STEALTHIER IN A FLATTER INTERNET

We now focus on the visible hijacks of Fig. 9. Looking across the X-axis we notice that as the number of ASes peering over IXPs increases, fewer monitors report all hijacker kinds, i.e., both baseline and realistic hijackers. This change is significant for some of the simulations. For example, when looking at the 90th percentile, baseline Type-1, baseline Type-4, and realistic hijackers were visible by 314, 269, and 182 monitors (respectively) out of the 367 that exist in the original topology. These numbers decrease by up to 28% (226 monitors), 50.9% (132 monitors), and 48.3% (94 monitors) for Type-1, Type-4, and realistic hijackers (respectively), as we

[12]Omniscient hijackers were always stealthy. Therefore, for clarity reasons, we removed them from this figure.

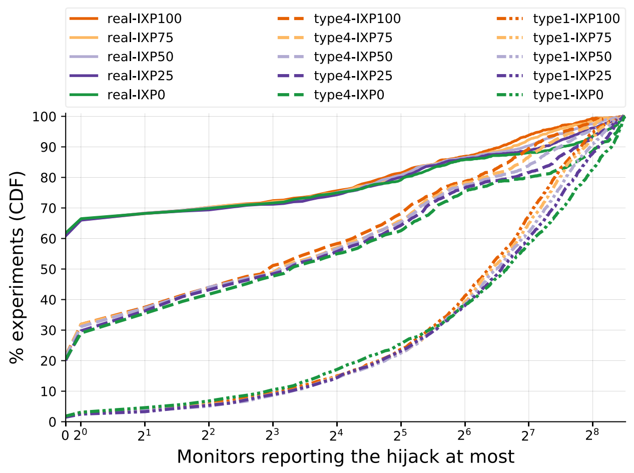


FIGURE 9. **Baseline, realistic, Omni: Number of monitors reporting the attack when adding more IXP links in the topology. Lines closer to the X-axis mean more visible attacks.**

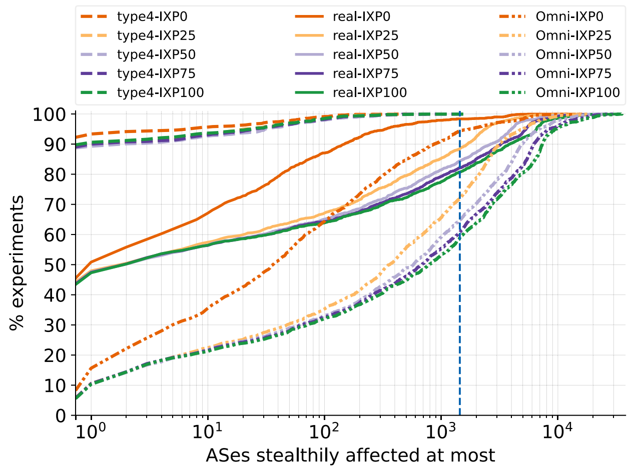


FIGURE 10. **Baseline, realistic, Omni: Number of ASes affected by stealthy interceptions as we increase the number of IXP links. Lines closer to X-axis mean more ASes affected.**

increase the number of IXP peer-to-peer links in the topology. These results of reduced visibility are a direct consequence of the above statement i.e., that peer-to-peer links are not well-observed by route collectors. For information on more percentile values, see Table 10 in the appendix.

3) STEALTHY ATTACKS ARE MORE EFFECTIVE IN A FLATTER INTERNET

Similar to Fig. 7, Fig. 10 shows for a certain percentage of simulations (Y-axis) the number of ASes that hijackers were at most able to stealthily affect as we increase the number of peer-to-peer interconnections established in each IXP by 0%, 25%, 50%, 75%, and 100% of its total ASN members (respectively). A question that remains open from the previous figure is how many ASes are using the hijacker that were not prior to the attack. To answer this, from this and all forward figures, we choose to ignore from the list of affected ASes all ASes that preferred the hijacker prior to the attack.[13] As the lines across the X-axis clearly indicate, adding more peer-to-peer links at IXPs causes stealthy interceptions to become more effective. Type-4 hijackers were originally able to affect up to 514 ASes in the traditional CAIDA topology (IXP0) without being observed by public route collectors. This increases to 915 ASes, 1460 ASes, 1540 ASes, and 1630 ASes for topologies of 25%, 50%, 75%, and 100% IXP links (respectively). Similarly, realistic and omniscient hijackers that deliberately try to avoid the monitors were the most profited from the flatter topology. Realistic hijackers were originally able to affect up to 11.7K (16.2%) ASes in the traditional topology without being observed by route collectors, and this increases to 14.9K (20.7%), 22.4K (31.0%), 25.9K (35.9%), and 32.9K (45.5%) ASes for topologies of 25%, 50%, 75%, and 100% IXP links (respectively). Similarly, omniscient hijackers affected up to 17K (23.5%) ASes in the traditional topology and this increases to 18.1K (25.1%), 24.5K (33.9%), 30K (41.7%), and 35.4K (49.0%) ASes for topologies of 25%, 50%, 75%, and 100% IXP links (respectively). While the above numbers should be received with skepticism, as they may overestimate reality, the following trend becomes clear: Stealthy hijacks are more effective the flatter the Internet topology. For a more complete list of percentile values, see Appendix C.

4) CONCLUSIONS

Having analyzed the number of monitors that report the hijacker and the number of ASes that hijackers can stealthily hijack without being observed by route collectors, we reach the following conclusions:

- ***Visibility:*** Routes over peer-to-peer relations are not well-observed by route collectors. We believe a hijacker could take advantage of the natural stealthiness that such routes provide to potentially design stealthy hijacks in the real-world.
- ***Impact:*** Stealthy attacks are more effective the flatter the Internet topology. Compared to the original CAIDA topology where baseline hijackers could not stealthily hijack more than 2% of the Internet (i.e., 1440 ASes), in full-flat topologies where every member of an IXP peers with other members, baseline hijackers increase their impact by a factor of 1.1×, realistic hijackers by a factor of 22.7× (up from the 8×), and omniscient hijackers by a factor of 24.5× (up from the 11.7×).

E. FINDINGS–INCREASING NUMBER of MONITORS

We now focus on future topologies where more ASes establish BGP peering sessions with public route collectors. We call such ASes *monitors*. We seek to answer what are the benefits of increasing the number of monitors given that limited ASes are willing to peer and disclose their routes to public route collectors.

[13]This explains why the % failed simulations slightly increase for the traditional topology in Fig. 10.

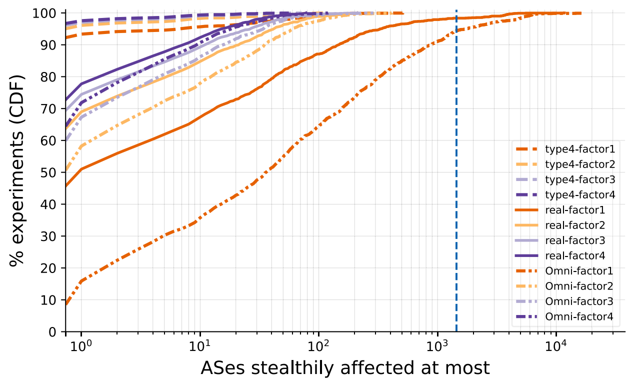


FIGURE 11. **Baseline, realistic, omniscient: Number of ASes affected by stealthy interceptions as the number of monitors increases in the traditional CAIDA topology (IXP0). Hijacker is not aware of the new monitors. Lines closer to X-axis mean more affected ASes. Factor1 refers to the original monitors.**

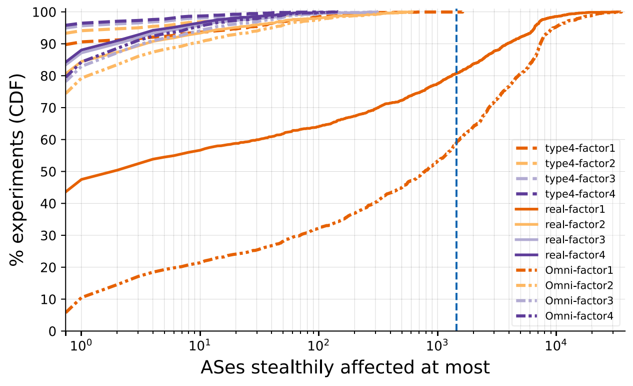


FIGURE 12. **Baseline, realistic, omniscient: Number of ASes affected by stealthy interceptions as the number of monitors increases in the full-flat topology (IXP100). Hijacker is not aware of the new monitors. Lines closer to X-axis mean more affected ASes. Factor1 refers to the original monitors.**

To select the additional monitor ASes, we look at CAIDA's AS-Rank [48] distribution of today's monitors among Tier-1, Tier-2, and Tier-3 networks. Then, for each existing monitor, we select a new monitor so that the new distribution resembles the old one as closely as possible. In this way, we experiment with topologies that contain up to a factor of $4\times$ more monitors than those that exist today (i.e., for a total of $367 \times 4 = 1468$ monitor ASes).

1) IMPACT—HIJACKERS UNAWARE OF NEW MONITORS

We first seek to understand the feasible success of realistic and omniscient hijackers if these hijackers are unaware of the new monitors added to the topology. Fig. 11 shows for a certain percentage of simulations (Y-axis) the number of ASes that baseline, realistic, and omniscient hijackers were at most able to stealthily affect as the number of monitors peering with route collectors increases by up to a factor of $4\times$ the initial monitors that exist in the original topology (IXP0).[14] Fig. 12 shows the same information when considering the fully augmented IXP topology (IXP100), instead of the original topology, with the monitors remaining the same.

As can be seen in the two figures, increasing the monitors by just a factor of $2\times$ severely degrades the feasible success of realistic and omniscient hijackers in producing stealthy interceptions. In the traditional CAIDA topology (Fig. 11), omniscient hijackers that were unaware of the new monitors fail to create stealthy interceptions for 50% of the factor $2\times$ simulations, up from the 9% of the factor $1\times$ simulations (see the corresponding factor lines in the figure). At the same time, *none* of the hijackers that succeeded in the attack were able to stealthily intercept more than 2% of the Internet topology. The stealthy interceptions results for the full-flat IXP topology (Fig. 12) are worse for realistic and omniscient hijackers. For example, omniscient hijackers fail to create stealthy interceptions for 75% of factor $2\times$ simulations. Those findings indicate why it would be important for some monitors to remain *hidden* and perhaps *not for every* monitor to disclose its routes to the public collectors.

2) IMPACT—HIJACKERS AWARE OF THE NEW MONITORS

Having understood why some monitors should remain private, we now focus on how the feasible success changes if realistic and omniscient hijackers become aware of the new monitors. Similarly to Figs. 11 and 12, Figs. 13 and 14 show for the original (IXP0) and the fully connected IXP topology (IXP100), respectively, the number of ASes that hijackers were at most able to stealthily affect if hijackers became aware of the new monitors. The results show that, by increasing the number of monitors, the baseline hijackers could not stealthily affect more than 0.18% of the topology (i.e., 130 ASes). On the contrary, realistic and omniscient hijackers that modified the routes that they export (by AS-path poisoning or prepending the forged routes) based on what the new monitors disclose were able to effectively create new hijacks that do not reach the public route collectors, even when the number of monitors is increased by a factor of $4\times$. In the original topology, realistic hijackers were still able to stealthily intercept more than 2% of the Internet in 1% of the simulations, down from 1.65% when compared to the initial monitors. Similarly, in the fully IXP-connected topology, realistic hijackers were still able to stealthily intercept more than 2% of the Internet in 14.8% of the simulations, down from 19.2% when compared to the initial monitors. Similarly, omniscient hijackers were still able to stealthily affect more than 2% of the topology in 2.5% of the simulations (down from 5.65% with the initial monitors) and 26.6% of the simulations (down from 41.1% with the initial monitors) when comparing the original and fully augmented topology (respectively).

As expected, adding more monitors reduces the number of ASes that large-scale interception attacks can affect while

[14]The initial monitors refers to the 367 monitor ASes that were identified using BGPStream.

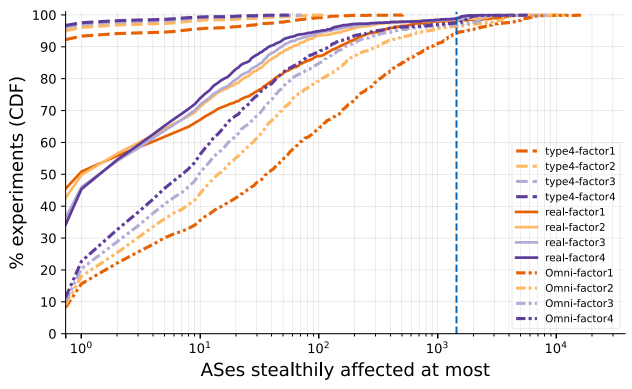


FIGURE 13. **Baseline, realistic, Omni: ASes stealthily affected as we increase the number of monitors in the traditional CAIDA topology (IXP0 topo). Hijacker is aware of the new monitors. Lines closer to X-axis mean more ASes affected.**

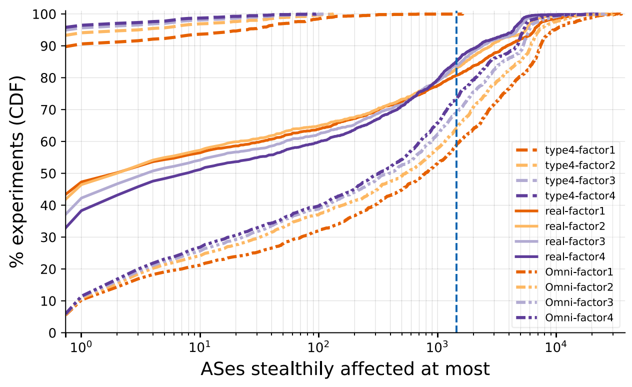


FIGURE 14. **Baseline, realistic, Omni: ASes stealthily affected as we increase the number of monitors (IXP100 topo). Hijacker is aware of the new monitors. Lines closer to X-axis mean more ASes affected.**

remaining stealthy to the public collectors. However, as our simulations show, unless many monitors are deployed, it may not be sufficient to completely eliminate all large-scale interception attacks from hijackers that deliberately react to avoid the new monitors.

Surprisingly, and contrary to the above, deploying more monitors may sometimes benefit small-scale realistic hijackers. Unlike baseline hijackers, which do not react to the new monitors, and omniscient hijackers, which already possess complete knowledge, deploying more monitors may benefit small-scale realistic hijackers which see an improvement on the success rate of their attacks. In the original CAIDA topology (Fig. 13), the percentage of failed realistic simulations decreases from 45.5% with a factor $1\times$ monitors to 34.2% with a factor $4\times$ monitors (a 11.3% improvement for the realistic hijacker). In the augmented IXP topology of Fig. 14, the percentage of realistic simulations that fail to produce stealthy interceptions decreases from 43.4% with a factor $1\times$ monitors to 32.9% with a factor $4\times$ monitors (a 10.5% improvement for the realistic hijacker). Our intuition behind this is that the realistic hijackers *learn more* about the Internet topology with more monitors. Recall that unlike omniscient hijackers, realistic hijackers do not possess complete knowledge of the routing policies that other ASes use to export their routes. Therefore, such hijackers can misclassify which monitors they need to avoid (i.e., the dangerous monitors) and which monitors they actually do not need to react against (i.e., the safe monitors). Fortunately, increasing the number of monitors causes realistic hijackers to overreact by exporting less preferred (i.e., longer) forged routes than required, ultimately leading to attacks not propagating to route collectors.

3) WHAT ABOUT SELECTING MONITOR ASES AT MORE STRATEGIC LOCATIONS?

Having evaluated the impact of increasing the number of monitors, a question that remains open is what if we had selected the monitor ASes among ASes in more strategic locations. Fig. 15 shows for each AS in the topology (shown as a percentage on the Y-axis) the percentage of times that this AS is at most affected by a stealthy attack among the 2000 simulations (X-axis) as we scale the number of IXP links in the original CAIDA topology. As we clearly observe from the figure at $X = 0$, roughly 12%, i.e., 8 640 of the ASes in the topology were ***never*** affected by either type of hijacker. This indicates the existence of potentially invulnerable ASes which are never affected by hijackers without the attack becoming visible to a route collector.[15] While further analysis and simulations are needed to verify that the above statement is accurate, one observation becomes clear: such ASes are not good locations to peer with route collectors to observe stealthy hijacks. We leave a deeper analysis of the characteristics that enable such AS locations to achieve natural immunity against stealthy hijacks for future work.

For the remaining 88% ASes (rest of the X-axis), most are rarely affected by stealthy hijacks. However, the frequency of hijack-affected ASes increases as the number of peer-to-peer IXP links increases in the topology. For example, in the traditional topology (i.e., IXP0), the realistic hijacker stealthily affects 90% of ASes in fewer than 0.5% (i.e., 10) of the simulations. In the fully connected IXP topology (i.e., IXP100), this percentage increases, with the realistic hijacker affecting 90% of the ASes in slightly more than 4% (i.e., 80) of the simulations. For the omniscient hijacker, in the traditional topology, 90% of the ASes are affected in fewer than 1.8% (i.e., 36) of the simulations. This increases to 9.6% (i.e., 192) of the simulations when considering the fully connected IXP topology. This difference in impact between the omniscient and the realistic hijacker configuration in both the traditional and the fully connected IXP topology indicates the existence of some vulnerable ASes that sufficiently knowledgeable attackers could easily hijack while remaining unnoticed by route collectors. Such ASes would appear to be good locations for monitors to reduce the *upper-bound* impact of such knowledgeable hijackers.

[15] Note that this number is much greater than the 1468 (i.e., factor-4) monitors which the hijackers deliberately avoid.

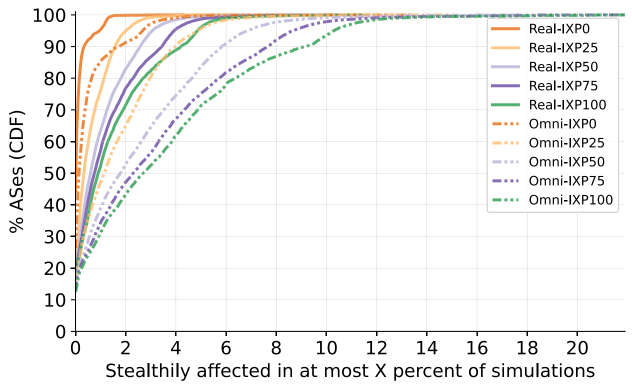


FIGURE 15. **Realistic vs Omni hijacker: Percentage of times that ASes were stealthily affected among the 2000 hijack simulations as we increase the number of ASes peering over IXPs in the topology.**

Looking at the tail of the CDFs, one can observe that there exist few ASes that are more frequently affected than the rest e.g., in more than 10% of our simulations. At first glance, such ASes would make good locations to pick as monitors to block realistic and omniscient hijackers from affecting large parts of the Internet. However, as further analysis showed, such ASes were actually both easily affected and easily avoidable locations for omniscient hijackers. We leave further analysis on the selection of better strategic locations for new monitors for future work.

4) CONCLUSIONS

By increasing the number of monitors peering with public route collectors, we reach the following conclusions:

- Increasing the number of monitors is sufficient to make large-scale stealthy attacks visible as long as the hijacker is not aware of the new monitors. As Fig. 12 shows, realistic and omniscient hijackers could not affect more than 700 ASes (1% of the topology) when the monitors simply increase by a factor of 2.
- Hijackers aware of the new monitors can redesign their attacks to potentially hide again from route collectors. realistic and omniscient hijackers could still create large-scale stealthy attacks that affect more than 2% of the Internet in 14.8% and 26.6% of the simulations (respectively).
- Naively selecting the most frequently affected AS locations as new monitors is not sufficient to deal with stealthy hijacks. While selecting such ASes as new monitors would make stealthy attacks that affect such locations visible, hijackers could also easily redesign their attacks to avoid the new monitors.

F. CORRELATION FINDINGS

Having presented our findings, the question that arises is what the properties are that allow attackers to create impactful stealthy hijacks that are not visible to route collectors. We correlated the number of ASes that hijackers stealthily affected with (*i*) the number of hijacker neighbors, (*ii*) the AS rank [48] of the hijacker, (*iii*) the AS rank of the victim, and (*iv*) the BGP distance of the hijacker from the victim based on the shortest AS-path length route the hijacker knows for the victim's prefix. Not surprisingly, according to our design of stealthy hijacks (Section V), the number of ASes the hijacker can stealthily affect shows a strong linear correlation with the number of hijacker neighbors, with a Pearson correlation coefficient of $r = 0.90$ in the traditional CAIDA topology (see Fig. 16). Then, surprisingly, the BGP distance of the hijacker from the victim follows with a weak negative correlation of $r = -0.12$, indicating a minimal relationship of hijackers closer to the victim with increased impact. Afterward, the AS rank of the hijacker follows with a correlation of $r = -0.10$ indicating an even more minimal relationship, and finally, the victim's AS rank with a correlation of $r = 0.04$ indicating almost no correlation. In topologies with more monitors, the correlation of stealthily affected ASes with the number of hijacker neighbors increases to $r = 0.94$, while in topologies with more IXP links the same correlation drops to $r = 0.8$, indicating a smaller but still strong correlation. The other properties compared (i.e., the AS rank of the victim and the hijacker) show small deviations, except for the distance of the hijacker from the victim, which shows a stronger negative correlation of $r = -0.23$ in topologies with more IXP links.

VII. REAL-WORLD EXPERIMENTS

Having understood how feasible stealthy attacks are in a simulated environment, we now focus on the feasibility of stealthy attacks in the real-Internet. As hijackers with complete knowledge of how routes propagate are commonly not viable in the real-world, we seek to understand how feasible stealthy attacks are from the perspective of hijackers with limited knowledge, such as the realistic hijacker we defined in Section V-B

Experiment Setup: We evaluate the realistic hijacker in the real-world using the PEERING Testbed [23]. We received consent to use two PEERING-owned IPv4 /24 prefixes in our experiments. The first /24 prefix belongs to the victim (ASN 61576) while the second /24 prefix belongs to the hijacker (ASN 61575). The hijacker uses its prefix to engineer the attack (i.e., compute the set of dangerous monitors) before announcing the victim's prefix. We connect the victim to the PEERING Testbed site of the University of Wisconsin (transit ASN 3128) while the hijacker connects, in our first experiment, to the PEERING Testbed site of GRNET (transit ASN 5408) and, in our second experiment, to the Amsterdam Exchange Point (transit ASNs 8283 & 12859). We chose the GRNET site because ASN 5408 is the only network peering with the PEERING testbed at that site. This simplifies the attack design. The Amsterdam Exchange Point (AMS) provides access to multiple BGP peers where we can experiment with stealthy hijacks, including the two transits: ASN 12859 and ASN 8283.

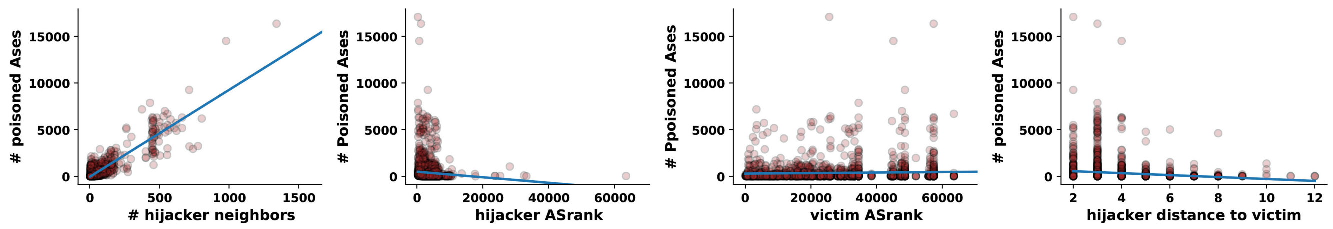


FIGURE 16. Correlation: number of stealthily poisoned ASes versus the number of hijacker neighbors, the hijacker ASRank, the victim ASRank, and the hijacker distance.

Limitations & Ground Truth: To not disrupt routing services, the Testbed limits the frequency of announcements and AS-path sizes to up to five ASes. The largest route that our hijacker can announce is the Type-4 hijack $< 61575, ASX, ASY, ASZ, 61576 >$, where ASX, ASY, and ASZ represent distinct AS numbers. However, due to ethical concerns, we could not run experiments with AS-path poisoning (see App. A). We therefore rely only on AS-path prepending allowing the hijacker to prepend its ASN in the announcements by up to three additional times. Under these restrictions, engineering an attack that was completely stealthy to all route collectors was not feasible. This section, therefore, focuses on evaluating the accuracy of the classifier in correctly labeling the monitors between *safe* and *dangerous* (i.e., part 1 of Section V-B). A dangerous (safe) monitor is a monitor that the hijacker would expect to (not) propagate the attack to route collectors. This expectation is based on the propagation of the routes for the legitimate IP prefix of the attacker (e.g. see Section VII-A for more details). We compare our classifier labeling with the ground truth collected from the routes the monitors themselves report to route collectors during BGP hijacks that we ethically perform in the real world. A good classifier accuracy for identifying dangerous monitors would indicate that there may be locations on the Internet where stealthy attack designs may be feasible with a combination of AS-path poisoning and AS-path prepending, similar to how the realistic attack was designed in simulations (see part 2 of Section V-B).

To label monitors between safe and dangerous, we experiment with two types of classifiers: (*i*) we use the classifier introduced in Section V-B, which labels each monitor according to the proximity of the monitor to the hijacker and to the victim (see Section VII-A), (*ii*) an *improved* classifier that first considers the BGP policy relations to label each monitor before computing proximities. To infer policy relations between ASes, we use the CAIDA AS relationship dataset (serial 2) [22] which labels relations based on traditional Gao-Rexford policies [38], that is, customer-to-provider, provider-to-customer, and peer-to-peer relationships (see Section VII-B for more details).

A. A PROXIMITY CLASSIFIER

1) CLASSIFICATION OF MONITORS

Same as the realistic hijacker in the simulations (see Section V-B), the hijacker in the real world needs to first compute the proximity matrix table using Equation (1) before it can classify monitors between safe and dangerous. Normally, this is achieved by the hijacker announcing a unique legitimate prefix that it owns to each neighbor. However, unlike the simulations, the hijacker in the Testbed controls a single /24 prefix. Because announcing this prefix simultaneously to all neighbors would give an incorrect neighbor to monitor proximity, we split the design of the proximity matrix into multiple announcement and withdrawal operations. In each iteration, the hijacker selects one neighbor to announce its own legitimate prefix. After waiting for 40 minutes for the BGP to converge, the hijacker collects all the routes reported for its own prefix from the monitors (i.e., the data providers of CAIDA's BGPstream [27]). The hijacker has now learned its distance to the monitors from the selected neighbor and proceeds by withdrawing the announced prefix as it has fulfilled its role. After waiting for 40 minutes for BGP to converge again, the hijacker proceeds to the next iteration, announcing its prefix to a different neighbor.

After collecting the above distances to the monitors from each selected neighbor, the hijacker proceeds to collect the distance of the victim to the monitors. The hijacker queries BGPstream [27], this time for the victim's prefix, and acquires the above information. The hijacker has now acquired the knowledge needed to compute the proximity matrix table (i.e., using eq. (1)), and thus classifies the monitors as safe and dangerous. As discussed, Type-4's are the longest forged routes that the hijacker can announce in our Testbed experiments (unlike the simulations). Therefore, the hijacker classifies all monitors with a $Prox_{M,P} <= 4$ as dangerous in the proximity table, whereas all monitors with a $Prox_{M,P} > 4$ as safe. As discussed, the hijacker would expect dangerous (safe) monitors to observe (not observe) the attack. However, misclassifications are possible as monitors may decide to propagate longer than shorter routes to the monitors. Therefore, for the remainder of this section, we focus on the accuracy, sensitivity, and specificity of the classifier and the reasons for misclassification.

2) COLLECTING GROUND TRUTH DATA

To evaluate the classifier for each selected hijacker neighbor, we individually perform Type-4 hijacks and gather ground truth information about the monitors that report the attack at each route collector. For example, Fig. 17 illustrates the collected ground truth visibility information per route collector

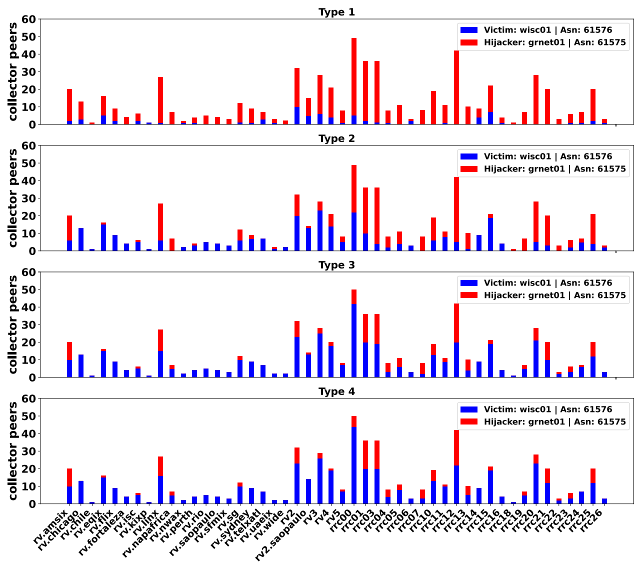


FIGURE 17. **Visibility of forged AS-paths per route collector (X-axis) and monitor (Y-axis) as the attack Type increases from a Type-1 to a Type-4 hijack.**

when the victim announces its prefix to the Wisconsin site while the hijacker announced its prefix at the GRNET site. The red bars indicate the number of monitors that observe the attack, while the blue bars indicate the number of monitors that do not observe it, per route collector. For completeness, we also show how the visibility per route collector changes from a Type-1 attack to a Type-3 attack. As expected, the figure clearly shows that the number of monitors that report the forged route decreases as the size of the forged AS-path increases. However, this reduction is smaller between a Type-3 to a Type-4 attack, indicating that peering relationships may exist for some monitors that play a more critical role than proximities (i.e., AS-path lengths). This observation leads us to develop a smarter classifier, presented in Section VII-B. We use the collected Type-4 ground truth attack data to evaluate the accuracy of the classifier for Type-4 hijacks.

3) CLASSIFIER EVALUATION

After explaining how the ground truth is collected, we now evaluate the performance of the proximity classifier. We announce the hijacker's prefix first to GRNET (AS5408), then to the two transits of the AMS site (AS8283 and AS12859) and to three Peers (AS9002, AS6461, and AS52320) each selected (*i*) due to not being among the list of monitors, and (*ii*) due to their high[16] CAIDA ASrank (ASrank 13, ASrank 8, and ASrank 17, respectively) as observed on the 15th Oct 2022. Once the hijacker classifies the monitors using its own legitimate prefix, we announce the actual Type-4 attack and collect the ground truth visibility at each location, similar to how it was previously described for the case of GRNET.

Table 6 shows the outcomes of the classification for the announcements at each location. On average, we observe 666 RIPE-RIS and Routeviews monitors[17] reporting either the valid route to the victim or the forged route to the hijacker. The exact number may slightly vary per experiment, likely due to the difference in the exported locations of the hijack announcement or due to changes in the underlying routing topology. Of these 666 monitors, on average 538 (633) monitors for transits (peers) were correctly classified as either safe or dangerous, with 302 (8) on average for transits (peers) actually reporting the hijack (i.e., ground truth). As expected, these numbers indicate that it is much easier to hide the attack when announcing it to a peer-to-peer neighbor rather than when announcing it to a transit, because transit routes propagate further and are commonly visible by multiple monitors.

We now focus on the confusion matrix values of Table 6 (i.e., TP, TN, FP, and FN). For a hijacker to create a completely stealthy attack, it first needs to identify the correct dangerous monitors from which the attack needs to be hidden. Therefore, with this in mind, we focus on how accurate a classifier would be to predict such dangerous monitors. We define the following classification terms:

- **TP**: monitors correctly classified as dangerous.
- **TN**: monitors correctly classified as safe.
- **FP**: monitors incorrectly classified as dangerous.
- **FN**: monitors incorrectly classified as safe.
- **Accuracy**: Ratio of correctly classified monitors over the total number of monitors. A value of 1 (0) indicates that all monitors have been correctly (incorrectly) classified.
- **Sensitivity**: Ratio of truly classified dangerous monitors over the total number of actual dangerous monitors, i.e., $TP/(TP+FN)$. A value of 1 (0) would mean that all dangerous monitors have been correctly (incorrectly) classified as dangerous (safe).
- **Specificity**: Ratio of truly classified safe monitors over the total number of actual safe monitors, i.e., $TN/(TN+FP)$. A value of 1 (0) indicates that all safe monitors have been correctly (incorrectly) classified as safe (dangerous).

Given the above definitions, the ideal classifier that would enable an impactful and completely stealthy attack would be one that correctly classifies every dangerous monitor, i.e., a *sensitivity* $=$ 1 (required property), while misclassifying as few actual safe monitors as possible, i.e., a specificity as close to 1 (desirable property). Although completely stealthy attacks may still be created by pure luck (e.g. see the baseline hijacker in simulations), requiring a sensitivity of 1 guarantees a completely stealthy attack if the hijacker can avoid

[16]Highly ranked ASes, if stealthily affected, would lead to a more impactful stealthy attack.

[17]There are 483 (367 active) monitor ASes but some may peer using multiple BGP devices, for a total of 666 monitor devices peering with route collectors.

TABLE 6. Statistics for the proximity classifier. The statistics presented are when comparing the classifier estimation vs. the ground truth for a Type-4 hijack.

Classifier Statistics	**GRnet Transit ASN 5408**	**AMS Transit ASN 8283**	**AMS Transit ASN 12859**	**AMS Peer ASN 9002**	**AMS Peer ASN 6461**	**AMS Peer ASN 52320**
# monitors (Total)	663	695	683	652	653	653
# monitors (Correctly Classified)	520	520	575	637	611	652
# monitors report hijack (Ground Truth)	162	417	329	10	9	6
TP	22	259	250	10	1	6
TN	498	261	325	627	610	646
FP	3	17	29	15	34	1
FN	140	158	79	0	8	0
Accuracy	0.78	0.74	0.84	0.97	0.93	0.99
Sensitivity	0.13	0.62	0.75	1.00	0.10	1.00
Specificity	0.99	0.93	0.91	0.97	0.94	0.99

TABLE 7. The reasons behind the proximity classifier misclassifications.

Proximity Classifier: Reason for Misclassification (FP / FN)	**GRnet Transit ASN 5408**	**AMS Transit ASN 8283**	**AMS Transit ASN 12859**	**AMS Peer ASN 9002**	**AMS Peer ASN 6461**	**AMS Peer ASN 52320**
1. Shortest AS-Path Violation	**FP: 1 FN: 140**	**FP: 2 FN: 158**	**FP: 0 FN: 79**	**FP: 0 FN: 0**	**FP: 1 FN: 8**	**FP: 0 FN: 0**
a) Longer Path preferred	*FP: 0 FN: 139*	*FP: 1 FN: 157*	*FP: 0 FN: 79*	*FP: 0 FN: 0*	*FP: 1 FN: 0*	*FP: 0 FN: 0*
b) Victim Path not observed	*FP: 1 FN: 0*	*FP: 1 FN: 0*	*FP: 0 FN: 0*	*FP: 0 FN: 0*	*FP: 0 FN: 0*	*FP: 0 FN: 0*
c) Hijacker Path not observed	*FP: 0 FN: 1*	*FP: 0 FN: 1*	*FP: 0 FN: 0*	*FP: 0 FN: 0*	*FP: 0 FN: 8*	*FP: 0 FN: 0*
3. Tie breakers Violations	**FP: 2 FN: 0**	**FP: 15 FN: 0**	**FP: 29 FN: 0**	**FP: 15 FN: 0**	**FP: 33 FN: 0**	**FP: 1 FN: 0**
d) Victim path preferred	*FP: 2 FN: 0*	*FP: 15 FN: 0*	*FP: 29 FN: 0*	*FP: 15 FN: 0*	*FP: 33 FN: 0*	*FP: 1 FN: 0*
Total (FP / FN)	**FP: 3 FN: 140**	**FP: 17 FN: 158**	**FP: 29 FN: 79**	**FP: 15 FN: 0**	**FP: 34 FN: 8**	**FP: 1 FN: 0**

the dangerous monitors. The second property is optional, as incorrectly classifying actual safe monitors would lead the hijacker to only *overreact* but *not reveal* the attack. However, it is still a desirable property to have as overreacting would cause the hijacker to announce longer forged paths and, therefore, produce a less-preferred attack.

Having discussed the ideal and desirable properties of the classifier, we now revisit Table 6. Not surprisingly, a classifier based on proximity has a very low sensitivity ($average \approx 0.5$) due to the high number of FNs caused by announcing the attack to transit neighbors. As expected, monitors sometimes prefer the longer routes forged by the hijacker rather than the shorter routes announced by the victim due to locally assigned preferences. On the other hand, from what we observe by looking at peers (i.e., AS9002, AS6461, and AS52320), a classifier based on proximity usually shows a very high sensitivity, probably because shortest path relations play a more important role for such peering networks. With the exception of the outlier AS6461 (which we will explain later), both AS9002 and AS52320 show a perfect $sensitivity = 1$ indicating that it is possible to correctly classify all dangerous monitors when announcing the attack to peers.

To our surprise, an unexpected trend shows when we looked at the specificity values of Table 6. Despite the sometimes low sensitivity values, the proximity classifier correctly classifies the majority of safe monitors both for announcements made to transits (average $specificity = 0.94$) and for announcements made to peers (average $specificity = 0.96$). These low-sensitivity but high-specificity values suggest that the proximity classifier is overestimating the number of safe monitors. Normally, we would expect the opposite to hold true because the classifier always marks equally-distant from the hijacker and the victim monitors as dangerous.

To better understand the causes of incorrect classification, we analyze the reasons behind the incorrect inferences in Table 7. We compare the actual safe and dangerous monitors (ground truth) with the expected safe and dangerous monitors (classifier's prediction). Based on the choices made by the classifier, we identify the following reasons for misclassification:

TABLE 8. Statistics for the Gao-Rexford classifier. The statistics presented are when comparing the classifier estimation vs. the ground truth for a Type-4 hijack.

Classifier Statistics	GRnet Transit ASN 5408	AMS Transit ASN 8283	AMS Transit ASN 12859	AMS Peer ASN 9002	AMS Peer ASN 6461	AMS Peer ASN 52320
# monitors (Total)	663	695	683	652	653	653
# monitors (Correctly Classified)	600	641	609	634	609	651
# monitors report hijack (Ground Truth)	162	417	329	10	9	6
TP	154	400	320	10	1	6
TN	446	241	289	624	608	645
FP	55	37	65	18	36	2
FN	8	17	9	0	8	0
Accuracy	0.90	0.92	0.89	0.97	0.93	0.99
Sensitivity	0.95	0.96	0.97	1.0	0.10	1.00
Specificity	0.89	0.86	0.81	0.97	0.94	0.99

TABLE 9. The reasons behind the Gao-Rexford classifier misclassifications.

Gao Rexford Classifier Reason for Misclassification (FP / FN)	GRnet Transit ASN 5408	AMS Transit ASN 8283	AMS Transit ASN 12859	AMS Peer ASN 9002	AMS Peer ASN 6461	AMS Peer ASN 52320
1. Gao Rexford Violation	**FP: 52 FN: 0**	**FP: 27 FN: 0**	**FP: 48 FN: 0**	**FP: 3 FN: 0**	**FP: 2 FN: 0**	**FP: 1 FN: 0**
a) customer - provider	*FP:1 FN:0*	*FP: 0 FN: 0*	*FP: 0 FN: 0*	*FP: 0 FN: 0*	*FP: 0 FN: 0*	*FP: 0 FN: 0*
b) customer - peer	*FP: 0 FN: 0*	*FP: 6 FN: 0*	*FP: 20 FN:0*	*FP: 0 FN: 0*	*FP: 0 FN: 0*	*FP: 0 FN: 0*
c) peer - provider	*FP: 51 FN: 0*	*FP: 21 FN: 0*	*FP: 28 FN:0*	*FP: 3 FN: 0*	*FP: 2 FN: 0*	*FP: 1 FN: 0*
2. Shortest AS-Path Violation	**FP: 1 FN: 8**	**FP: 2 FN: 17**	**FP: 0 FN: 9**	**FP: 0 FN: 0**	**FP: 1 FN: 8**	**FP: 0 FN: 0**
d) Longer Path preferred (Same Gao relation)	*FP:0 FN: 4*	*FP: 0 FN: 13*	*FP: 0 FN: 9*	*FP: 0 FN: 0*	*FP: 1 FN: 0*	*FP: 0 FN: 0*
e) Longer Path preferred (Unknown relation)	*FP: 0 FN: 3*	*FP: 1 FN: 3*	*FP: 0 FN: 0*	*FP: 0 FN: 0*	*FP: 0 FN: 0*	*FP: 0 FN: 0*
f) Victim Path not observed	*FP: 1 FN: 0*	*FP: 1 FN: 0*	*FP: 0 FN: 0*	*FP: 0 FN: 0*	*FP: 0 FN: 0*	*FP: 0 FN: 0*
g) Hijacker Path not observed	*FP: 0 FN: 1*	*FP: 0 FN: 1*	*FP: 0 FN: 0*	*FP: 0 FN: 0*	*FP: 0 FN: 8*	*FP: 0 FN: 0*
3. Tie breakers Violations	**FP: 2 FN: 0**	**FP: 8 FN: 0**	**FP: 17 FN: 0**	**FP: 15 FN: 0**	**FP: 33 FN: 0**	**FP: 1 FN: 0**
h) Victim path preferred	*FP: 2 FN: 0*	*FP: 8 FN: 0*	*FP: 17 FN: 0*	*FP: 15 FN: 0*	*FP: 33 FN: 0*	*FP: 1 FN: 0*
Total (FP / FN)	**FP: 55 FN: 8**	**FP: 37 FN: 17**	**FP: 65 FN: 9**	**FP: 18 FN: 0**	**FP: 36 FN: 8**	**FP: 2 FN: 0**

(a) The monitor (or an AS towards the monitor) prefers longer rather than shorter routes.
(b) The monitor did not report the victim's route during the proximity learning period.
(c) The monitor did not report the hijacker's route during the proximity learning period.
(d) The monitor prefers the victim's route, while the proximity to the hijacker and to the victim is the same.

From the above reasons, case (a) corresponds to shorter AS-path violations. Normally, a proximity classifier would expect ASes to propagate their shortest route to the monitors for the hijacked prefix. However, this is not always the case in reality. Cases (b) and (c) correspond to rare corner situations where the monitors did not report either the victim's prefix (case b) or the hijacker's legitimate prefix (case c). Normally, if a monitor does not report the victim's prefix but reports the hijacker-owned prefix, the classifier would expect the monitor to always be dangerous for any Type hijack (i.e., $Prox_{M,P} = \infty$). Rarely, however, this may not be the case. Similarly for the opposite scenario (c). Normally, if the monitor does not report the hijacker-owned prefix, the classifier would expect the monitor to always be safe for any Type hijack (i.e., $Prox_{M,P} = -\infty$). Rarely, however, this may again not be the case. Finally, case (d) captures the wrong inferences of our conservative assumption i.e., that monitors equally-distant from the hijacker and the victim are always labeled as dangerous.

As Table 7 shows, the vast majority of actual dangerous falsely classified as safe monitors (FN) are due to ASes that propagate longer than shorter routes, likely due to locally configured preferences with higher priority than shortest path preferences for the hijacked prefix. On average, we note 125 out of 126 such violations for transits. On the other hand, shortest path violations have hardly misclassified any actual safe monitor as dangerous (FPs) in our experiments (average of 1 violation for the transits). As expected, the vast majority of falsely classified as dangerous monitors are due to violations in our tie-breaker assumption. We leave the problem of better labeling such monitors between safe and dangerous as a potential future work.

Regarding the false inferences due to the corner situations of case (b) & case (c) in Table 7, as expected, they are quite rare (average of less than 1 FP and 2 FNs among all our experiments). However, all falsely classified dangerous monitors for the outlier (AMS peer 6461) were due to this corner situation. A possible reason behind this could be unexpected routing changes that caused the forged route to propagate to the monitor during the actual attack. Another reason could be due to the choices made by individual monitors who decided to report suspicious forged route during the attack. Either way, to identify the exact reason behind this, more dedicated experiments would be required – something we did not plan for in this work. A more stimulating question to ask would be whether the above corner cases could be prevented if we had extended the time of proximity learning period to more than 40 minutes. Given that these monitors did not report the hijacker's prefix within a 40 minute period, we consider this scenario unlikely, since BGP usually stabilizes within just a few minutes [16].

4) CONCLUSION

Having performed the above limited but in depth real-world analysis on the possibility of stealthy hijacks, we reach the following conclusions:

- A classifier based on proximities could be sufficient for the hijacker to correctly identify the dangerous monitors for announcements made to peers. This is what the ground truth announcements suggest for AS9002 and AS52320 with the exception of the routing anomaly in AS6461.
- A classifier based on proximities is not sufficient for the hijacker to correctly identify the dangerous monitors for announcements made to transits. This is what the high number of FNs suggest for all transit announcements (AS5408, AS8283, and AS12859).

B. A MORE ADVANCED CLASSIFIER

Having analyzed the proximity classifier, we now proceed to design an improved classifier that captures BGP policies based on the traditional Gao-Rexford model [38], that is, customer paths are preferred over peer-to-peer paths which are preferred over provider paths. We seek to understand whether Gao-Rexford policies offer the appropriate granularity of information for the hijacker to correctly classify the monitors between safe and dangerous for hijacks announced to transits. As before, we verify the new classifier in the same PEERING Testbed experiments by simply replacing the old classifier predictor with the new one. The ground truth about how we collected the actual safe and dangerous monitors has already been explained in Section VII-A (see the ''collecting ground truth data'').

1) AN EXAMPLE

We now explain the Gao-Rexford classifier with the help of a real-world example taken from our experiments and presented in Fig. 18. The figure shows the two routes reported by the monitor (AS2895) to the public route collector (RRC13). The route originating from AS61576 corresponds to the prefix owned by the victim while the route originating from AS61575 corresponds to the legitimate prefix owned by the hijacker announced to classify the monitors (same as before). The arrows show the direction that the two routes followed from each origin to the route collector. As the figure shows, the two routes meet at AS 3267. Assuming that the hijacked prefix propagates the same way (or in a very similar way) as to the hijacker's legitimate prefix, the hijacker expects AS3267 to be the one responsible for deciding which of the two routes will actually propagate to the route collector. AS3267 observes the victim over a route of 5 AS hops, i.e., $\{2603, \ldots, 61576\}$, and the hijacker over a route of 3 AS hops, i.e., $\{8283, \ldots, 61575\}$. The proximity of the hijacker to AS3267 (and thus the monitor) is therefore 2 hops closer than the proximity of the victim to AS3267 meaning that Type-2 hijacks or below are dangerous to propagate to the route collector. A classifier based on proximity will therefore incorrectly infer for Type-4 attacks, such as the one the hijacker plans to announce, that the monitor is safe. Meanwhile, a classifier based on relations would first look at the relation of AS8283 to AS3267, and then at the relation of AS2603 to AS3267 before deciding on whether the hijacker's route or the victim's route propagates to the route collector. If AS3267 prefers more AS8283, the relation classifier would then infer that the monitor is dangerous as the hijacker replaces the victim's route. Meanwhile, if AS3267 prefers more AS2603, the relation classifier would then infer that the monitor is safe, as AS3267 would not propagate the hijacker's route during the attack. In the case of tied relations, the hijacker classifies the monitors based on proximities (i.e., the previous classifier), as it assumes that the proximity of the monitor to the hijacker and to the victim is the most determining factor for accurately classifying the monitors.

In this example, AS3267 receives the hijacker's route over a peer (AS8283), and the victim's route over a provider (AS2603). We derive those relations from the CAIDA AS-relation dataset. Unlike the proximity-based classifier, the Gao-Rexford-based classifier correctly classifies the monitor as dangerous. The route that propagates to the route collector during the actual hijack (ground truth) is the route over

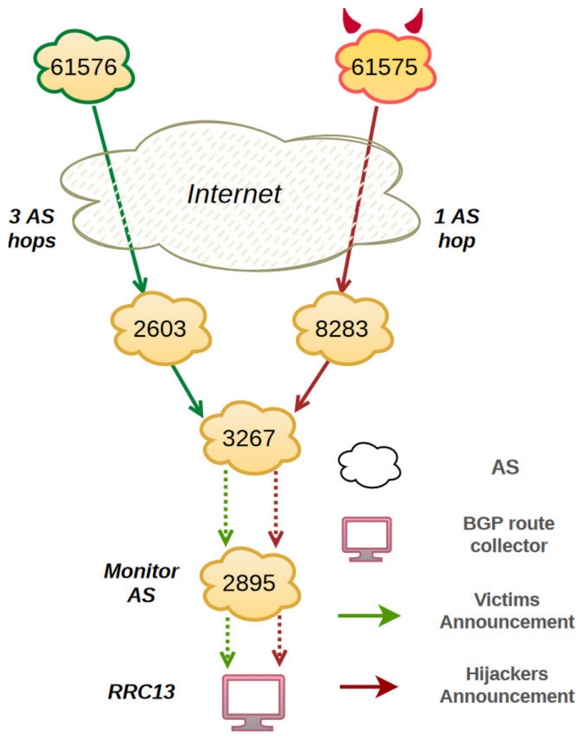


FIGURE 18. Example of an AS responsible for the route that propagates to the route collector. Arrows show the AS-path from each origin (hijacker AS 61575, victim AS 61576) towards the route collector. Dashed lines mean only one of the two routes propagates during the attack.

ASN8283, i.e., {2895, 3267, 8283, …, 61575, …, 61576}, where {61575, …, 61576} is the forged part of the hijack announcement.

2) CLASSIFIER EVALUATION

Similar to the proximity classifier, we now evaluate the performance of the Gao-Rexford classifier. Table 8 shows the outcome of the monitor classification for the Type-4 attacks announced at each location. As we observe, when comparing the proximity-based classifier with the Gao-rexford-based classifier, the classification accuracy increases severely from 0.78 to 0.90 for announcements made to transits (average values). At the same time, it remains virtually the same for announcements made to peers, as relations are not that important for such neighbors. For the proximity classifier, most of the wrong inferences were due to a high number of actual dangerous monitors incorrectly classified as safe (i.e., ≈125 FNs on average for transits). On the other hand, the Gao-Rexford classifier severely reduces those FNs by more than 91% (≈11 remaining) at the cost of incorrectly classifying some actual safe monitors as dangerous (i.e., an average FP increase 13 to 52, At the same time, the classification sensitivity increases from ≈0.5 to 0.96 (average values), while the classification specificity reduces from 0.94 to 0.85 (average values). Recall that a *sensitivity* ≈ 1 is the property required to enable the creation of stealthy attacks. A value of 1 would mean that the hijacker has correctly classified all the dangerous monitors from which it needs to hide the attack. The Gao-Rexford classifier is a first step towards making such stealthy attacks feasible in practice.

We now examine the reasons behind the Gao-Rexford-based classifier misclassifications. Similar to Table 7 of the proximity classifier, Table 9 illustrates the reasons behind the Gao-Rexford-based classifier misclassifications. Based on the AS responsible for deciding the route[18] that propagates to the route collector (i.e., AS3267 in the example above), these reasons can be grouped into the following categories:

(a) *Gao-Rexford violations*. The AS responsible propagated (*i*) a provider route over a customer route (case a), or (*ii*) a peer route over a customer route (case b), or (*iii*) a provider route over a peer route (case c).

(b) *Shortest AS-path violations*. The AS responsible propagated the longer route while: (*i*) the two routes were of a tied Gao-Rexford relation (case d), (*ii*) the relations were unknown, i.e., either one missing from the CAIDA dataset (case e), or (*iii*) and (*iv*) either the victim's route (case f) or the hijacker's route (case g) were not reported by the monitor, respectively (similar to cases (b) and (c) in Table 7).

(c) *Tie-breaker violations*. The AS responsible preferred to propagate the victim's path (case h). Similarly to the proximity classifier, the Gao-Rexford classifier assumes that all tied-length routes are potentially dangerous.

As Table 9 shows, most of the monitors that were incorrectly classified as dangerous (FPs) are due to Gao-Rexford violations. We order those violations in the table in decreasing order, from serious to less serious, based on what we expect to be the economic incentives of networks to accept (and export) routes from (to) their neighbors. As expected, the rarest violations are those in which the provider route was preferred over a customer. We observed a total of one such violation during all our experiments, where the AS responsible chose to propagate a provider route over a customer route to the route collector. We then observe a total of 26 violations (summed values) where the AS responsible chose to propagate a peer over a customer route. Finally, we observe a total of 106 violations (summed values over all experiments) where the AS responsible chose to propagate a provider route over a peer route. As expected, the majority of ASes propagate routes based on the expected profit. The large number of peer-provider violations may indicate local relations dependent on prefixes or wrongly inferred relations in the CAIDA dataset.

However, when looking at the incorrectly classified safe monitors (FNs), surprisingly, ***none*** of the FNs were due to Gao-Rexford violations. This indicates that, while Gao-Rexford could introduce some false positives, Gao-Rexford relations offer the appropriate granularity of information for the hijacker to *not misclassify* the dangerous monitors. Would this suggest that Gao-Rexford is sufficient

[18]Refers to the hijacker's route and to the victim's route.

for the hijacker to correctly identify the dangerous monitors? The answer is no. When looking at the AS-path length violations, we quickly see that the majority of False Negatives come from equally preferred Gao-Rexford neighbors where the AS responsible chooses to propagate the longer AS-path. As expected, this indicates that more complex routing policies exist, which are not captured by simple Gao-Rexford relations.

3) CONCLUSIONS

Having done the above analysis, our results indicate that:

- Gao-Rexford policies offer the appropriate granularity of information for the hijacker to *not misclassify* the actual dangerous monitors that will observe the attack.
- A classifier based on Gao-Rexford policies, while severely reducing misclassifications by 91%, alone is *not sufficient* for the hijacker to correctly classify all monitors that will observe the attack for hijacks announced to transits. As Table 9 suggests from the FNs in step (d), more complex policies exist that are not captured by simple Gao-Rexford relations.

VIII. FREQUENTLY ASKED QUESTIONS & FUTURE WORK

Having analyzed stealthy attacks in-depth in simulations and in the real-world, we now focus on three core questions: (*i*) How feasible are stealthy hijacks against route collectors in today's Internet? (*ii*) What could operators do better to defend against stealthy hijacks? and (*iii*) What could hijackers do better to enable stealthier attacks?

A. HOW FEASIBLE ARE STEALTHY ATTACKS ON TODAY'S INTERNET?

There are two properties that would enable completely stealthy hijacks to route collectors. First, the potential of the hijacker to correctly identify all dangerous monitors. Second, the potential of the hijacker to circumvent attacks from propagating from dangerous monitors to route collectors. We have shown in simulations that omniscient hijackers with *complete* knowledge about the inter-domain routing policies of other ASes accurately identify all dangerous monitors without failure. Realistic hijackers with *no* knowledge or *limited* knowledge can still identify (in most circumstances) all dangerous monitors for forged routes announced to customers and peer-to-peer neighbors (i.e., see the low responsibility % for realistic hijackers in Table 5 and the fact that realistic hijackers correctly classified all dangerous monitors for ASN 9002 and ASN 52320 in Table 7). Although this is not a golden rule, since misclassifications are possible (i.e., 3% for p2p in simulations in Table 5 and 8 FN for ASN 6461 in Table 5), we believe that performing stealthy attacks would be feasible in some parts of the Internet, with real-world attackers potentially already capitalizing on some of the techniques [49]. We leave a more thorough investigation on the feasibility of performing stealthy attacks from different geographical locations as potential future work. We also leave the investigation of unexpected routing deviations, such as this of ASN 6461, their frequency of occurrence, and how the hijacker could design the attack around such deviations for potential future work.

Regarding the ability of the hijacker to circumvent attacks from reaching route collectors, a core mechanism that would enable that, especially for transits, would be AS-path poisoning. In this paper, we assumed that monitor ASes are vulnerable to AS-path poisoning. Although for ethical reasons (see Appendix A), we did not perform real-world experiments with AS-path poisoning, we strongly believe that this would be the case. However, previous work dedicated to AS-path poisoning [50] has noted that not all parts of the Internet are vulnerable to AS-path poisoning. We leave a more dedicated analysis to discover vulnerable locations as a potential future work.

B. WHAT COULD OPERATORS DO BETTER TO DEFEND AGAINST SUCH ATTACKS?

From the perspective of the current RIPE-RIS and Route-views route collectors, there are two reasons why hijackers are able to hide their attacks from traditional route collectors (as also explained in Section II). First, the fact that route collector peers (i.e., the monitors) propagate their best routes to route collectors [51]. Second, the fact that the RIPE-RIS and Routeviews route collectors publicly disclose the recorded routes. A hijacker that observes the routes reported by such public route collectors can take advantage of the disclosed information to design attacks that are not observable by route collectors. We have two recommendations to hinder hijackers from designing stealthy attacks. First, peers should propagate more routes to route collectors than just the best route. Second, public route collectors should perhaps not disclose all routes but perhaps keep some (new) peers hidden from public view.[19] For example, by enabling BMP [32] (BGP Monitoring Protocol) support, peers could disclose all routes they have in their Adj-RIB-In table. For peers that consider intrusive the disclosure of all the routes, reporting the longest route could perhaps be sufficient, given that hijackers purposely forge longer routes to hide from route collectors.[20] The second suggestion, although against the objective of a public route collector, it would help deal with stealthy hijacks as long as the hijacker does not become aware of those new private monitors (as we have also shown when comparing Figs. 11 and 12 with Figs. 13 and 14).

C. IS THERE A RISK FOR EVEN SMARTER STEALTHY HIJACKERS?

So far in this work, we have experimented with two classifiers that hijackers could utilize to identify dangerous monitors: one based on proximities and another based on Gao-Rexford

[19]Existing peers should remain public as BGP anomaly detection solutions may rely on them.

[20]Hijackers that utilize other methods instead of announcing longer routes, e.g., BGP communities, would still remain hidden under this approach.

relations. Both classifiers have various limitations that lead to the misclassifications presented in Table 7 and 9. As Table 2 shows, different regions on the Internet have different conformities to Gao-Rexford relations. Dedicated hijackers could design more specialized classifiers taking into account their relevant geographical location, the geographical location of the victim, and the geographical location of the route collector they are trying to avoid. For example, hijackers could pre-analyze Table 2 to learn patterns on how other networks select their best routes. This analysis would help hijackers improve the identification of dangerous monitors. Other hijackers, to design stealthier attacks, could focus on accurate prediction of network paths. Although accurate AS-path prediction was not the goal of our work, inference of how routes propagate would enable hijackers to improve the circumvention of attacks around dangerous monitors. For example, hijackers could utilize systems such as PredictRoute [43] or RouteInfer [52] as well as other knowledge, e.g., from papers such as [40], [45], etc., to build more advanced classifiers that better avoid the monitors. More dedicated hijackers could actively poison AS-paths to deny networks of their best routes and therefore learn the preference of backup routes [41]. Finally, hijackers could also experiment with BGP communities during the preparation phase of the attack to design more surgical attacks [28] that better avoid the monitors. We leave further investigation on the potential of stealthy hijackers for future work.

IX. RELATED WORK

We now focus on a review of stealthy BGP attacks in the literature and alternative solutions that could deal with stealthy hijacks to control-plane route collectors. Although none of the existing works have rigorously investigated the capabilities of hijackers to avoid public route collectors,[21] existing works have focused on the design properties of stealthy hijacks and the discovery of previously unnoticed hijackers.

Birge-Lee et al. [28] used BGP communities to create surgical hijacks. Communities are *optional* transitive attributes that can be inserted into BGP messages for customers to influence the propagation and filtering policies of their providers. While communities could be leveraged to design stealthy hijacks, neither the support nor propagation of these attributes is guaranteed by every transit provider network. As communities could introduce security vulnerabilities, some networks have even been reported to remove them from BGP advertisements [54]; an unintended outcome for hijackers wishing to remain unnoticed. In contrast, McArthur et al. [55] focused on increasing the AS-path length and exporting hijacks to specific neighbors, techniques that we also utilize in this study. Although these techniques cause hijacks to propagate to a smaller part of the Internet, thus enabling stealthier attacks, none of the previous studies have focused on the capability of hijackers to deliberately avoid public route collectors.

[21]Except for the thesis by Milolidakis A. [53], which this article consequently improves.

Other studies, such as those of Testart et al. [56] and Vervier et al. [57], have focused on identifying hard-to-notice persistent hijackers that utilize either unallocated prefixes or spread their attacks throughout time to disguise their advertised routes as valid. The authors of these studies used machine learning and other path score-based techniques to identify potentially malicious patterns that match those of previously identified hijackers. As a consequence, these patterns need to be observed by public route collectors, which would not help against hijackers that deliberately avoid such collectors.

To deal with the limited visibility of control-plane monitors, alternative solutions commonly rely on data-plane signals to detect attacks, for example those collected by active probing measurements such as pings and traceroutes [58], [59], [60], [61]. Although such works could potentially detect stealthy in control-plane hijacks, data-plane systems have been harder to commercialize due to noise introduced from data-plane disruptions, such as link failures, congestion, etc. Other notable works have focused on proactive-based mitigation of hijacks instead of reactive-based detection to remove attacks before they propagate far on the Internet. The most common technique is the Resource Public Key Infrastructure (RPKI [62]) which digitally signs each prefix to associate it with its legitimate AS origin. RPKI can eliminate Type-0 hijacks, yet as we show in this paper, impactful stealthy attacks are still feasible by increasing the hijack Type. In particular, Artemis [16] could offer quick detection and mitigation of Type-1 and potentially higher hijacks, but still relies on the attack being visible to the route collectors.

Finally, a recent alternative work by Alfroy et al. [63] on public route collectors (i.e., the RIPE-RIS and Routeviews route collectors) has focused on identifying the most valuable peers that monitoring solutions should monitor. While this work provides useful insights into how vantage points should be selected, caution is needed, as monitoring from fewer locations may enable stealthier hijacks to public route collectors.

X. CONCLUSION

Internet-wide BGP route collector infrastructures are clearly effective in reporting traditional hijack attacks. In this work, we made the first steps towards investigating the potential existence of hijackers able to avoid public route collectors. Our simulations show that hijackers that observe the routes that public route collectors disclose are able to modify their attacks and forge routes not reported by public route collectors. Such hijackers thrive in networks with many peer-to-peer links, can intercept traffic from many ASes, and are hard to prevent by naively increasing the monitors.

Our real-world experiments support the potential existence of such stealthy hijackers. As we show, a simple proximity heuristic could be sufficient for a hijacker to predict whether or not an attack would be observed by a monitor. We believe

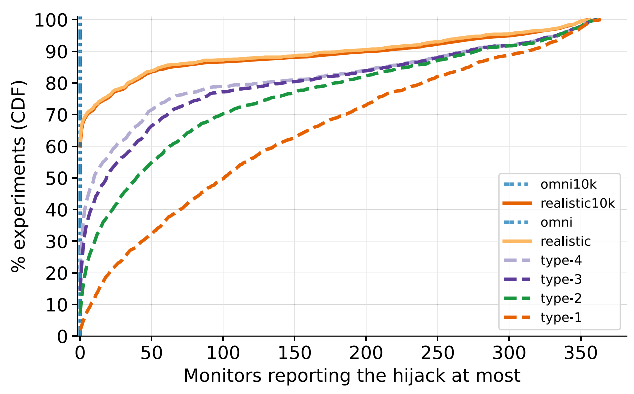


FIGURE 19. Baseline (Type-1 to Type-4), realistic, omniscient hijackers: Number of monitors reporting the attack in the traditional CAIDA topology. Lines closer to the X-axis means more visible attacks.

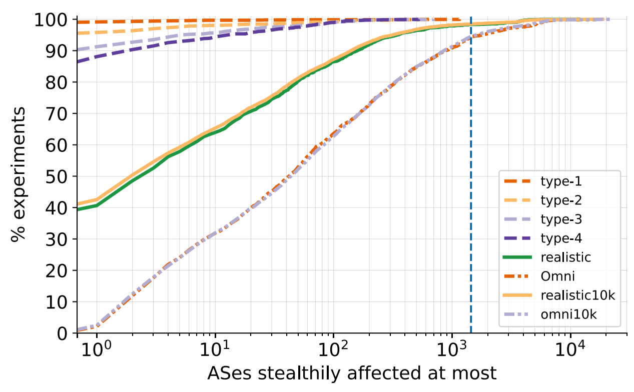


FIGURE 20. Baseline, realistic, omniscient: Number of ASes affected by stealthy interceptions in the traditional CAIDA topology. Lines closer to X-axis mean more ASes affected. The dashed vertical line illustrates 2% of the topology (i.e., 1440 ASes).

that our results will spur additional research on hijack detection solutions.

APPENDIX

A. ETHICAL CONSIDERATIONS

We stress the fact that our attack techniques do not reveal new fundamental vulnerabilities of BGP, we rather carefully leverage existing vulnerabilities. Our goal here is to show that previous analysis have largely underestimated the impact of sophisticated stealthy hijacks. For real-world experiments, we followed the guidelines of the PEERING testbed. The frequency of our announcements did not exceed the limit of one per 40 minutes, and the experimental prefixes that were given to us did not host any traffic services. We always used the correct origin for our experiments and hijacked only the prefixes provided by the testbed. Therefore, no production services were affected by our attacks.

We initially planned to conduct experiments using AS-path poisoning. Following the Peering testbed guideline, we informed the Nanog community about our planned experiments using AS-path poisoning and we asked operators to opt-out from the experiment. However, a certain fraction of network operators raised concerns about the ethical usage of someone else's ASN. The concerns raised include unnecessary triggering of security alerts, reputation lost due to wrongly inferred AS relationships, and more complex post-mortem analysis of BGP data within an organization [64]. We finally note that a few previous papers have performed AS-path poisoning experiments on the real-world. For these previous papers, we could not find any discussion on the network operators' mailing lists. This is particularly true for the work of Smith et al. [50], which claims to have informed the Nanog mailing list, but we could not find any such messages.

TABLE 10. Number of monitors reporting the hijack per percentile for different percentages of IXP links. 0% IXP links refers to the traditional CAIDA topology, while 100% refers to the full topology with all the IXP links.

Percentile Experiments	*Percentage of IXP Links*				
	0%	**25%**	**50%**	**75%**	**100%**
	Baseline Type-1 Hijacker				
70th	185	167	154	146	135
80th	239	210	192	179	169
85th	272	239	215	199	191
90th	314	273	250	236	226
95th	340	313	292	269	259
100th	365	364	365	364	365
	Baseline Type-4 Hijacker				
70th	47	42	39	39	35
80th	128	107	87	78	73
85th	212	168	137	118	103
90th	269	216	179	149	132
95th	334	280	245	209	187
100th	363	363	363	363	363
	Realistic Hijacker				
70th	5	5	4	4	4
80th	34	32	30	30	27
85th	57	52	51	50	48
90th	182	148	123	108	94
95th	283	233	194	165	145
100th	357	356	356	356	356
	Omni Hijacker				
70th	0	0	0	0	0
80th	0	0	0	0	0
85th	0	0	0	0	0
90th	0	0	0	0	0
95th	0	0	0	0	0
100th	0	0	1	1	1

B. RUNNING MORE THAN 2K SIMULATIONS

Similar to Figs. 6 and 7, Figs. 19 and 20 show the number of monitors reporting the hijack and the ASes affected by the attack (respectively) when we perform 10K instead of 2K simulations. For ease of comparison, we also plot in these figures the results of the realistic and omniscient hijackers for the 2K simulations (label realistic and Omni, respectively). As we show for both the realistic and omniscient hijackers, the 10K lines almost overlap with the 2K lines in the figures. This overlap indicates that 2K simulations are sufficient (as a

TABLE 11. Number of ASes stealthily poisoned by the hijack per percentile for different percentages of IXP links. 0% IXP links refers to the traditional CAIDA topology, while 100% IXP links refers to the full topology with all the IXP links.

Percentile Experiments	*Percentage of IXP Links*				
	0%	**25%**	**50%**	**75%**	**100%**
	Baseline Type-1 Hijacker				
90th	0	0	0	0	0
95th	0	0	0	0	0
100th	1158	374	598	481	347
Topology %	1.6	0.5	0.8	0.6	0.4
	Baseline Type-4 Hijacker				
90th	0	2	2	1	1
95th	7	21	31	26	24
100th	514	915	1460	1540	1630
Topology %	0.7	1.2	2.0	2.1	2.2
	Realistic Hijacker				
50th	1	2	2	2	2
70th	14	166	233	279	319
80th	43	540	835	1098	1317
85th	75	963	1557	1983	2363
90th	147	1673	2636	3232	3841
95th	333	2390	3979	5336	6571
100th	11731	14982	22437	25923	32967
Topology %	16.2	20.7	31.0	35.9	45.5
	Omni Hijacker				
50th	37	361	578	682	787
70th	157	1257	1963	2455	2718
80th	327	2112	3340	4273	4818
85th	501	2567	4118	5422	6419
90th	868	3154	4910	6196	7342
95th	1750	4995	6805	8330	9635
100th	17005	18144	24517	30121	35409
Topology %	23.5	25.1	33.9	41.7	49.0

number) to represent what realistic and omniscient hijackers are capable of in simulations.

C. VISIBILITY AND IMPACT–INCREASING NUMBER OF IXP LINKS

To better clarify Fig. 9 and Fig. 10, Table 10 and Table 11 provide percentile values about the monitors reporting the hijackers (Table 10) and the ASes that the hijackers were able to stealthily affect (Table 11) as we increase the percentage of IXP links in the traditional CAIDA topology by 0%, 25%, 50%, 75%, and 100% of their total values. As we can see from Table 10, omniscient hijackers were completely stealthy in all IXP simulations except for a single one, which we are investigating, where the omniscient hijacker was visible by one monitor. Table 11 clearly shows the benefits that more peer-to-peer links provide to stealthy hijackers in simulations.

REFERENCES

[1] D. Goodin. (2018). *How 3VE's BGP Hijackers Eluded the Internet-and made 29M*. [Online]. Available: https://arstechnica.com/information-technology/2018/12/how-3ves-bgp-hijackers-eluded-the-internet-and-made-29m/

[2] C. Cimpanu. (2020). *Russian Telco Hijacks Internet Traffic for Google, AWS, Cloudflare, and Others*. [Online]. Available: https://www.zdnet.com/article/russian-telco-hijacks-internet-traffic-for-google-aws-cloudflare-and-others/

[3] D. Goodin. (2018). *Russian-Controlled Telecom Hijacks Financial Services' Internet Traffic*. [Online]. Available: https://arstechnica.com/information-technology/2017/04/russian-controlled-telecom-hijacks-financial-services-internet-traffic/

[4] H. Birge-Lee, Y. Sun, A. Edmundson, J. Rexford, and P. Mittal, "Bamboozling certificate authorities with BGP," in *Proc. 27th USENIX Secur. Symp. (USENIX Security)*. Baltimore, MD, USA: USENIX Association, Aug. 2018, pp. 833–849. [Online]. Available: https://www.usenix.org/conference/usenixsecurity18/presentation/birge-lee

[5] A. Naik. (2018). *Anatomy of a BGP Hijack on Amazon's Route 53 DNS Service*. [Online]. Available: https://techbeacon.com/security/bgp-hijack-steals-aws-ip-range-cryptocurrency-theft-ensues

[6] C. Demchak and Y. Shavitt, "China's maxim–leave no access point unexploited: The hidden story of China telecom's BGP hijacking," *Mil. Cyber Affairs*, vol. 3, no. 1, p. 7, Jun. 2018.

[7] M. Smith. (2015). *Hacking Team Helped Italian Police to Hijack Internet Addresses*. [Online]. Available: https://www.engadget.com/2015-07-13-hacking-team-helped-italian-police-to-hijack-internet-addresses.html

[8] A. Siddiqui. *BGP Security in 2021*. Accessed: May 14, 2022. [Online]. Available: https://www.manrs.org/2022/02/bgp-security-in-2021/

[9] A. Mathurin. (2021). *A Regional Look into BGP Incidents in 2020*. Accessed: May 14, 2022. [Online]. Available: https://www.manrs.org/2021/03/a-regional-look-into-bgp-incidents-in-2020/

[10] T. Hlavacek, P. Jeitner, D. Mirdita, H. Shulman, and M. Waidner, "Stalloris: RPKI downgrade attack," in *Proc. 31st USENIX Secur. Symp. (USENIX Security)*, 2022, pp. 4455–4471.

[11] Y. Gilad, T. Hlavacek, A. Herzberg, M. Schapira, and H. Shulman, "Perfect is the enemy of good: Setting realistic goals for BGP security," in *Proc. 17th ACM Workshop Hot Topics Netw.*, Nov. 2018, pp. 57–63.

[12] NIST. *NIST RPKI Monitor*. Accessed: Mar. 22, 2022. [Online]. Available: https://rpki-monitor.antd.nist.gov/

[13] ThousandEyes. (2020). *ThousandEyes BGP Monitoring Service*. Accessed: May 25, 2021. [Online]. Available: https://www.thousandeyes.com/solutions/bgp-and-route-monitoring/

[14] M. Candela. (Apr. 2020). *Easy BGP Monitoring with BGPalerter*. [Online]. Available: https://labs.ripe.net/author/massimo_candela/easy-bgp-monitoring-with-bgpalerter/

[15] Cisco. *Cisco BGP Alerter*. Accessed: May 25, 2021. [Online]. Available: https://bgpstream.com/

[16] P. Sermpezis, V. Kotronis, P. Gigis, X. Dimitropoulos, D. Cicalese, A. King, and A. Dainotti, "ARTEMIS: Neutralizing BGP hijacking within a minute," *IEEE/ACM Trans. Netw.*, vol. 26, no. 6, pp. 2471–2486, Dec. 2018.

[17] X. Shi, Y. Xiang, Z. Wang, X. Yin, and J. Wu, "Detecting prefix hijackings in the internet with Argus," in *Proc. Internet Meas. Conf.*, Nov. 2012, pp. 15–28.

[18] J. Li, T. Ehrenkranz, and P. Elliott, "Buddyguard: A buddy system for fast and reliable detection of IP prefix anomalies," in *Proc. 20th IEEE Int. Conf. Netw. Protocols (ICNP)*, Oct. 2012, pp. 1–10.

[19] A. Lutu, M. Bagnulo, C. Pelsser, O. Maennel, and J. Cid-Sueiro, "The BGP visibility toolkit: Detecting anomalous internet routing behavior," *IEEE/ACM Trans. Netw.*, vol. 24, no. 2, pp. 1237–1250, Apr. 2016, doi: 10.1109/TNET.2015.2413838.

[20] N. Ripe. (2006). *Routing Information Service*. [Online]. Available: http://www.ripe.net/projects/ris/docs/peering.html

[21] Oregon RouteViews. (2013). *University of Oregon Routeviews Project*. Eugene. [Online]. Available: http://www.routeviews.org

[22] (Jul. 1, 2021). *The CAIDA AS Relationships Dataset*. [Online]. Available: http://www.caida.org/data/active/as-relationships/

[23] B. Schlinker, T. Arnold, I. Cunha, and E. Katz-Bassett, "PEERING: Virtualizing BGP at the edge for research," in *Proc. 15th Int. Conf. Emerg. Netw. Experiments Technol.*, Dec. 2019, pp. 51–67.

[24] Y. Rekhter, S. Hares, and T. Li, *A Border Gateway Protocol 4 (BGP-4)*, document RFC 4271, Jan. 2006. [Online]. Available: https://rfc-editor.org/rfc/rfc4271.txt

[25] Z. Jin, X. Shi, Y. Yang, X. Yin, Z. Wang, and J. Wu, "TopoScope: Recover AS relationships from fragmentary observations," in *Proc. ACM Internet Meas. Conf.*, Oct. 2020, pp. 266–280.

[26] P. Sermpezis. *Bias in Internet Measurement Infrastructure*. Accessed: Mar. 30, 2022. [Online]. Available: https://labs.ripe.net/author/pavlos_sermpezis/bias-in-internet-measurement-infrastructure/

[27] C. Orsini, A. King, D. Giordano, V. Giotsas, and A. Dainotti, ''BGPStream: A software framework for live and historical BGP data analysis,'' in *Proc. Internet Meas. Conf.*, Nov. 2016, pp. 429–444.

[28] H. Birge-Lee, L. Wang, J. Rexford, and P. Mittal, ''SICO: Surgical interception attacks by manipulating BGP communities,'' in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Nov. 2019, pp. 431–448.

[29] S. Goldberg, M. Schapira, P. Hummon, and J. Rexford, ''How secure are secure interdomain routing protocols,'' *ACM SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 4, pp. 87–98, Aug. 2010.

[30] M. Chiesa, G. Di Battista, T. Erlebach, and M. Patrignani, ''Computational complexity of traffic hijacking under BGP and S-BGP,'' in *Proc. ICALP*, 2012, pp. 476–487.

[31] H. Ballani, P. Francis, and X. Zhang, ''A study of prefix hijacking and interception in the internet,'' in *Proc. Conf. Appl., Technol., Archit., Protocols Comput. Commun.* New York, NY, USA: Association for Computing Machinery, Aug. 2007, pp. 265–276, doi: 10.1145/1282380.1282411.

[32] J. Scudder, R. Fernando, and S. Stuart, ''BGP monitoring protocol (BMP),'' Tech. Rep., 2016.

[33] E. Gregori, A. Improta, L. Lenzini, L. Rossi, and L. Sani, ''On the incompleteness of the AS-level graph: A novel methodology for BGP route collector placement,'' in *Proc. Internet Meas. Conf.*, Nov. 2012, pp. 253–264.

[34] G. Huston. (2021). *BGP in 2021*. [Online]. Available: https://blog.apnic.net/2022/01/06/bgp-in-2021-the-bgp-table/

[35] Cisco. (2016). *BGP Best Path Selection Algorithm*. Accessed: Mar. 30, 2022. [Online]. Available: https://www.cisco.com/c/en/us/support/docs/ip/border-gateway-protocol-bgp/13753-25.html

[36] M. Caesar and J. Rexford, ''BGP routing policies in ISP networks,'' *IEEE Netw.*, vol. 19, no. 6, pp. 5–11, Nov. 2005.

[37] G. Yossi, C. Avichai, H. Amir, S. Michael, and S. Haya, ''Are we there yet? On RPKI's deployment and security,'' NDSS, 2017.

[38] L. Gao and J. Rexford, ''Stable internet routing without global coordination,'' *IEEE/ACM Trans. Netw.*, vol. 9, no. 6, pp. 681–692, 2001.

[39] P. Sermpezis and V. Kotronis, ''Inferring catchment in Internet routing,'' *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 3, no. 2, pp. 1–31, Jun. 2019, doi: 10.1145/3341617.3326145.

[40] W. Mühlbauer, S. Uhlig, B. Fu, M. Meulle, and O. Maennel, ''In search for an appropriate granularity to model routing policies,'' *ACM SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 145–156, Oct. 2007.

[41] R. Anwar, H. Niaz, D. Choffnes, Ì. Cunha, P. Gill, and E. Katz-Bassett, ''Investigating interdomain routing policies in the wild,'' in *Proc. Internet Meas. Conf.*, Oct. 2015, pp. 71–77.

[42] P. Gill, M. Schapira, and S. Goldberg, ''A survey of interdomain routing policies,'' *Comput. Commun. Rev.*, vol. 44, no. 1, pp. 28–34, 2014.

[43] R. Singh, D. Tench, P. Gill, and A. McGregor, ''PredictRoute: A network path prediction toolkit,'' *Proc. ACM Meas. Anal. Comput. Syst.*, vol. 5, no. 2, pp. 1–24, Jun. 2021.

[44] N. Feamster, J. Winick, and J. Rexford, ''A model of BGP routing for network engineering,'' *ACM SIGMETRICS Perform. Eval. Rev.*, vol. 32, no. 1, pp. 331–342, Jun. 2004.

[45] S. Kastanakis, V. Giotsas, and N. Suri, ''Understanding the confounding factors of inter-domain routing modeling,'' in *Proc. 22nd ACM Internet Meas. Conf.*, Oct. 2022, pp. 758–759.

[46] H. Ballani, P. Francis, and X. Zhang, ''A study of prefix hijacking and interception in the Internet,'' *ACM SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 265–276, 2007.

[47] (Jul. 2021). *The CAIDA UCSD IXPs Dataset*. [Online]. Available: https://www.caida.org/data/ixps/

[48] (Sep. 2021). *CAIDA AS Rank*. [Online]. Available: https://asrank.caida.org/

[49] L. Heng. (May 2016). *Invisible Hijacking, a Case Study of Hijacking Millions of IP Address Invisibly*. [Online]. Available: https://ripe72.ripe.net/presentations/45-Invisible_Hijacking.pdf

[50] J. M. Smith, K. Birkeland, T. McDaniel, and M. Schuchard, ''Withdrawing the BGP re-routing curtain: Understanding the security impact of BGP poisoning through real-world measurements,'' in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2020, pp. 1–18.

[51] S. Vissicchio, L. Cittadini, M. Pizzonia, L. Vergantini, V. Mezzapesa, and M. L. Papagni, ''Beyond the best: Real-time non-invasive collection of BGP messages,'' in *Proc. INM/WREN*, 2010, pp. 1–6.

[52] T. Wu, J. H. Wang, J. Wang, and S. Zhuang, ''Routeinfer: Inferring interdomain paths by capturing ISP routing behavior diversity and generality,'' in *Proc. Int. Conf. Passive Act. Netw. Meas.* Cham, Switzerland: Springer, 2022, pp. 216–244.

[53] A. Milolidakis, ''Understanding the capabilities of route collectors to observe stealthy hijacks: Does adding more monitors or reporting more paths help?'' Ph.D. dissertation, Dept. Comput. Sci., Division Softw. Comput. Syst., KTH Roy. Inst. Technol., Stockholm, Sweden, 2022.

[54] F. Streibelt, F. Lichtblau, R. Beverly, A. Feldmann, C. Pelsser, G. Smaragdakis, and R. Bush, ''BGP communities: Even more worms in the routing can,'' in *Proc. ACM IMC*, Boston, MA, USA, Oct. 2018, pp. 279–292.

[55] C. McArthur and M. Guirguis, ''Stealthy IP prefix hijacking: Don't bite off more than you can chew,'' in *Proc. IEEE Global Telecommun. Conf.*, Nov. 2009, pp. 1–6.

[56] C. Testart, P. Richter, A. King, A. Dainotti, and D. Clark, ''Profiling BGP serial hijackers: Capturing persistent misbehavior in the global routing table,'' in *Proc. Internet Meas. Conf.* New York, NY, USA: Association for Computing Machinery, Oct. 2019, pp. 420–434. [Online]. Available: https://doi-org.focus.lib.kth.se/10.1145/3355369.3355581

[57] P.-A. Vervier, O. Thonnard, and M. Dacier, ''Mind your blocks: On the stealthiness of malicious BGP hijacks,'' in *Proc. Netw. Distrib. Syst. Secur. Symp.*, 2015, pp. 1–15. [Online]. Available: https://www.ndss-symposium.org/ndss2015/ndss-2015-programme/mind-your-blocks-stealthiness-malicious-bgp-hijacks/

[58] X. Hu and Z. M. Mao, ''Accurate real-time identification of IP prefix hijacking,'' in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2007, pp. 3–17.

[59] Z. Zhang, Y. Zhang, Y. C. Hu, Z. M. Mao, and R. Bush, ''ISPY: Detecting IP prefix hijacking on my own,'' *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 327–338, 2008.

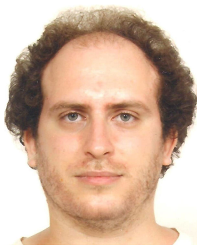
[60] C. Zheng, L. Ji, D. Pei, J. Wang, and P. Francis, ''A light-weight distributed scheme for detecting Ip prefix hijacks in real-time,'' *ACM SIGCOMM Comput. Commun. Rev.*, vol. 37, no. 4, pp. 277–288, 2007.

[61] K. Balu, M. L. Pardal, and M. Correia, ''DARSHANA: Detecting route hijacking for communication confidentiality,'' in *Proc. IEEE 15th Int. Symp. Netw. Comput. Appl. (NCA)*, Oct. 2016, pp. 52–59.

[62] R. Bush and R. Austein, *The Resource Public Key Infrastructure (RPKI) to Router Protocol*, Internet Request for Comments, document RFC, 6810, Jan. 2013. [Online]. Available: http://www.rfc-editor.org/rfc/rfc6810.txt

[63] T. Alfroy, T. Holterbach, and C. Pelsser, ''MVP: Measuring internet routing from the most valuable points,'' in *Proc. 22nd ACM Internet Meas. Conf.*, Oct. 2022, pp. 770–771.

[64] A. Milolidakis. (2022). *Nanog: Announcement of Experiments*. [Online]. Available: https://seclists.org/nanog/2022/May/2

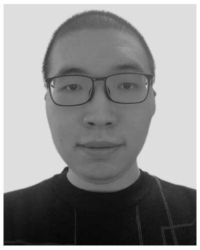


ALEXANDROS MILOLIDAKIS received the B.Sc. degree in computer science and the M.Sc. degree in computer networks and telecommunications from the Computer Science Department, University of Crete, in 2015 and 2018, respectively. He is currently pursuing the Ph.D. degree with the KTH Royal Institute of Technology (KTH), Sweden.

From 2016 to 2018, he was a Research Assistant with the Telecommunications and Networks Laboratory (TNL), Institute of Computer Science (ICS), The Foundation for Research and Technology—Hellas (FORTH). In 2019, he joined the Prof. Dejan Kostic's Communication Systems Group, KTH. His research interests include monitoring network infrastructures, traffic analysis, identification of network anomalies and attacks, and modeling internet-wide accurate route propagation.

TOBIAS BÜHLER received the B.Sc. and M.Sc. degrees in electrical engineering and information technology from ETH Zürich, Switzerland, in 2013 and 2015, respectively, where he is currently pursuing the Ph.D. degree.

He successfully defended his Ph.D. thesis, in March 2023. Then, he joined the Prof. Laurent Vanbever's Networked Systems Group. His research interests include network monitoring, the analysis of internet path signals and new ways to generate representative network traffic.



KUNYU WANG received the B.S. degree in communication engineering from Beijing Jiaotong University, Beijing, China, in 2021. He is currently pursuing the M.S. degree in communication systems with the KTH Royal Institute of Technology, Stockholm, Sweden.

In 2021, he was a Student Assistant with the KTH Royal Institute of Technology. His research interests include network routing protocols, algorithms for congestion control, and routing algorithms in practice.

MARCO CHIESA received the Ph.D. degree in computer engineering from Roma Tre University, in 2014. He is currently an Associate Professor with the KTH Royal Institute of Technology, Sweden. His research interests include network architectures, systems, and protocols, including aspects of design, optimization, security, and privacy. He received the 2022 Usenix NSDI Community Award, the 2020 IEEE William R. Bennett Prize, the 2013 IEEE ICNP Best Paper, the 2012 IETF Applied Network Research Prize, and a three-time distinguished reviewer at IEEE Infocom. He was the TPC Co-Chair of ACM CoNEXT'21. His research has been funded by the Swedish Research Council and KTH Digital Futures.

LAURENT VANBEVER received the Ph.D. degree in computer science from the University of Louvain, in October 2012, under the guidance of Olivier Bonaventure.

His Ph.D. dissertation was titled, "Methods and Techniques for Disruption-Free Network Reconfiguration." After his Ph.D., he spent two years with Princeton University working with Jennifer Rexford as a Postdoctoral Researcher. He is currently an Associate Professor with ETH Zürich, where he also leads the Networked Systems Group (NSG). His research interests include the crossroads of theory and practice, with a focus on network programmability and internet routing. Overall, he aims at making (large) computer networks more performant and easier to manage.

STEFANO VISSICCHIO received the master's and Ph.D. degrees in computer science from Roma Tre University, in 2008 and April 2012, respectively. He is currently a Lecturer with University College London. Before joining University College London, he was a Postdoctoral Researcher with Université catholique of Louvain. His research interests include network management, routing theory and protocols, measurements, and network programmability.