Computer Networks 70 (2014) 260–287

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

How secure are secure interdomain routing protocols?

Sharon Goldberg [a,*], Michael Schapira [b], Pete Hummon [c], Jennifer Rexford [d]

[a] Computer Science, Boston University, Boston, MA 02215, USA
[b] Hebrew University, Jerusalem, Israel
[c] AT&T, NJ, USA
[d] Princeton University, Princeton, NJ, USA

ARTICLE INFO

Article history:
Received 8 September 2013
Received in revised form 8 March 2014
Accepted 12 May 2014
Available online 13 June 2014

Keywords:
Security
Interdomain routing
BGP

ABSTRACT

In response to high-profile Internet outages, BGP security variants have been proposed to prevent the propagation of bogus routing information. The objective of this paper is to inform discussions of which variant should be deployed in the Internet. To do this, we *quantify* the ability of the key protocols (origin authentication, soBGP, S-BGP, and data-plane verification) to limit the impact of *traffic-attraction attacks*; *i.e.*, when an attacker deliberately draws traffic to its own network, in order to drop, tamper, or eavesdrop on packets. Our results and contributions are as follows:

(1) One might expect that an attacker could maximize the volume of traffic it attracts by using the following intuitive strategy: the attacker should announce, to as many of its neighbors as possible, the *shortest* path that is not flagged as bogus by the secure protocol. Through simulations on an empirically-determined AS-level topology, we show that this strategy is surprisingly effective, even when an advanced security solution like S-BGP or data-plane verification is fully deployed.
(2) Next, we show that these results *underestimate* the severity of attacks. In fact, counterintuitive strategies, like announcing longer paths, announcing to fewer neighbors, or triggering BGP loop-detection, can be used to attract even more traffic than the strategy above. We illustrate this using counterintuitive examples. We also demonstrate that these attacks are not merely hypothetical, by searching the empirical AS-level topology and identifying specific ASes that can launch these attacks.
(3) We prove that it is NP hard to find a traffic-attraction attack strategy that attracts the maximum volume of traffic.

Our results suggest that a clever export policy (*i.e.*, where the attacker announces a legitimate path to a carefully chosen set of neighbors) an often attract almost as much traffic as a bogus path announcement. Thus, our work implies that mechanisms that police export policies (*e.g.*, prefix filtering) are crucial, even if more advanced cryptographic solutions like S-BGP are fully deployed.

© 2014 Elsevier B.V. All rights reserved.

1. Introduction

The Internet is notoriously vulnerable to *traffic attraction* attacks, where Autonomous Systems (ASes) manipulate BGP to attract traffic to, or through, their networks [3,5,9,10,21,40,44–46]. Attracting extra traffic

* Corresponding author. Tel.: +1 617 353 8919.
E-mail addresses: goldbe@cs.bu.edu (S. Goldberg), schapiram@huji.ac.il (M. Schapira), jrex@cs.princeton.edu (J. Rexford).

http://dx.doi.org/10.1016/j.comnet.2014.05.007
1389-1286/© 2014 Elsevier B.V. All rights reserved.

enables the AS to increase revenue from customers, or drop, tamper, or snoop on packets. While the proposed extensions to BGP prevent many attacks (see [6] for a survey), even these secure protocols are susceptible to a strategic manipulator who deliberately exploits their weaknesses to attract traffic to its network. Given the difficulty of upgrading the Internet to a new secure routing protocol, it is crucial to understand how well these protocols blunt the impact of traffic attraction attacks.

1.1. Quantifying the impact of attacks

We evaluate the four major security extensions that allow ASes to validate paths learned via BGP, ordered from weakest to strongest: origin authentication [39,41], soBGP [49], Secure BGP (S-BGP) [32], and data-plane verification [6,50]. We also evaluate an orthogonal security mechanism: prefix filtering [6]. While the stronger protocols prevent a strictly larger set of attacks than the weaker ones, these security gains often come with significant implementation and deployment costs. To inform discussions about which of these secure protocols should be deployed, we would like to quantitatively compare their ability to limit traffic attraction attacks. Thus, we simulate attacks on each protocol on an empirically-measured AS-level topology [1,8,12], and determine the percentage of ASes that forward traffic to the manipulator.

Performing a quantitative comparison requires some care. It does *not* suffice to say that one protocol, say S-BGP, is four times as effective as another protocol, say origin authentication, at preventing a specific type of attack strategy; there may be other attack strategies for which the quantitative gap between the two protocols is significantly smaller. Since these more clever attack strategies can just as easily occur in the wild, our comparison must be in terms of the worst possible attack that the manipulator could launch on each protocol. To do this, we put ourselves in the mind of the manipulator, and look for the optimal strategy he can use to attract traffic from as many ASes as possible.

However, before we can even begin thinking about optimal strategies for traffic attraction, we first need a model for the way traffic flows in the Internet. In practice, this depends on local routing policies used by each AS, which are not publicly known. However, the BGP decision process breaks ties by selecting shorter routes over longer ones, and it is widely believed [18,27] that policies depend heavily on economic considerations. Thus, conventional wisdom and prior work [15,17,27–29] suggests basing routing policies on business relationships and AS-path lengths. While this model (used in many other studies, *e.g.*, [3,19,30]) does *not* capture all the intricacies of interdomain routing, it is still very useful for gaining insight into traffic attraction attacks. All of our results are obtained within this model.

1.2. Thinking like a manipulator

If routing policies are based on AS path lengths, then intuition suggests that it is optimal for the manipulator to use the following "smart" attack strategy: announce the shortest path that the protocol does not reject as bogus, to as many neighbors as possible. Depending on the security protocol, this means announcing: (a) a direct connection to the victim IP prefix (*i.e.*, a "prefix hijack" as in [9,40]), or (b) a bogus edge to the legitimate destination AS, or (c) a short path that exists but was never advertised, or (d) a short path that the manipulator learned but is not using, or (f) a legitimate path that deviates from normal export policy (*i.e.*, a "route leak" as in [44]). Indeed, we use simulations on a measured AS-level topology to show that this "smart" attack strategy is quite effective, even against advanced secure routing protocols like S-BGP and data-plane verification.

Worse yet, we use counterexamples show that our simulations *underestimate* the amount of damage manipulator could cause, because the "smart" attack is not optimal. In fact, the following bizarre strategies can sometimes attract even more traffic than the "smart" attack: announcing a longer path, exporting a route to fewer neighbors, or using "path poisoning" to trigger BGP's loop-detection mechanism (*cf.*, [31]). In fact, we present counterexamples that show that prefix hijacking (*i.e.*, originating a prefix you do not own) is *not* always the most effective attack against BGP! These counterexamples are not merely hypothetical—we identify specific ASes in the measured AS-level topology that could launch them. Moreover, we prove that it is NP-hard to find the manipulator's optimal attack, suggesting that a comprehensive comparison across protocols must remain elusive.

1.3. Our findings and recommendations

While we necessarily underestimate the amount of damage a manipulator could cause, we can make a number of concrete statements. Our main finding is that secure routing protocols only deal with one half of the problem: while they do restrict the paths the manipulator can announce, they fail to restrict his export policies. Thus, our simulations show that, when compared to BGP and origin authentication, soBGP and S-BGP significantly limit the manipulator's ability to attract traffic by announcing bogus short paths to all its neighbors. However, even in a network with S-BGP or data-plane verification, we found that a manipulator can still attract traffic by cleverly manipulating his export policies. Indeed, we found that announcing a short path can be less important than exporting that path to the right set of neighbors (an attack strategy that has also been called a "route leak" [11,44]). Thus:

- Advanced security protocols like S-BGP and data-plane verification do *not* significantly outperform soBGP for the "smart" attacks we evaluated.
- Prefix filtering of paths exported by stub ASes (*i.e.*, ASes with no customers) provides a level of protection that is at least comparable to that provided soBGP, S-BGP and data-plane verification.
- Tier 2 ASes are in the position to attract the largest volumes of traffic, even in the presence of data-plane verification and prefix filtering (of stubs).
- *Interception attacks* [3,9,45]—where the manipulator silently intercepts traffic and delivers it to the destination—are easy for many ASes, especially large ones.

We could quibble about whether or not manipulating export policies even constitutes an *attack*; after all, each AS has the right to decide to whom it announces paths. However, our results indicate that a clever export policy can attract almost as much traffic as a bogus path announcement. Indeed, Section 6.1 presents an example where an AS in the measured topology gains almost as much exporting a provider-learned path to another provider, as he would by a prefix hijack (announcing that he owns the IP prefix). Thus, our results suggest that addressing traffic attraction attacks requires both mechanisms that prevent bogus path announcements (*e.g.*, soBGP or S-BGP) as well as mechanisms that police export policies (*e.g.*, prefix filtering).

1.4. Roadmap

We start by presenting the routing model, threat model, and experimental setup (Section 2), and move on to describing the vulnerabilities of different secure routing protocols and how a manipulator can exploit them (Section 3). We then describe and evaluate the "smart" *attraction* attacks (Section 4), and then use both theory and simulation to analyze *interception* attacks (Section 5). We then present counterexamples, found in measured AS graph, that prove that the "smart" attacks are not optimal (Section 6), and show that finding the optimal attack strategy is NP hard (Section 7). We conclude by discussing related work (Section 8), and the effect of our modeling assumptions on our results (Section 9).

Appendices. This is the extended version of a work that appeared in SIGCOMM'10 [21] and therefore contains a variety of supplementary information. Appendices A, B and C discuss issues related to our experimental methodology. Appendix D presents a supplementary example that shows how an traffic interception attack can fail, and points out an error in [3]. Proofs of our theorems are in Appendices E and F. Finally, all the results presented in the body of this paper are based on the CAIDA AS graph from November 20, 2009 [12]; thus, to highlight the robustness of these results, Appendix G presents results computed on a different AS-level graph [1,8]. Section 8 has bibliographical notes on the relationship between this paper and its conference version [21].

2. Model and methodology

We first present a model of interdomain routing and routing policies, based on the standard models in [16,17,25,28,29], followed by our threat model for traffic attraction, and finally our experimental setup.

2.1. Modeling interdomain routing

The AS graph. The interdomain-routing system is modeled with a labeled graph called an *AS graph*, as in Fig. 1. Each AS is modeled as a single node and denoted by its AS number. Edges represent direct physical communication links between ASes. Adjacent ASes are called *neighbors*. Since changes in topology typically occur on a much longer timescale than the execution of the protocol, we follow [25] and assume the AS-graph topology is static. BGP computes paths to each destination IP prefix separately, so we assume that there is a unique destination IP prefix to which all other nodes attempt to establish a path. As shown in Fig. 1, we assume that a single AS v is authorized to announce the destination IP prefix under consideration; we say that v is authorized to *originate* the IP prefix.

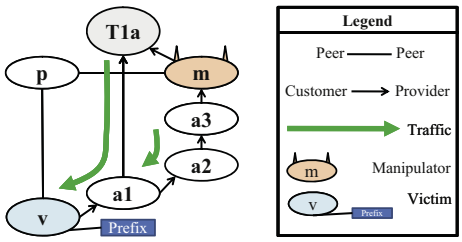


Fig. 1. Anonymized subgraph of CAIDA's AS graph.

Establishing paths. In BGP, an AS first chooses an outgoing edge on which it forwards traffic based on a local ranking on outgoing paths, and then announces this path to some subset of its neighbors. To model this, we assume that each node n has a set of *routing policies*, consisting of (a) a *ranking* on outgoing paths from n to the destination d, and (b) a set of *export policies*, a mapping of each path P to the set of neighbors to which n is willing to announce the path P. We say that node n has an *available path aPd* if n's neighbor a announced the path "*aPd*" to n. If an available path *aPd* is ranked higher than the outgoing path that node n is currently using, then an *normal* node n will (a) forward traffic to node a, and (b) announce the path *naPd* to all his neighbors as specified by his export policies.

Business relationships. We suppose the AS graph is annotated with the standard model for business relationships [17,28,29]; while more complicated business relationships exist in practice, the following is widely believed to capture the majority of the economic relationships in the Internet. As shown in Fig. 1, there are two kinds of edges: *customer-provider* (where the customer pays the provider for connectivity, represented with an arrow from customer to provider), and *peer-to-peer* (where two ASes owned by different organizations agree to transit each other's traffic at no cost, represented with an undirected edge). Because some of our results are based on CAIDA's AS graph [12], we also consider *sibling-to-sibling* edges. Details about our treatment of siblings is in Appendix A. Finally, our theoretical results sometimes use [17]'s assumption that an AS cannot be its own indirect customer:

GR1 The graph has no customer-provider cycles.

2.2. Modeling routing policies

In practice, the local routing policies used by each AS in the Internet are arbitrary and not publicly known. However, because we want to understand how false routing information propagates through the Internet, we need to concretely model routing policies. Since it is widely

believed that business relationships play a large role in determining the routing policies of a given AS [17,27], and we have reasonably accurate empirical maps of the business relationships between ASes [1,8,12], we base our model on these relationships.

Rankings. BGP is first and foremost designed to prevent loops. Thus, we assume that node a rejects an announcement from its neighbor b if it contains a *loop*, *i.e.*, if node a appears on the path that node b announces. Beyond that, we can think of the process ASes use to select routes as follows; first applying local preferences, then choosing shortest AS paths, and finally applying a tie break. Since the local preferences of each AS are unknown, and are widely believed to be based (mostly) on business relationships, we model the three-step process as follows:

LP **Local preference.** Prefer outgoing paths where the next hop is a customer over outgoing paths where the next hop is a peer over paths where the next hop is a provider.
SP **Shortest paths.** Among the paths with the highest local preference, chose the shortest ones.
TB **Tie break.** Among these, choose the path whose next hop has the lowest AS number.[1]

Our model of local preferences is based on Gao-Rexford condition **GR3**, and captures the idea that an AS has an economic incentive to prefer forwarding traffic via customer (that pays him) over a peer (where no money is exchanged) over a provider (that he must pay). Notice that this implies that an AS can sometimes prefer a longer path! (*e.g.*, in Fig. 1, AS m prefers the five-hop customer path through $a3$ over the four-hop provider path through Tier 1 $T1$.)

Export policies. Our model of export policies is based on the Gao-Rexford condition **GR2**:

GR2 AS b will only announce a path via AS c to AS a if at least one of a and c are customers of b.

GR2 captures the idea that an AS should only be willing to load his own network with transit traffic if he gets paid to do so. However, because **GR2** does *not* fully specify the export policies of every AS (for instance, an AS could decide to export paths to only a subset of his customers), it does not suffice for our purposes. Thus, we model normal export policies as follows:

NE An AS will announce all paths to all neighbors except when **GR2** forbids him to do so.

2.3. Threat model

One strategic manipulator. We assume that all ASes in the AS graph behave *normally*, *i.e.*, according to the policies in 2.1-2.2, except for a *single manipulator* (*e.g.*, AS m in Fig. 1). We leave models dealing with colluding ASes for future work.

Normal ASes and normal paths. We assume that every *normal* AS uses the routing policies in Section 2.2; thus, the *normal path* is the path an AS (even the manipulator) would choose if he used the normal rankings of Section 2.2, and *normal export* is defined analogously. (*e.g.*, In Fig. 1, the manipulator m's normal path is through his customer AS $a3$.) We shall assume that every normal AS knows its business relationship with his neighbors, and also knows the next hop it chooses for forwarding traffic to a given destination. In order to evaluate the effectiveness of each secure routing protocol, we assume that ASes believe everything they hear, except when the secure routing protocol tells them otherwise. As such, we do not assume that ASes use auxiliary information to detect attacks, including knowledge of the network topology or business relationships between distant ASes, *etc.*, unless the secure routing protocol specifically provides this information.

Attraction *vs.* interception attacks. In an *attraction attack*, the manipulator's goal is to attract traffic, *i.e.*, to convince the maximum number of ASes in the graph to forward traffic that is destined to the victim IP prefix via the manipulator's own network. To model the idea that a manipulator may want to eavesdrop or tamper with traffic before forwarding it on to the legitimate destination, we also consider *interception attacks*. In an interception attack, the manipulator has the additional goal of ensuring that he has an available path to the victim. This is in contrast to an attraction attack, where the manipulator is allowed, but not required, to create a blackhole where he has no working path to the victim IP prefix (*e.g.*, Fig. 12). Refs. [5,40] are examples of attraction attacks, while [9,45] are examples of interception attacks.

The fraction of attracted ASes. In this paper, we measure the success of an attack strategy by counting the fraction of ASes in the internetwork from which that manipulator attracts traffic; this amounts to assuming that every AS in the internetwork is of equal importance to the manipulator.[2] However, it is well known that the distribution of traffic in the Internet is *not* uniform across the ASes; to address this, we also report the fraction of ASes of various sizes from which the manipulator attracts traffic, where we measure size by the number of direct customers the AS has.

Attack strategies. To capture the idea that the manipulator is strategic, we allow him to be more clever than the normal ASes; specifically, we allow him to use knowledge of the global AS graph and its business relationships in order to launch his attacks. (However, most of the strategies we considered require only knowledge that is locally available at each AS.) An attack strategy is a set of routing announcements and forwarding choices that deviates from the normal routing policies specified in

[1] We need a consistent way to break ties. In practice, this is done using the intradomain distance between routers and router IDs. Since our model does not incorporate geographic distance or individual routers, we use AS number instead.

[2] While a manipulator may want to attract traffic from a specific subset of ASes, we do not analyze this, because we lack empirical data to quantify that subset of ASes that a given manipulator may want to attract.

Section 2.2. An attack strategy may include, but is not limited to:

- Announcing an unavailable or non-existent path.
- Announcing different paths to different neighbors.
- Announcing a legitimate available path that is different from the normal path.
- Exporting a path (even the legitimate normal path) to a neighbor to which no path should be announced to according to the normal export policies.

Indeed, one might argue that some of these strategies do not constitute 'dishonest behavior'. However, it is important to consider these strategies in our study, since we shall find that they can sometimes be used to attract as much traffic as the traditional 'dishonest' strategies (*e.g.*, announcing non-existent paths).

Scope of this paper. This paper focuses on traffic attraction attacks when a secure routing protocol is "fully deployed", *i.e.*, deployed by every AS in the internetwork; we do not consider other routing security issues, for instance, mismatches between the control- and data-plane [21,50], or traffic deflection attacks, where a manipulator wants to divert traffic from himself or some distant, innocent AS [6]. See Section 8 for more discussion.

2.4. Experiments on empirical AS graphs

All our results and examples are based on measured AS-level Internet topologies, annotated with business relationships.

Algorithmic simulations. At the core of our experiments is a algorithm that takes in an AS graph and outputs the paths that each AS uses to reach the destination prefix, under the assumption that each AS 'normally' uses the routing policies of Section 2.2. We also use our algorithm, which is based on breadth-first search and described in [20], to simulate the result of a manipulator's attack strategy; see Appendix B for details.

Average case analysis. Since the influence of an attack strategy depends heavily on the locations of the manipulator and the victim in the AS graph, we run simulations across many (manipulator, victim) pairs. Rather than reporting average results, we plot the *distribution* of the fraction of ASes that direct traffic to the manipulator. While a manipulator would certainly not select its victim at random, reporting distributions allows us to measure the extent to which a secure protocol can blunt the power of the manipulator, determine the fraction of victims that a manipulator could effectively target, and identify positions in the network that are effective launching points for attacks. Our experiments are run on randomly-chosen (manipulator, victim) pairs. We found that 60 K experiments of each type were sufficient for our results to stabilize.

Multiple AS graphs. Because the actual AS-level topology of the Internet remains unknown, and inferring AS relationships is an active area of research, we run simulations on a number of different datasets: multiple years of CAIDA data [12], and Cyclops data [8] augmented with 21,000 peer-to-peer edges from [1]'s IXP dataset. Even though these datasets use different relationship-inference algorithms, the trends we observed across datasets were remarkably consistent. Thus, all the results we present are from CAIDA's November 20, 2009 dataset (with slight modifications to the sibling relationships, see Appendix A.2); counterparts of these graphs, computed from Cyclops and IXP data [1,8] are in Appendix G.

Realistic examples. Rather than providing contrived counterexamples, we give evidence that the attack strategies we discuss could succeed in wild by ensuring that every example we present comes from real data. To find these examples, we (algorithmically) searched the measured AS graph for specific subgraphs that could induce specific counterexamples, and then simulated the attack strategy. All the examples we present here were found in CAIDA's November 20, 2009 dataset [12], and then anonymized by replacing AS numbers with symbols (*e.g.*, in Fig. 1, m for manipulator, v for victim, $T1$ for a Tier 1 AS, *etc.*).

3. Circumventing BGP security protocols

This section overviews the security protocols we consider. Each one of these protocols protects against a specific set of attack strategies; in this section, we present the set of attack strategies that succeed against each security protocol, *i.e.*, the set of (possibly) bogus paths that a manipulator m can announce to each neighbor without getting caught. We demonstrate these strategies using an anonymized subgraph of CADIA's AS graph in Fig. 1. We use simulations to demonstrate the fraction of ASes that are fooled into sending traffic to the manipulator in Fig. 1 with each of these strategies.

Our focus is on protocols with well-defined security guarantees. Thus, we consider the five major BGP security variants, ordered from weakest to strongest security, as follows: *(unmodified) BGP, Origin Authentication, soBGP*, *S-BGP*, and *data-plane verification*. Because we focus on security guarantees and not protocol implementation, we use these as an umbrella for many other proposals (see [6] for a survey) that provide similar guarantees using alternate, often lower-cost, implementations. Furthermore, our ordering of protocols is strict: an attack that succeeds against a strong security protocol, will also succeed against the weaker security protocol. We also consider *prefix filtering* as an orthogonal security mechanism.

BGP. BGP does not include mechanisms for validating information in routing announcements. Moreover, BGP has no restriction on the set of export policies that an AS can use. Thus the following set of attack strategies succeeds against BGP: the attacker announces, to each of its neighbors, any path it like (or not path at all). We mention a few important instances of this attack strategy:

The most important of these attack strategies, which has been seen in the wild on numerous occasions [9,10,40,46], is called a *prefix hijack*. In a prefix hijack, the hijacking AS (falsely) claiming that he is the owner of the victim's IP prefix. For example, suppose manipulator m in Fig. 1 (an anonymized Canadian Tier 2 ISP) launches this attack on the v's IP prefix (an anonymized Austrian AS),

by announcing the path (*m*, Prefix) to its neighbors. Our simulations show that he attracts traffic from 75% of the ASes in the internetwork.[3]

Other attack strategies, including *path shortening attacks* also succeed against BGP; in a path shortening attack, an attacker exports a shortened version of a path that it learned from its neighbors. For example, suppose manipulator *m* in Fig. 1 learns the path ($a3, a2, a1, v$, Prefix). The manipulator *m* can announce the shorter path ($a3, v$, Prefix) to its neighbors *p* and *T*1*a*, even though no such path exists in the network.

Another class of attack strategies that succeed against BGP are known as *route leaks* [11]. In a route leak, a manipulator violates its normal export policies (in our model, **GR2**) by announcing a legitimate route to a larger set of neighbors than normal. For example, suppose *p* in Fig. 1 was a manipulator; in a route leak, *p* would announce the path (*p*, *v*, Prefix) to neighbor *m*, even though doing this is a violation of normal export policies of *p* per **GR2** (since neither *m* nor *v* are customers of *p*).

Origin authentication. Origin authentication [39,41] uses a trusted database to create a binding between prefixes and ASes are authorized to originate them. (For example, the database would having a binding from prefix "Prefix" to AS *v* in Fig. 1.) Origin authentication (also known as "prefix validation" [41]) is gradually being rolled out on the Internet, using the RPKI [35] as the trusted database. The RPKI is a database that stores cryptographic public keys for ASes and routers (which may eventually be used in future deployments of soBGP, S-BGP, or BGPSEC, see the subsequent discussion), and Route Origin Authorizations (ROAs) that bind IP prefixes to the ASes that are authorized to originate them in BGP [35] (which are used for origin authentication). Any BGP message with a (prefix, origin AS)-pair that does not have a corresponding binding in the RPKI, is ignored by all ASes in the internetwork (see Table 1).

Origin Authentication therefore guarantees that an AS cannot falsely claim to be the rightful origin or an IP prefix. Origin Authentication therefore prevents the prefix- and subprefix hijack attack strategies that succeeded on BGP; this follows because the trusted database will not contain a binding between the hijacked prefix and the hijacking AS. (For example, *m* can no longer hijack the prefix "Prefix" in Fig. 1, because the trusted database does not contain a binding from "Prefix" to *m.)*

The set of attack strategies that does succeed against Origin Authentication is as follows: the attacker announces, to each of its neighbors, any path it like (or not path at all), as long at that path *ends* at the AS that rightfully originates the victim IP prefix. For instance, in Fig. 1, the manipulator *m* can attract traffic from 25% of the ASes in the internetwork by announcing the path (m, v, Prefix) to each of his neighbors, since *v* is the legitimate origin for the prefix; this path is bogus, however, because no link exists between *m* and *v*. Route leaks also succeed against origin authentication, as well as any other path-shortening attack where the last hop on the path is *v*, the rightful origin of the prefix.

Table 1
Summary of attacks presented in this paper, and their ability to circumvent different secure routing protocol variants. Prefix filtering can be used in combination with any secure routing protocol; when this is done, the attacks shown may only be realized by manipulators that are *not* stub ASes.

	BGP	OrAuth	soBGP	S-BGP
Prefix hijack	✓	X	X	X
direct link to legitimate origin	✓	✓	X	X
Existing (but unavailable) path	✓	✓	✓	X
Route leak	✓	✓	✓	✓
Longer path (Section 6.1)	✓	✓	✓	✓
Export less (Section 6.2)	✓	✓	✓	✓
Game loop detection (Section 6.3)	✓	✓	✓	X

soBGP. Secure Origin BGP (soBGP) [49] augments origin authentication with an additional trusted database that guarantees that any announced path *exists* in the AS-level topology of the internetwork. To realize soBGP, the cryptographic keys (certified by the RPKI) could be used by neighboring ASes *a*1 and *a*2 to jointly sign a statement certified the existence of a physical link between them. Any BGP message where the AS path contains an edge, or a (prefix, origin AS)-pair that is not certified by the trusted database is ignored by all ASes in the internetwork. soBGP therefore prevents the class of path-shortening attacks where a manipulator announces a path that does not exist in the network, thus preventing some of the attacks that succeeded against Origin Authentication: if *m* announced the path (m, v, Prefix) in Fig. 1, soBGP would cause that path to be discarded, because no link exists between *m* and *v*.

The set of attack strategies that does succeed against soBGP is therefore as follows: the attacker announces, to each of its neighbors, any path it like (or not path at all), as long at that path *exists* in the internetwork. Thus, a manipulator can still announce a path the *exists* but is not actually *available*; for example, in Fig. 1, the manipulator *m* can attract traffic from 10% of the ASes in the internetwork by announcing the path (m, p, v, Prefix). Notice that this path is unavailable; **GR2** forbids the Swiss Tier 2 ISP *p* to announce a peer path to another peer, so *m* would not actually have learned this path from *p*. We note that this class of attacks requires the manipulator to find paths that actually exist, which requires knowledge of the global topology of the network. However, obtaining this information is not especially difficult; an industrious manipulator could obtain this information from AS graph datasets [1,8,12], or even from the soBGP database itself!

S-BGP/BGPSEC. Secure BGP [32] and BGPSEC [34] also augments origin authentication. In addition to a trusted database binding prefixes to ASes that are authorized to originate them (*i.e.*, origin authentication), S-BGP also augments BGP routing announcements with cryptographically-signatures, to provide a property called *path verification*. Path verification guarantees that every AS *a* can only announce a path *abP* to its neighbors if it has a neighbor *b* that announced the path *bP* to *a*.

[3] In fact, an even more damaging strategy, called a *subprefix hijack*, is available to manipulator; by announcing a longer, more specific subprefix of the victim's IP prefix, he can attract traffic from 100% of the ASes in the internetwork. The most famous instance of this attack is Pakistan Telecom's hijack of YouTube's traffic in 2008 [5]. This work does not discuss subprefix hijacks in detail, because the fraction of traffic that the attacker can attract is well understood (*i.e.*, 100% of traffic, in the absence of prefix filtering).

S-BGP therefore limits a manipulator to announcing *available paths*, and prevents some of the attacks that succeeded on soBGP: if m announced the path $(m, p, v$, Prefix), the path would be discarded because it is unavailable (since **GR2** prevents p from announcing the path $(p, v$, Prefix) to m).

However, there is still a non-empty set of attack strategies that succeeds against S-BGP: the attacker can announce, to each of its neighbors, any path it like (or not path at all), as long as it is *available*. Route leaks, for example, fall within this class of attacks. A manipulator could also announce an available path that is different from the normal path is uses for forwarding traffic. For instance, in Fig. 1, the manipulator's *normal path* (see Section 2.3) is the five-hop customer path $(m, a3, a2, a1, v$, Prefix); announcing that path allows him to attract traffic from 0.9% of the ASes in the internetwork. However, with S-BGP the manipulator could instead announce the *shorter* four-hop provider path $(m, T1, a1, v$, Prefix), thus doubling the fraction of ASes attracted to 1.7%. Thus, the manipulator can announce the shorter, more expensive, provider path, while actually forwarding traffic on the cheaper, longer customer path.

Data-plane verification. Data-plane verification [6,42,50] prevents an AS from announcing one path, while forwarding on another. Thus, if the manipulator in Fig. 1 wants to maximize his attracted traffic by announcing the shorter, most expensive provider path $(m, T1, a1, v$, Prefix), he must also forward traffic on that path. Since we do not model the data plane in this paper, for our purposes, S-BGP, BGPSEC and data-plane verification are all treated in a similar manner.

Prefix filtering. For the purpose of this paper, we suppose that prefix filtering polices the BGP announcements made by *stub ASes*. A stub is an AS with no customers. Because stubs are consumers (rather than providers) of Internet service, they only carry ingress traffic that is destined to their own prefixes; this is implicit in the model as well, since **GR2** implies that a stub should *never* announce a path to a prefix it does not own. Thus, we suppose that prefix filtering has each provider keep a "prefix list" of the IP prefixes owned by its direct customers that are stubs. If a stub announces a path to *any* IP prefix that it does not own, the provider drops/ignores the announcement, thus enforcing **GR2**. In most of our analysis, we assume that every provider in the internetwork correctly implements prefix filtering. (The implications of partially-deployed prefix filtering are in Section 4.11.)

Thus, we suppose that prefix filtering completely prevents all attack strategies that are launched by stub ASes. (For example, route leaks by stubs, which succeed against BGP, Origin Authentication, soBGP, and S-BGP, are completely eliminated.) Meanwhile, if the manipulator is not a stub AS, prefix filtering does not impact its ability to launch attacks. Thus, we will often consider prefix filtering in combination with other routing security variants. For example, when prefix filtering is used with S-BGP, non-stub manipulators can launch any of the attacks that succeeded against S-BGP, while stub manipulators cannot launch any attacks at all.

4. Smart attraction attacks

We simulate attraction attacks on measured graphs of the Internet's AS-level topology [1,8,12] to determine how many ASes the manipulator can attract in the *average case*. This section first presents the attack strategies we simulated, and then reports our results.

4.1. A smart-but-suboptimal attack strategy

We assumed that ASes make routing decisions based on business relationships and path length, and that a manipulator m cannot lie to his neighbor a about their business relationship (*i.e.*, between m and a). Thus, intuition suggests that the manipulator's best strategy is to widely announce the shortest possible path.

"Shortest-Path Export-All" attack strategy. Announce to *every* neighbor, the *shortest* possible path that is *not flagged as bogus by the secure routing protocol*.

An "Shortest-Path Export-All" attack strategy on BGP is a prefix hijack; on origin authentication it is when the manipulator announces that he is directly connected to the legitimate origin AS; on soBGP it is when the manipulator announces the shortest path that *exists* in the graph, from itself to the legitimate origin AS; on S-BGP it is when the manipulator announces the shortest *available* path in the graph, from itself to the legitimate origin AS. Every "Shortest-Path Export-All" attack strategy on S-BGP/BGPSEC is also an attack on data-plane verification. The "Shortest-Path Export-All" attack strategy on S-BGP/BGPSEC has the manipulator announce his shortest *legitimate available path* to the victim, instead of his *normal path* (see Sections 2.3 and 3). Notice that if the manipulator actually decides to *forward* his traffic over the announced path, he has a successful attack on data-plane verification as well. Thus, the "Shortest-Path Export-All" attack strategy on data-plane verification is *identical* to the attack on S-BGP. (To reduce clutter, the following mostly refers to the attack on S-BGP.) Finally, when prefix filtering is in place, we suppose a stubs cannot launch a "Shortest-Path Export-All" attack strategy.

We underestimate damage. Section 6 shows that the "Shortest-Path Export-All" attack strategy is *not* actually optimal for the manipulator, and Section 7 shows that finding the optimal attack strategy is NP-hard. Thus, we give up on finding the *optimal* attack strategy, and run simulations assuming that the manipulator uses this smart-but-suboptimal attack. This means that the results reported in this section *underestimate* the amount of damage a manipulator could cause, and we usually *cannot* use these results to directly compare different secure routing protocols. In spite of this, our simulations do provide both (a) useful lower bounds on the amount of damage a manipulator could cause, and (b) a number of surprising insights on the strategies a manipulator can use to attract traffic to his network.

4.2. Prefix filtering is crucial

Our first observation is that prefix filtering is a crucial part of any Internet security solution:

Fig. 2: We show the probability that, for a randomly chosen (manipulator,victim) pair, the manipulator can attract traffic destined to the victim from at least 10% of the ASes in the internetwork. The manipulator uses the "Shortest-Path Export-All" attack strategy. The first four bars on the left assume that network does *not* use prefix filtering. We show the success of the manipulator's strategy on each of the four BGP security variants, in a network with and without prefix filtering of stubs. The horizontal line in Fig. 2 shows the fraction of attacks that are completely eliminated by prefix filtering; since 85% of ASes in the CAIDA graph are stubs, properly-implemented prefix filtering guarantees that only 15% of manipulators can successfully attack any given victim.

Despite the fact that we used *sub-optimal* strategies for the manipulator, we can make two observations:

1. Even if we assume the manipulator runs the sub-optimal "Shortest-Path Export-All" attack strategy on a network that has S-BGP but not prefix filtering, he can still attract 10% of the ASes in the internetwork with probability $> 10\%$. Furthermore, more clever strategies for S-BGP (*e.g.*, Figs. 15 and 16) might increase the manipulator's probability of success to the point where prefix filtering alone performs even better than S-BGP alone.
2. Even if both S-BGP *and* prefix filtering are used, there is still a non-trivial 2% probability that the manipulator can attract 10% of the ASes in the internetwork. Better attack strategies could increase this probability even further. This is particularly striking when we compare with the normal case, where the manipulator manages to attract 10% of the ASes in the internetwork with about 10^{-4} probability (not shown).

4.3. Attack strategy on different protocols

In the interests of simplicity, Section 4.2 focused specifically on the probability of attracting 10% of the ASes in the internetwork in Fig. 2. We now present the full picture.

Fig. 3: We show the complimentary cumulative distribution function (CCDF) of the probability that at least a x-fraction of the ASes in the internetwork forward traffic to the manipulator when he uses the "Shortest-Path Export-All" attack strategy. Probability is taken over the uniform random choice of a victim and manipulator, and observe that Fig. 2 simply presents a cross-section of these

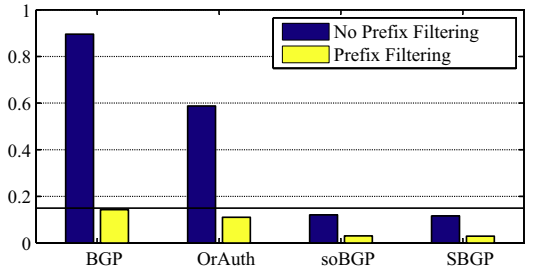


Fig. 2. Lower bounds on the probability of attracting at least 10% of ASes in the internetwork.

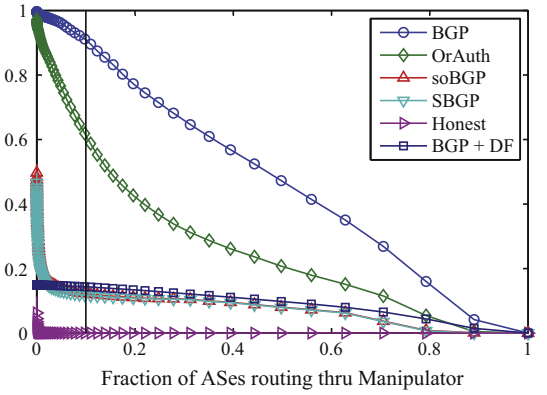


Fig. 3. CCDF for the "Shortest-Path Export-All" attack strategy.

results at the x-axis value of $x = 10\%$. Because this figure carries quite a lot of information, we walk through a few interesting points:

BGP curve. Here, the manipulator *originates*, *i.e.*, announces that he is directly connected to, the victim prefix. This curve looks almost like the CCDF of a uniform distribution, since the manipulator and the victim both announce one-hop paths to the prefix, and are thus are about equally likely to attract traffic.

Origin authentication curve. This time the manipulator announces that he has a direct link to the AS that legitimately owns the victim prefix. Because the manipulator's path is now two hops long, the amount of traffic he can attract on average is reduced.

soBGP and S-BGP curves. For the attack on soBGP, the manipulator announces the shortest path that *exists* in the AS graph. For the attack on S-BGP (and data-plane verification), the manipulator announces the shortest *available* path that he learned from his neighbors. Interestingly, the soBGP and S-BGP curves are almost identical, despite the fact that S-BGP provides stronger security guarantees than soBGP (see also Section 4.4).

Normal curve. Here the manipulator behaves 'normally', *i.e.*, using the ranking and export policies of Section 2.2.

This curve looks almost like a delta-function at $x = 0$. That is, a randomly-chosen AS is likely to attract only a negligible fraction of the ASes in the internetwork by behaving normally.

BGP + PF (prefix filtering) curve. Prefix filtering eliminates all "Shortest-Path Export-All" attack strategies on BGP by stubs, *i.e.*, by 85% of ASes. Thus, this is approximately 'BGP' curve scaled down to 15%.

Different-sized ASes are equally affected. This paper consistently measures the manipulator's success by counting the number of ASes that route through him as a result of his attack strategy. Of course, certain ASes might be more important than others. To this end, we produced versions of Fig. 3 that count the fraction of ASes *of a given size* that route through the manipulator: (a) All ASes, (b) ASes with at least 25 customers, and (c) ASes with at least 250 customers. We omit these graph as they were almost identical to Fig. 3.

4.4. S-BGP forces long path announcements

Figs. 2 and 3 show that S-BGP is *not* much more effective in preventing "Shortest-Path Export-All" attack strategies than the less-secure soBGP. To understand why, let's compare the lengths of the path that the manipulator can announce with soBGP and S-BGP:

Fig. 4: We show the probability that the manipulator can announce a path that is shorter than the normal path, *i.e.*, the path he would have chosen if had used the rankings in Section 2.2. Probability is taken over a randomly-chosen victim, and a manipulator that is randomly chosen from one of the following four *classes*: (a) Any AS in the graph, (b) *Non-stubs*, or ASes with at least one customer (c) Medium-sized ASes with at least 25 customers, and (d) Large ASes with at least 250 customers. If we focus on the results for S-BGP, it is clear that *larger* ASes are more likely to find *shorter* paths through the network; this follows from the fact that these ASes are both more richly connected (*i.e.*, they have large degree), as well more central (*i.e.*, they are closer to most destinations in the internetwork). Furthermore, we can also see that ASes (especially small ASes) are more likely to find short paths with soBGP than they are with S-BGP.

From Fig. 4, we can conclude that S-BGP is doing exactly what it is designed to do: it is limiting the set of paths the attacker can announce, thus forcing him to announce longer paths. However, in light of the results in Figs. 2 and 3, we must ask ourselves why forcing the manipulator to announce longer paths does not seem to significantly limit the amount of traffic he attracts. We could explain by arguing that path lengths in the Internet are fairly short, (averaging about 5 hops in our simulations, see Appendix C); so the paths that the manipulator can get away with announcing in soBGP are only slightly shorter than those he can announce with S-BGP. Indeed, as we show in the next section, the fact that AS paths are normally so short means that the length of the manipulator's path often plays less of a role than the set of neighbors that he exports to.

4.5. Export policy matters as much as length...

We now show that the attacker's export policy is as important as the length of the path he announces:

Fig. 5: We show another CCDF of the probability that at least a x-fraction of the ASes in the internetwork forward traffic to the manipulator; probability is taken over a randomly-chosen victim, and a manipulator chosen randomly from the class of ASes that have at least 25 customers. We consider three different strategies: (a) Announce the shortest available path to all neighbors (equivalent to the "Shortest-Path Export-All" attack strategy on S-BGP), (b) Announce the normal path to *all* neighbors, and (c) Announce the normal path using the normal (**GR2** and **NE**) export policy.

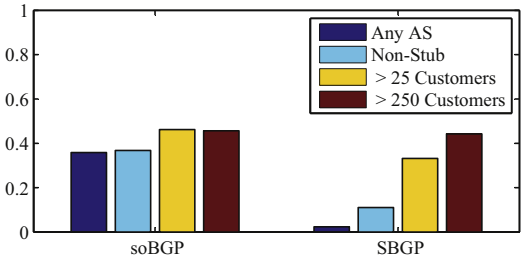


Fig. 4. Probability of finding a shorter path.

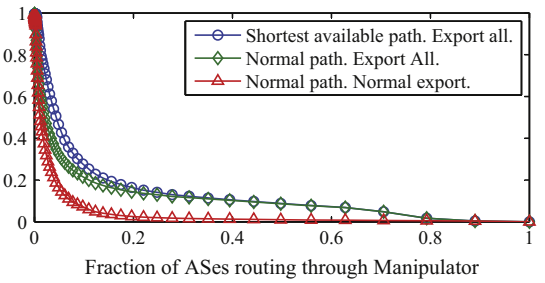


Fig. 5. Aggressive export policies.

This figure shows that, on average, announcing a *shorter* path is less important than announcing a path to more neighbors (*i.e.*, the curves for (a) and (b) are very close, while the curves for (b) and (c) are quite far apart). When we considered smaller manipulators (not shown), the curves for (a) and (b) are even closer together. We can explain the small gap between (a) and (b) by noting that the manipulator's normal path is very often also his shortest path (this holds for 64% of (manipulator, victim) pairs from this class); and even when it is not, his normal path tend to be quite short.

To understand the larger gap between (b) and (c), we note that by violating the normal export policy, the manipulator can announce paths to his providers, even when his normal path is not through a customer. His providers are more likely to choose the customer path through the manipulator, over some possibly shorter, non-customer path. This attack strategy is also sometimes called a "route leak" [11], and the Moratel incident [44] in 2012 is one example of a such a route leak occurring the wild.

4.6. ...Especially when using provider paths!

The effectiveness of the export-all strategy is particularly pronounced when we zoom in on the cases where the normal path is a *provider path* (which happens for about 34% of (manipulator,victim) pairs conditioning on the manipulator having at least 25 customers).

Fig. 6: This is Fig. 5 conditioned on the fact the manipulator's normal path is through a provider. In this case, the manipulator's normal path is always his shortest available path, [4] so we show only two strategies instead of three (*cf.*, Fig. 5): (b) Announce the normal path to all neighbors (c) Announce the normal path using the normal (**GR2** and **NE**) export policy.

[4] By **LP**, if the normal path is a provider path, then all paths available to the manipulator must be provider paths, and by **SP**, he chooses the shortest one.

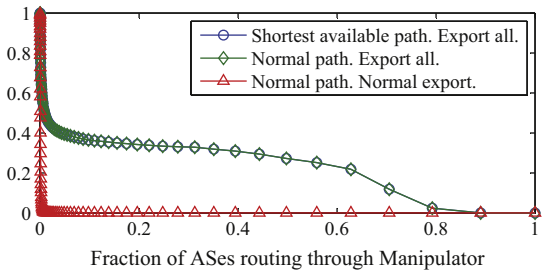


Fig. 6. Aggressive export policies when the normal path is through a provider.

The figure shows that exporting to all neighbors dramatically increases the amount of traffic attracted by the manipulator. This follows from the fact that the normal (**GR2** and **NE**) export policy requires the manipulator to export provider paths to *customers only* (curve (c)); when the manipulator violates this export policy by exporting to providers and peers as well (curve (b)), his providers will prefer the customer path through the manipulator, which significantly increases the amount of traffic the manipulator attracts. This effect is particularly pronounced here because we considered manipulators with at least 25 customers in this figure (roughly modeling 'Tier 2' ASes), that stand to gain by attracting traffic from their providers, the Tier 1s.

4.7. Tier 2s usually cause the most damage

Next, we would like to determine which ASes in the Internet are likely to be the most successful manipulators. We consider non-stub manipulators from three different classes: (a) Non-stubs (ASes with at least 1 customer), (b) ASes with at least 25 customers, (roughly modeling "Tier 2 ASes"), and (c) Large ASes with at least 250 customers ("Tier 1 ASes").

Fig. 7: We once again show a CCDF of the probability that at least a x-fraction of the ASes in the internetwork forward traffic to the manipulator, when the manipulator launches the "Shortest-Path Export-All" attack strategy on BGP. Despite the fact that the "Tier 1" manipulators are more central than the "Tier 2s", we make the surprising observation that "Tier 2s" manage to attract more traffic than "Tier 1s". In fact, for certain regimes, even smaller non-stub ASes tend to attract more traffic than the "Tier 1s"!

This strange observation is actually easy to explain. In the "Shortest-Path Export-All" attack strategy on BGP, every manipulator (regardless of its size or location in the network) announces a single-hop path to the victim prefix. Thus, announced path length does *not* play a role when we compare across classes of manipulators. On the other hand, despite their centrality, Tier 1 ASes are more expensive to route through than every other AS in the internetwork; a Tier 1 is always a provider or peer of its neighbors, so even if those neighbors learn a short path through the Tier 1, they will prefer to route over a (potentially longer) path through one of their own customers. Furthermore, Tier 2s are more central and richly connected than smaller ASes on the edge of the internetwork, and thus they tend to attract more on average than the smaller ASes ("Non-Stubs").

The reader may be troubled by the fact that the (red triangle) curve for the manipulators with at least 250 customers has a different shape than the other curves in Fig. 7. We saw exactly this effect on all our experiments across different datasets, and one main reason it occurs is because the AS graph we used only has 34 ASes (out of a total of 33 K ASes) that have at least 250 customers; this is consistent with the idea that are about 12 (or so) Tier 1 ASes in the Internet. Because we had so few manipulators to choose from, the effect of individual manipulators on the results become more pronounced, and the curves become less smooth.

4.8. S-BGP is vulnerable to stubs

The picture for origin authentication looks about the same as Fig. 7. However, the results change for soBGP and S-BGP/data-plane verification.

Fig. 8: This is the CCDF for S-BGP/data-plane verification (*cf.*, to Fig. 7). "Tier 2" manipulators usually come out on top, except when we consider manipulations that attract 10% of the ASes in the internetwork or less. In this regime, the Tier 1 ASes come out on top, so that the S-BGP curve mimics normal behavior (not shown). Tier 1s tend to attract more traffic than others when everyone is behaving normally, because they are likely to have short customer paths they can announce to all of their (many) neighbors.

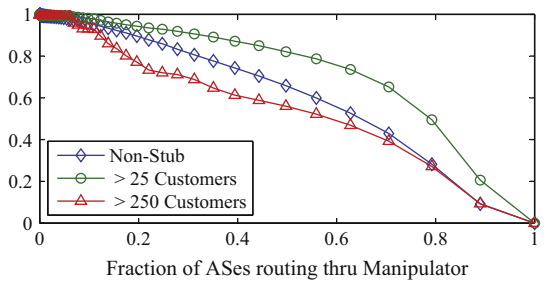


Fig. 7. "Shortest-Path Export-All" attack strategy on BGP by different manipulators.

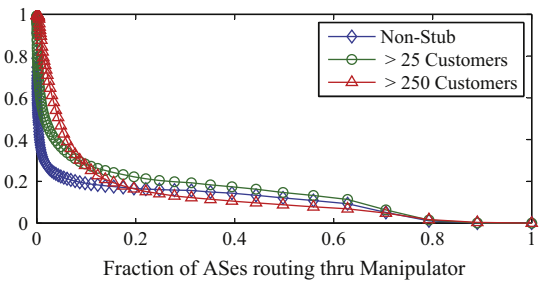


Fig. 8. "Shortest-Path Export-All" attack strategy on S-BGP/data-plane verification by different manipulators.

4.9. Tier 1s are more vulnerable to attacks!

Next, we determine which ASes in the internetwork are most vulnerable to attack. This time, we consider *victims* from three classes: (a) All ASes, (b) ASes with >25 customers, and (c) Large ASes with >250 customers.

Fig. 9: This is another CCDF of the probability that at least a x-fraction of the ASes in the internetwork forward traffic to the manipulator, when the manipulator launches the "Shortest-Path Export-All" attack strategy on BGP. Probability is over all manipulators, and all victims from one of the three classes above.

We make the surprising observation that the "Tier 2" ASes (">25 Customers") tend to be *less vulnerable* than "Tier 1" ASes (">250 Customers), despite the fact that the "Tier 1" ASes tend to be more central and richly connected. To explain this, we once again observe that despite their centrality, Tier 1 ASes are always providers or peers of their neighbors, so that their neighbors will prefer (potentially longer) customer paths that lead to a manipulator at the edge of the internetwork, over a shorter path to legitimate victim Tier 1 ASes. On the other hand, Tier 2 ASes are the customers of the Tier 1s; thus, when they are the victims of an attack strategy, their Tier 1 neighbors, and the customers of these Tier 1s, will tend to prefer the short customer path to the victim (a Tier 2), over the longer path to a manipulator (at the edge of the internetwork). We also note that smaller ASes (represented by the curve corresponding to "All ASes") tend to be the most vulnerable to the "Shortest-Path Export-All" attack strategy on BGP, since legitimate paths to these ASes tend to be slightly longer than the paths to the larger, more central ASes.

The results are even more unexpected when we look at soBGP and S-BGP/data-plane verification:

Fig. 10: This is the CCDF for S-BGP/data-plane verification (*cf.*, to Fig. 9). While the "Tier 2" ASes remain the least vulnerable (for the reasons we discussed above), here we see that the "Tier 1" ASes are even more vulnerable than the smaller ASes at the edge of the internetwork. We explain this roughly as follows: For attacks on S-BGP, the manipulator is forced to announce only available paths that may be quite long. Thus, the amount of traffic he attracts tends to decrease (as compared to the "Shortest-Path Export-All" attack strategy on BGP). Thus, manipulators on the edge of the internetwork tend to attract traffic mostly because (by **LP**) other ASes prefer (possibly long) customer paths over any non-customer paths. Next, because Tier 1 ASes have no providers, Tier 1 victims *cannot* rely on the fact that other ASes prefer customer routes in order to attract traffic to their network; thus, their legitimate routes tend to be less preferable than the ones announced by manipulators at the edge of the internetwork. By contrast, smaller ASes and Tier 2s do have providers, and these providers will prefer shorter, legitimate customer paths to the smaller ASes and Tier 2s, rather than longer customer routes to manipulators at the edge of the internetwork.

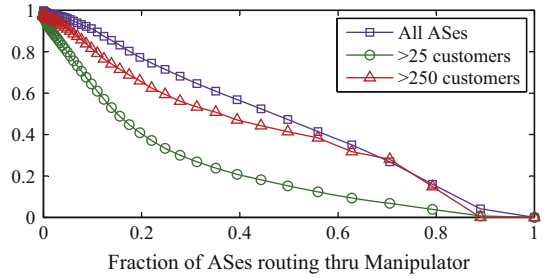


Fig. 9. "Shortest-Path Export-All" attack strategy on BGP for different victims.

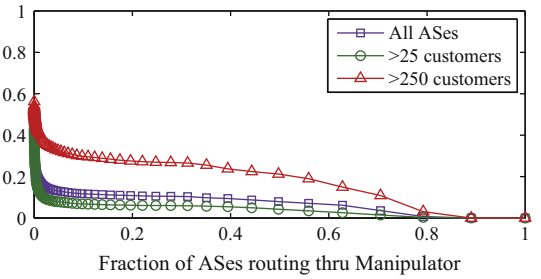


Fig. 10. "Shortest-Path Export-All" attack strategy on S-BGP for different victims.

Even when there is soBGP or S-BGP or data-plane verification (but no prefix filtering), the "Tier 1" ASes remain surprisingly vulnerable to attack by stub ASes. (Section 6.1 has an example of this type of attack.)

4.10. Summary of simulation results

In some sense, our results suggest that secure routing protocols like S-BGP and soBGP are only dealing with one half of the problem: while they do restrict *the path* the manipulator can choose to announce, they fail to restrict his *export policies*. Indeed, because prefix filtering restricts both the export policies and the paths announced by stubs, we find that it provides a level of protection that is at least comparable to that provided by S-BGP, and even data-plane verification, alone.

Even if we eliminate attacks by stubs via prefix filtering, Figs. 7 and 8 show that the internetwork is still vulnerable to non-stub ASes that both (a) deviate from normal routing policies by announcing shorter paths, and (b) deviate from normal export policies by announcing non-customer paths to *all* their neighbors. Furthermore, we have seen that it is exactly these non-stub ASes (and in particular, the Tier 2s) that are in the position to launch the most devastating attacks. The success of these attack strategies can be limited with soBGP, S-BGP, or data-plane verification.

4.11. Prefix filtering challenges

We conclude this section briefly discuss some of the challenges involved in implementing prefix filtering. While the results of this section compare the efficacy of prefix filtering to that of soBGP and S-BGP, these mechanisms differ greatly in (a) the number of ASes that use them on the

Internet today, as well as (b) the trust model for which they were designed.

Implementing prefix filtering. While prefix filtering is a best common practice (BCP) on the Internet today, and is anecdotally known to be used by several large ISPs, its implementation is far from perfect. First, the incentives to implement prefix filtering are lopsided; in some sense, the provider derives little local benefit for itself or its customers, and is instead altruistically protecting the rest of the Internet from attacks by its customers. Secondly, the provider has to maintain up-to-date prefix lists of the IP addresses owned by each of its stub customers, a problem that, thus far, has proved to be challenging [47]. To address the second issue, information in the RPKI can be used by each provider to *automatically* derive prefix lists for their stub customers, an idea that is currently being explored by practitioners [4,43].

What if only large ASes filter? Thus far, we considered a perfect world in which every provider implements prefix filtering, including tiny ASes with only a few customers. In the following, we consider what happens when only the *large* ASes filter announcements from their stub customers:

Fig. 11: Attacks by a given stub are thwarted only if *all* its providers implement prefix filtering. Thus, we presents a pie chart of the stubs (*i.e.*, ASes with no customers), breaking them up by the size of their *smallest* provider. First, note that we present only 85% of the pie; the other 15% of ASes are non-stubs. Thus, the figure shows that if only providers with more the 500 customers were to implement prefix filtering, then attacks by 14% of the ASes in the internetwork would be eliminated (the white slice of the pie only). Similarly, if only providers with more than 25 customers filter, then attacks by $14\% + 14\% + 20\% = 48\%$ of ASes in the internetwork would be eliminated. Thus, reasonable improvements can be obtained even if only ISPs with more than 25 customers implement prefix filtering.

Trust models. We caution that prefix filtering operates in a problematic trust model. Because it is a purely *local mechanism* at each provider, there is no known way for an AS to validate that another AS has implemented prefix filtering properly. This trust model essentially amounts to assuming that every provider is honest. This is in contrast to the trust model used in S-BGP and soBGP; S-BGP, for instance, ensures than even a malicious AS may only announce available paths (as long as it does not collude with, or comprise the keys of, some other AS), and also allows any AS to validate the paths announced by any other AS.

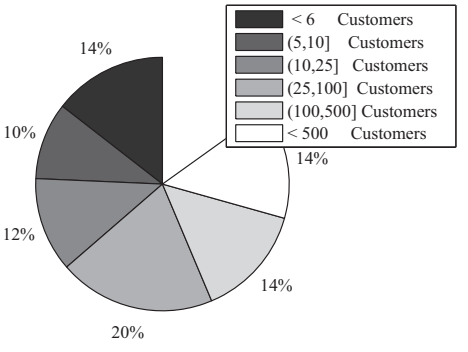


Fig. 11. Distribution of stubs, according to the size of their smallest provider.

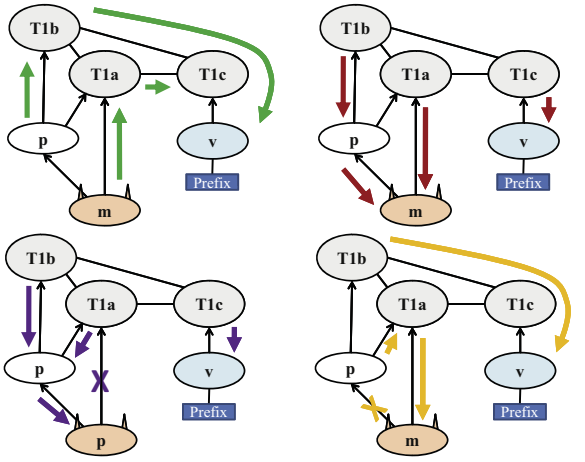


Fig. 12. (a) Normal outcome. (b)–(d) Blackhole.

5. Smart interception attacks

We now turn our attention to traffic interception attacks [3,6,9,45]. In an interception attack, the manipulator would like to attract as much traffic as possible to his network (in order to eavesdrop or tamper with traffic) before forwarding it on to the victim IP prefix. Here was say that an interception attack is a strategy that preserves an available path from the manipulator to victim.

5.1. A stub that creates a blackhole

To provide some intuition, we first show how a manipulator could lose a working path to a victim:

Fig. 12: For simplicity, let's consider an attack on BGP where the manipulator falsely originates the victim's prefix. The manipulator m is a web-hosting company in Illinois, and wants to attract traffic destined for the victim v a web-hosting company in France. The manipulator is a multi-homed stub with two providers, a Tier 1 AS $T1a$, and a Chicago-area telecom provider p. The left figure shows the normal outcome, where the manipulator has a path to victim available through each of his providers. The right figure shows what happens when the manipulator announces the victim's prefix to each of his providers; since each of them prefer short customer paths, they will forward their traffic through the manipulator. The manipulator has now created a *blackhole*; he has no available path to the victim v through either of his providers.

Suppose now that the manipulator tried to be a little more clever, and did *not* announce the victim's prefix to his Tier 1 provider $T1a$. Unfortunately for the manipulator, this strategy still creates a blackhole. As show in the bottom left (purple) figure, $T1a$ will prefer his customer path through manipulator $(T1a, p, m,$ Prefix) over his peer path

to the legitimate prefix ($T1a, T1c$, v, Prefix). Thus, both the manipulator's providers will still forward their traffic to the manipulator, and the blackhole remains. It is easy to see that a blackhole also occurs when the manipulator only announces the victim prefix to his Chicago provider p (see the bottom right (orange) figure).

5.2. When do interception attacks succeed?

The reader may be surprised to learn that there are many situations in which blackholes are *guaranteed not to occur*. We can prove that, within our model of routing policies, the manipulator can aggressively announce paths to certain neighbors while still preserving a path to the victim:

Theorem 5.1. *Assume that* **GR1** *holds, and that all ASes use the routing policies in Section 2.2. Suppose the manipulator has an available path through a neighbor of a type x in the normal outcome. If there is ✓ in entry (x, y) of Table 2, then a path through that neighbor will still be available, even if the manipulator announces any path to any neighbor of type y.*

Appendix E presents the proofs. We also not that the results marked with ✓* hold even if the internetwork does not obey **GR1**. We also observe that this theorem is 'sharp'; if there is an X in entry (x, y) of Table 2, we show by counterexample that the manipulator *can sometimes* lose an available path of type x if he announces certain paths to a neighbor of type y. Indeed, Fig. 12 is a counterexample that proves the X in the lower-right entry of Table 2.

Results of this form were presented in an earlier work [3]. However, Ballani et al. [3] claims that a peer-path *cannot* be lost by announcing to a provider (and vice versa). In Appendix D we present an example contradicting this, that proves the remaining X entries in Table 2.

Tier 1s and stubs. Theorem 5.1 leads to a number of observations, also noted by [3]. First, interception is easy for Tier 1s. Since Tier 1s have no providers, they need only concern themselves with the four upper-left entries in Table 2, which indicate that they can announce paths to all their neighbors. Secondly, interception is hard for stubs. A stub's neighbor is almost always a provider, putting it in the bottom-right entry of Table 2, indicating that aggressive announcements could cause a blackhole as in Fig. 12.

5.3. When do "Shortest-Path Export-All" attack strategies cause a blackhole?

The observations of Section 5.2 are borne out by our experiments. We now show that the "Shortest-Path Export-All" attack strategy often allows the manipulator to intercept traffic without creating a blackhole:

Fig. 13: We show the probability that the manipulator has some available path to the victim if he uses the "Shortest-Path Export-All" attack strategy for each of the four BGP security variants. We present results for a randomly-chosen victim, and a manipulator chosen from the usual four classes (see Fig. 4). We assume that manipulator runs the "Shortest-Path Export-All" attack strategy on each BGP security variant. We can make a number of observations:

1. Manipulators with the *most* customers are *least* likely to create a blackhole. As discussed in Section 5.2, these manipulators are most likely to have an available customer path to the victim, and as shown in the first row of Table 2, can get away with announcing to *all* their neighbors without creating a blackhole.

2. The attack on BGP is most likely to cause a blackhole (*cf.*, the attack on origin authentication, or soBGP). Because the manipulator announces a short path, he is more likely to convince *all* of his neighbors to forward traffic to him, and thus create a blackhole.

We note that our empirical results generally agree with Theorem 5.1; whenever there was a gap between the two, we found a customer-provider loop (*i.e.*, a violation of **GR1**) in the AS graph that we used for running our simulations.

Table 2
Guidelines for interception.

To preserve a path of type...	May announce to neighboring...		
	Customers	Peers	Providers
Customer	✓*	✓*	✓
Peer	✓*	✓*	X
Provider	✓	X	X

5.4. Two interception strategies

Fig. 13 immediately suggests a simple interception strategy that seems to work every time:

"Shortest-Available-Path Export-All" attack strategy: The manipulator should announces his shortest *available* path from the normal outcome to all his neighbors. In fact, this is exactly the "Shortest-Path Export-All" attack strategy on S-BGP.

Fig. 3 shows that this strategy attracts more traffic than the normal strategy, but also suggests that when the network does *not* use S-BGP, there may be better interception attack strategies. Indeed, Fig. 13 shows that there is a non-trivial probability that the manipulator has an available path to the victim, even if he launches the "Shortest-Path Export-All" attack strategy on the BGP. This suggests the following two-phase strategy:

"Hybrid Interception" attack strategy: First, run the "Shortest-Path Export-All" attack strategy on the secure routing protocol, and check if there is an available path to the victim. If no such path is available, announce the

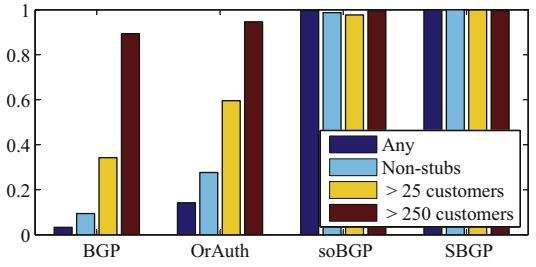


Fig. 13. Probability that the "Shortest-Path Export-All" attack strategy does *not* create a blackhole.

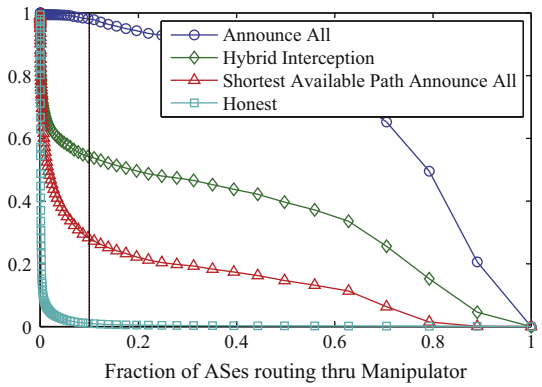


Fig. 14. Interception attacks on BGP.

shortest path that was available in the normal outcome to all neighbors.[5]

By no means do we believe that these two strategies are optimal; indeed, while we evaluated more clever attack strategies, we omitted them here in the interest of brevity and simplicity. What is surprising is that even these simple strategies can be quite effective for certain manipulators.

5.5. Evaluating interception strategies

From the discussion above (Figs. 12 and 13, Section 5.2), it is clear that ASes with very few customers are unlikely to attract large volumes of traffic without blackholing themselves. For this reason, we focus our evaluation on manipulators with at least 25 customers, and for brevity only present attacks on BGP:

Fig. 14: This is a CCDF of the probability that at least a x-fraction of the ASes in the internetwork forward traffic to the manipulator, under the assumption that the network uses BGP. We compare the (a) "Shortest-Path Export-All" attack strategy where the manipulator *is allowed to create a blackhole* (and thus tends to attract more traffic than the interception strategies above), with (b) the two interception strategies above, as well as (c) the normal strategy. Our key observation is that the "Hybrid Interception" attack strategy intercepts a large fraction of traffic; *e.g.*, at least 10% of the ASes in the internetwork with probability over 50%!

5.6. Summary

On average, traffic interception is difficult for stubs, but a manipulator with many customers can quite easily launch an interception attack. Indeed, manipulators with many customers can intercept a large volume of traffic with even the highly simple "Hybrid Interception" attack strategy. Furthermore, as we shall discuss in Section 6, there may be more clever traffic interception attacks that allow the manipulator to attract even larger portions of the internetwork.

[5] We note that while this strategy will attract at least as much traffic as the "Shortest-Available-Path Export-All" attack strategy, the manipulator stands a higher chance of getting caught if he creates a blackhole in the first phase of the strategy.

6. Smart attacks are not optimal

We now prove that the "Shortest-Path Export-All" attack strategy is *not* optimal for the manipulator. We present three surprising counterexamples, found in CAIDA's AS graph and then anonymized, each one of which contradicts the optimality of one aspect of the "Shortest-Path Export-All" attack strategy. In Section 6.1, we show that announcing *longer* paths can be better than announcing shorter ones. In Section 6.2 we show that announcing to *fewer* neighbors can be better than to announcing to more. In Section 6.3 we show that the *identity* of the ASes on the announced path matters, since it can be used to strategically trigger BGP loop detection; in fact, this example also proves that announcing a longer path can be better than a prefix hijack (where the manipulator originates a prefix he does not own)!

6.1. Attract more by announcing longer paths!

Our first example is for a network with soBGP, S-BGP or data-plane verification. We show a manipulator that *triples* his attracted traffic by announcing a *legitimate path to the victim, that is not his shortest path*. (This contradicts the optimality of the "Shortest-Path Export-All" attack strategy, which requires announcing shortest paths.) In fact, this strategy is so effective, that it attracts almost as much traffic as an aggressive prefix hijack on unmodified BGP!

Fig. 15: The manipulator m is a small stub AS in Basel, Switzerland, that has one large provider $a1$ that has almost 500 customers and 50 peers, and one small provider AS $a2$ in Basel that has only four neighbors. The victim is European broadband provider v with over 100 customers and 26 peers.

Prefix hijack. In a network with (unmodified) BGP, the manipulator could run a simple prefix hijack, announcing "m, Prefix" to both his providers, and attract traffic from 62% of the ASes in the internetwork (20550 ASes), including 73% of ASes with at least 25 customers, and 88% of ASes with at least 250 customers. However, this strategy both creates a blackhole at the manipulator, and fails against soBGP or S-BGP.

Naive strategy. The upper (green) figure shows the "Shortest-Path Export-All" attack strategy, where the manipulator naively announces a *three-hop* available path, $(m, a1, v$, Prefix) to his provider $a2$. Since ASes $a2$ and $a3$ prefer the customer path that leads to the manipulator, over their existing peer paths, both will forward traffic to the manipulator. He intercepts traffic from 16% of the ASes in the internetwork (5569 ASes), including 25% of ASes with at least 25 customers, and 41% of ASes with at least 250 customers.

Clever strategy. The lower (purple) figure shows the manipulator cleverly announcing a *four-hop* available path $(m, a2, a3, v$, Prefix) to his provider $a1$. The large ISP $a1$ will prefer the longer customer path through the manipulator over his shorter peer connection to victim v, but this time,

the manipulator *triples* the amount of traffic he attracts, intercepting traffic from a total of 56% of the ASes in the internetwork (18664 ASes), including 69% of ASes with at least 25 customers, and 85% of ASes with at least 250 customers.

Thus, we have shown that announcing a longer path, allows the manipulator to attract almost as many ASes as the aggressive prefix hijack!

Why it works. Notice that the manipulator's large provider $a1$ has hundreds more neighbors then his small provider, $a2$, and that the clever strategy attracts large ISP $a1$'s traffic while the naive strategy attracts small AS $a2$. Attracting traffic from the larger AS is crucial to the manipulator's success; in fact, it is more important than announcing short paths.

When it works. This strategy only involves deviating from normal export policy, rather than *lying* about paths. Thus, it succeeds against *any* secure routing protocol (except when it is launched by stubs in a network with prefix filtering).

6.2. Attract more by exporting less!

This example is for a network with origin authentication, soBGP, S-BGP, data-plane verification, and/or prefix filtering. We show a manipulator that intercepts traffic from 25% *more* of the ASes in the internetwork by exporting to *fewer* neighbors. (This contradicts the optimality of the "Shortest-Path Export-All" attack strategy, which requires exporting to all neighbors.)

Fig. 16: The victim v is a stub network for a liberal arts college in Illinois. The manipulator is a large ISP m, and is competing with the victim's other provider $p1$, a local ISP in Illinois, to attract traffic destined for v.

Naive strategy. The "Shortest-Path Export-All" attack strategy requires the manipulator to announce his path to all his neighbors. On the left, when the manipulator announces a path to his Tier 2 provider $T2$, both $T2$ and its two Tier 1 providers $T1a$ and $T1b$ will route through the manipulator. As a result, $T1a$ and $T1b$ use *four-hop* paths to the victim, and the manipulator attracts traffic from 40% of the ASes in the internetwork, (13463 ASes), including 44% of the ASes with at least 25 customers, and 32% of ASes with at least 250 customers.

Clever strategy. On the right, the manipulator increases his traffic volume by almost 25%, by suppressing paths to his Tier 2 provider $T2$. Because $T2$ no longer has a customer path to the victim, he is forced to use a peer path through $T1c$. Because $T2$ now uses a peer path, he will not export a path to the two Tier 1 $T1a$ and $T1b$. The Tier 1s $T1a$ and $T1b$ are now forced to choose shorter three-hop peer paths to the victim through the manipulator. Because the $T1a$ and $T1b$ now announce shorter paths to their customers, they become more attractive to the rest of the internetwork, the volume of traffic they send to the manipulator quadruples. Thus, the manipulator attracts 50% of the ASes in the internetwork (16658 ASes), including 59% of the ASes with at least 25 customers, and 29% of ASes with at least 250 customers.

Why it works. The manipulator's strategy forces influential ASes (*i.e.*, Tier 1s) to choose shorter peer paths over longer customer paths. He does this by suppressing announcements to certain providers, thus eliminating certain customer paths from the internetwork.

When it works. This strategy only involves using a clever export policy, rather than lying about paths, and therefore succeeds against *any* protocol, including data-plane verification. While one might argue that this manipulator has not done anything wrong here, we present this example as a proof that announcing paths as widely as possible is, surprisingly, not optimal for attracting traffic.

6.3. Attract more by gaming loop detection!

To show that the identity of the ASes on the announced path can affect the amount of attracted traffic, our last example involves gaming BGP loop detection. (This contradicts the optimality of the "Shortest-Path Export-All" attack strategy, which suggests announcing *any shortest path*, regardless of the identity of the ASes on that short path.) While gaming loop detection was explored in other works, *e.g.*, [45,6,21], this example is singular in that it proves that this attack strategy can attract more traffic than an aggressive prefix hijack.

Fig. 17: The manipulator m is a stub in Clifton, NJ with two providers. The manipulator wants to blackhole traffic destined for a prefix owned by the victim v, a stub in Alabama.

Naive strategy: Aggressive Prefix Hijack. In a prefix hijack, the manipulator m announces the path (m, Prefix) to both of his providers, $a1$ a NJ-area ISP, and $T1x$ a large American

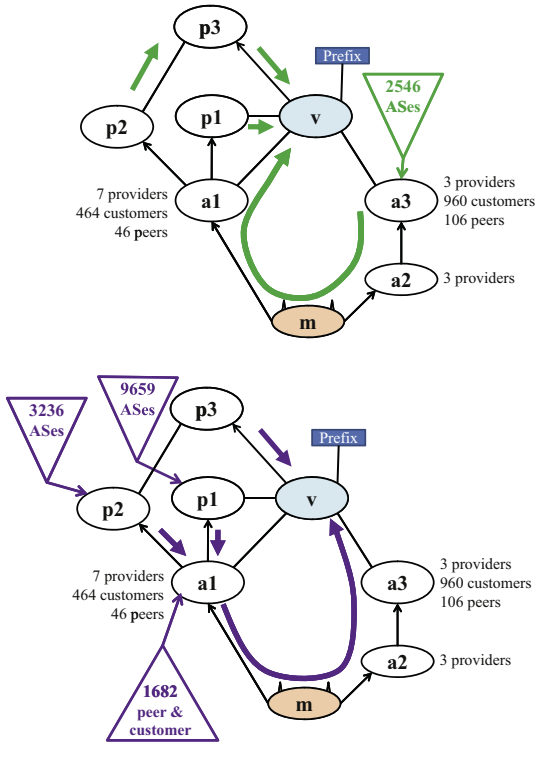


Fig. 15. Announcing a longer path.

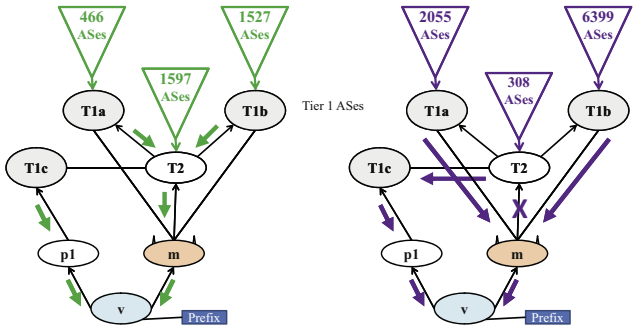


Fig. 16. Exporting less.

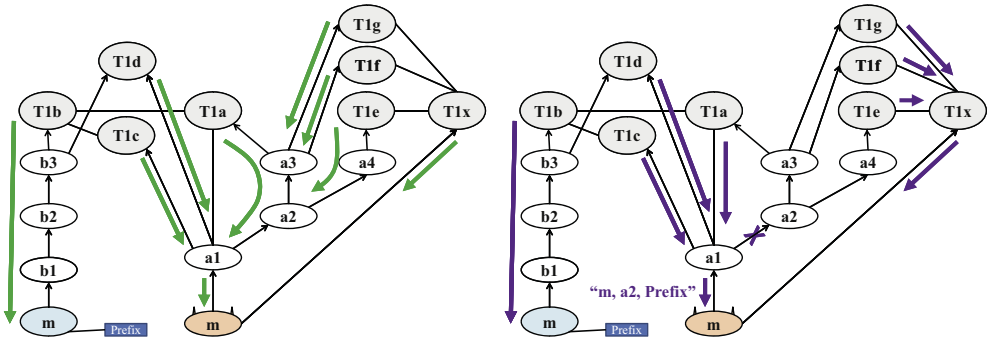


Fig. 17. Using false loops.

backbone provider that is often considered to a be a Tier 1 network. The manipulator manages to attract traffic from most of the Tier 1 ASes in the internetwork. However, many of these Tier 1's, namely $T1a$, $T1e$, $T1f$, and $T1g$, use long, five-hop customer paths to the manipulator. The results of the attack is that the manipulator manages to blackhole traffic from a total of 32010 ASes.

Clever strategy: False Loop Prefix Hijack. We now show how the manipulator can attract traffic from an additional 360 ASes by using a clever 'false-loop prefix hijack' attack. Now, the manipulator's clever strategy is to announce the path (m, Prefix) to his large provider AS $T1x$, while announcing the false loop ($m, a2$, Prefix) to his other provider AS $a1$. As such, AS $a2$ will no longer forward traffic to his customer $a1$, choosing to forward traffic over an alternate peer path (not shown). Thus, the manipulator has eliminated a customer path from the network, and many of the Tier 1 ASes, including $T1a, T1e, T1f$, and $T1g$, will be forced to forward traffic over shorter peer paths. (Thus, $T1e, T1f$, and $T1g$, now use a three-hop peer path, instead of five-hop customer paths used in the simple prefix hijack.) These ASes now become more attractive to the rest of the internetwork, increasing the volume of traffic flowing through the manipulator to 32370 ASes. Notice that the manipulator's strategy ensures that his provider $a1$ still forwards its traffic to the manipulator. Since quite a few Tier 1 ASes, namely $T1a, T1c$, and $T1d$, route through the manipulator's provider $a1$, the false loop prefix hijack strategy ensures that the manipulator does not lose a large amount of traffic by eliminating customer paths from the network.

Why it works. The manipulator games BGP loop detection, effectively removing edges from the network (*i.e.*, the edge between $a1$ and $a2$), to force large ISPs to choose shorter peer paths over longer customer paths.

When it works. This strategy involves lying about the path announced by an innocent AS (*i.e.*, AS $a2$). Because S-BGP and data-plane verification prevent lying about paths, this strategy only works with BGP, origin authentication, or soBGP.

7. Finding optimal attacks is hard

After all the bizarre attack strategies in Section 6, the reader might not be surprised by the following:

Theorem 7.1. *If ASes use the routing policies of Section 2.2, then finding a manipulator's optimal traffic attraction attack strategy is NP-hard.*

This theorem holds for (a) any of the secure protocols variants and (b) also covers interception attacks; our proof uses a reduction to the standard NP-hard problem of finding the maximum independent set of nodes in a graph. We also show that it is hard to approximate the optimal attack within a constant factor *i.e.*, we cannot even design an algorithm that gets "close" to the optimal attack on a general AS graph. This suggests that a full characterization the manipulator's optimal attack strategy will remain elusive.

We present a version of this theorem that shows that in the case of BGP, origin authentication, or soBGP, it is hard

for the manipulator to decide which path to announce to each neighbor. (The result holds even if the manipulator has a small constant number of neighbors.) On the other hand, the reader might suspect that the finding the optimal attack strategy becomes easier if the manipulator is only allowed to announce an available path, as with S-BGP. Surprisingly, this is not the case; we present another version of this theorem that shows that even if the manipulator is forced to announce his normal path, it is still hard for him to choose the optimal set of neighbors to announce paths to. (These results are meaningful only when the manipulator has a large number of neighbors.)

Proof sketch (Fig. 18). Our proof is in Appendix F and proceeds in two stages. First, we present a special internetwork topology 'gadget' called DILEMMA, and then we use the DILEMMA gadget to reduce from our problem (*i.e.*, finding the most damaging traffic attraction strategy) to the standard NP-hard problem of finding the maximum independent set of nodes in a graph. Then, we show how a DILEMMA can exist for the different secure routing protocols considered in this paper. In a DILEMMA internetwork (Fig. 18), the manipulator m wants to attract the traffic for the victim d from two influential ASes c_1 and c_2, who carry traffic from the majority of the network. A DILEMMA construction must guarantee that m can attract each of the ASes individually, but cannot attract both ASes simultaneously.

8. Related work

Early papers on routing security have typically focused on designing new security extensions to BGP (see [6] for a survey). These papers typically use a particular attack model to analyze the proposed protocol, and compare it to BGP, but understandably do not address attacks that exist outside of their models, like the traffic-attraction attacks we studied here.

Another, more theoretical line of work [13,14,21,36] considers strategic attacks launched by economically-motivated ASes. These papers construct example topologies—sometimes quite contrived—where an AS can manipulate a particular variant of BGP. However, these papers do not define a specific attack strategy, investigate the optimality of attacks, or demonstrate whether the example topologies exist in practice. In contrast, we evaluate attacks on an empirically-measured AS-level topology, and show that our counterexamples are realistic by finding them in the AS level topology.

There have been many works that empirically investigate attacks on BGP (see [6] for a survey). Our work is most closely related to an earlier study of prefix-hijack and interception attacks [3] and a study [38] that appeared subsequently to the publication of this work in SIGCOMM'10. Ballani et al. [3] focuses on (unmodified) BGP and two specific attacks (i.e., prefix-hijack and invalid-next-hop attacks); in contrast, here we consider attacks against a variety of secure routing protocols. We also show that the attacks considered in [3] are suboptimal (Section 6.3), and prove that finding the optimal attack is NP-hard. The work in [3] also suggests guidelines for interception similar to the ones we present in Table 2, but our guidelines correct a small error in [3]'s work (Section 5.2). Subsequently to our work, Lychev et al. [38] used a similar model to study the security benefits provided by *partially-deployed* soBGP/S-BGP/BGPSEC, vis-a-vis those already provided by origin authentication. This work focuses on the efficacy of protocols in *full deployment*, (*i.e.*, when every AS on the Internet has adopted the security protocol). By contrast, Lychev et al. [38] considers a world in which soBGP/S-BGP/BGPSEC is partially deployed (*i.e.*, when some ASes have adopted the protocol, but others have not), and discusses the security benefits and new vulnerabilities that are introduced when the secure protocol coexists alongside legacy insecure BGP.

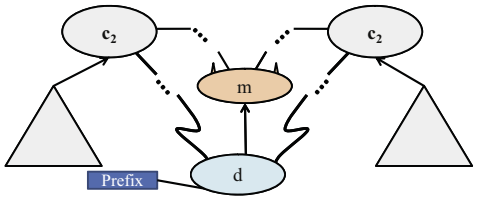


Fig. 18. DILEMMA for proving hardness.

Our work is also related to an earlier work [7] that (among other things) compares several BGP security protocols under partial deployment in a simplified model that ignores business relationships, and instead assumed that normal ASes prefer shortest paths and export paths to all neighbors. This simplification means that soBGP and S-BGP are the same within their model, making it difficult to compare across protocols. In contrast, we focus here on *full* deployment, using a model that more realistically captures realistic routing policies. Chang et al. [7] is also part of a complementary line of work [19,22] that studies the *incentives* for ASes to adopt routing security in their networks.

Another relevant work is [48], which suggests that "data delivery" should be secured instead of "routing". Our analysis has operated within an important constraint of BGP – namely, that a single best path to the destination must be chosen by every AS – and we have analyzed protocols that obey these constraints (*i.e.*, origin authentication, soBGP, S-BGP, prefix filtering, *etc.*). Meanwhile, in [48], Wendlandt et al. suggest dispensing with this constraint and allowing an AS to route on multiple paths to a single destination. Their proposal requires cryptographic techniques (*e.g.*, "Stealth Probing" [2] or "Path Quality Monitoring" [24]) for monitoring the delivery of traffic in the data-plane, and switching between paths when quality degrades. Thus, the proposal in [48] can be thought of as orthogonal to the security mechanisms we have presented here; we note, however, the deployment of the proposed data-plane security mechanisms [2,24] requires the adoption of line-rate symmetric cryptography at multiple routers in the Internet, which creates deployment challenges that have some commonalities with those that arise with protocols like S-BGP.

Bibliographical note. This work is the extended version of work that appeared at SIGCOMM'10 [23]; the current version presents new simulation results (Figs. 5–11), robustness tests (Appendix G), proofs of all theorems in [23] (see Appendices F and E), and details of our simulation

methodology (Appendices A, B and C). The details of the algorithms used in our simulations were published as a separate short paper [20].

9. Conclusions

Because we work within a model of routing policies, we caution against interpreting our results as hard numbers that measure the impact of an attack launched by a specific manipulator in the wild. However, the trends uncovered by our quantitative analysis do allow us to arrive at a number of useful insights; indeed, many of these insights are obtained by averaging over multiple possible (manipulator, victim) pairs, and we suspect that they hold up even if some ASes deviate from the policies in our model. Furthermore, the trends we identified were remarkably consistent across multiple AS topology datasets [1,8,12].

While secure routing protocols can blunt traffic attraction attacks, we found that export policies are a very effective attack vector that these protocols do not address. Thus, we suggest that secure routing protocols (*e.g.*, soBGP and S-BGP) should be deployed in combination with mechanisms that police export policies (*e.g.*, prefix filtering). We believe both are needed; prefix filtering to eliminate attacks by stub ASes, and secure routing protocols to blunt attacks launched by larger ASes, (especially since we found that large ASes can launch the most damaging attacks). We note, however, that policing export policies is a significant challenge in practice. Prefix filtering of stubs requires voluntarily compliance from each provider, and it is difficult to check for proper implementation. Moreover, given the complexity of routing policies used in practice on the Internet, we lack even a definition of what it means to deviate from normal export policies. Thus, while anomaly-detection techniques that flag suspicious routes [30,33] could help, understanding these issues remains an area for future research.

Appendix A. Siblings

Because some of our results are based on CAIDA's AS graph from November 20, 2009 [12], which contains sibling-to-sibling edges, our model also includes *sibling* relationships, where two different ASes are owned by the same organization.

A.1. Modeling sibling relationships

Liao et al. [37] provide an excellent treatment of sibling-to-sibling relationships that we adapt here. First, the model of export policies must account for sibling relationships:

GR2s AS b only exports a path via AS c to AS a if at least one of a and c are customers or *siblings* of b.

Our **NE** export rule now uses the modified **GR2s** (see Section 2.2). Next, in addition to considering customer, peer, and provider paths, the work of [37] introduces two new path types:

Sibling down. The first edge(s) on the path are sibling edges, and the first non-sibling edge is a customer-provider edge. A path that contains exclusively sibling edges is also considered sibling down.

Sibling up. The first edge(s) on the path are sibling edges, and the first non-sibling edge a peer-to-peer or provider-customer edge.

Our modified model of local preferences is also based on [37]:

LPs Prefer customer paths, over sibling down paths, over peer paths, over provider paths, over sibling-up paths.

As discussed in [37], captures a type of "hot potato routing", where the AS prefers to send traffic outside its organization rather than carrying it through its own network.

A.2. Siblings in CAIDA's AS graph

Sibling-to-sibling relationships significantly complicate research on AS-level business relationships. In late 2009, CAIDA's approach was based on manually assigning these relationships to two ASes if they are owned by the same organization. This means that a large AS (*e.g.*, AS1239, with almost 1400 customers) can be a sibling of a tiny AS (*e.g.*, AS1803, with only four customers) if the two are owned by the same organization (*e.g.*, Sprint). The problem this causes is best illustrated by an example.

Fig. A.19: We show CAIDA's snapshot of the local topology around AS 1239 and AS 1803. Because CAIDA classes AS1803 and AS1239 as siblings, our model suggest that tiny AS 1803 will carry traffic from his provider AS 11427 to the large network of AS 1239; in fact, our model suggests that AT&T Worldnet's Teir 1 AS 7018, that has over 2.2 K customers, would route all traffic for AS 1239 over the long customer path through the sibling AS 1803. This is, of course, completely ridiculous. In practice, AS1803 is unlikely to advertise transit paths through AS1239 to any of it's providers; AS 1239 essentially acts like a provider for AS1803, despite the fact that the two ASes are owned by the same organization.

To deal with these unbalanced sibling relationships, we preprocess CAIDA's data as follows:

Sibling preprocessing: We convert sibling-to-sibling relationships to customer-provider relationships when at least one sibling has more then seven customers, and one sibling is at least twice the size of the other sibling.

This approach does remove some of the artificially long paths we describe above. However, because CAIDA's

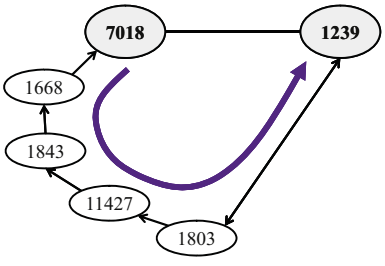


Fig. A.19. The trouble with siblings.

AS-relationship inference algorithms *starts* by using heuristics to assign sibling relationships, and then proceeds to infer the other relationships, we suspect that these sibling relationships can introduce inaccuracies in the results. On the other hand, these inaccuracies do not seem to matter very much, given that the results we obtained on the preprocessed CAIDA dataset matches well with the results we obtained from the Cyclops dataset that has no sibling edges (see Appendix G).

Appendix B. Simulation methodology

At the core of our experiments is a *routing tree algorithm* that simulates the paths that each AS will choose to reach a prefix owned by a legitimate destination AS d. The routing tree algorithm assumes that ASes use the routing policies of Section 2.2, and is implemented using a specialized three-stage breadth-first search (BFS) on the AS graph that we described in detail in [20].

Simulating the "Shortest-Path Export-All" attack strategy. Given a (manipulator, victim) pair (m, d), we use the routing tree algorithm to determine the outcome of each "Shortest-Path Export-All" attack strategy on each secure routing protocol as follows:

BGP. In this attack, both the manipulator m, and the legitimate destination d originate the IP prefix (See Sections 3 and 4.1). To simulate this, we run the routing tree algorithm with *two* roots, m and d.

Origin Authentication/soBGP/S-BGP/data-plane verification. Observe that this strategy requires the manipulator to announce, to all his neighbors, an attack path that is no longer than his shortest available path (see Sections 3 and 4.1). We simulate the "Shortest-Path Export-All" attack strategy using the following trick: First, we augment the AS graph with *fake nodes* corresponding to all the ASes on the manipulator's attack path, excluding the manipulator and victim themselves. These fake nodes are given negative AS numbers. Then, we connect the victim to the manipulator via customer-provider edges through these fake nodes. Thus, the *fake path* is always the manipulator's shortest customer path to victim, that is through an AS with lowest possible AS number (a negative number). Thus, our routing policies in Section 2.2 require the manipulator to choose this path. Thus, to simulate the "Shortest-Path Export-All" attack strategy, it suffices to run the routing tree algorithm on the AS graph augmented with the fake path.[6]

Appendix C. Path distribution

As a sanity check of our routing model, we show the distribution of path lengths and path types.

Fig. C.20: We show the distribution of path length and path type when all ASes behave normally. (The 'None' bars refer to ASes that do not have paths to the destination.) The distribution is over a randomly-chosen destination, and a source chosen from the same four classes as in Fig. 4. We can see majority of paths in the internetwork are short (about 5 hops on average), and further that larger ASes tend to have slightly shorter paths. Furthermore, as expected, we find that smaller ASes tend to use provider paths most frequently, while larger ASes tend to use customer paths most often, and that medium sized "Tier 2" ASes with at least 25 customers uses the largest (relative) fraction of peer paths.

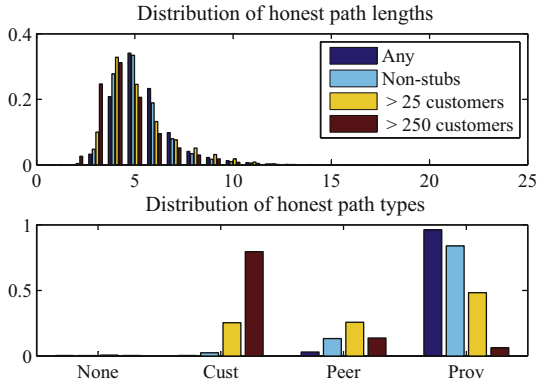


Fig. C.20. Path length and type distributions.

Appendix D. Failed interception attacks

We provide an example that proves the bottom-middle and middle-right X entries in Table 2, and shows that there is an error in [3, Section 2.2].

D.1. Export to provider, disrupt peer

We prove that the manipulator can lose a peer path to the victim by announcing an attractive path to his provider:

Fig. D.21: We consider an attack on BGP, where the manipulator falsely originates the victim prefix. The manipulator m a not-for-profit corporation in New York State, while the victim v is an ISP providing services to multiple universities in Austria. The manipulator has a single provider $a1$, a single peer p, and 44 customers (not shown). The left (green) figure shows the normal outcome, where the manipulator has a paths to victim available through both his peer and his provider. The middle (red) figure shows what happens when the manipulator announces the victim's prefix to his provider AS $a1$; now, his peer AS p has *two* available customer paths of equal length. Since AS $a1$ has a lower AS number than AS $a2$, our **TB** rule requires p to choose the path through $a1$ that leads to the manipulator. The manipulator has now "blackholed" himself; both his peer and his provider forward traffic to the manipulator, and none of the manipulator's customers have any path to the victim prefix.

D.2. Export to peer, disrupt provider

We can also use the example of Fig. D.21, with a slight modification, to prove that the manipulator can lose a

[6] To account for BGP loop detection, we also include a simple check in the routing tree algorithm that cause a real node to reject a path that contains its fake counterpart.

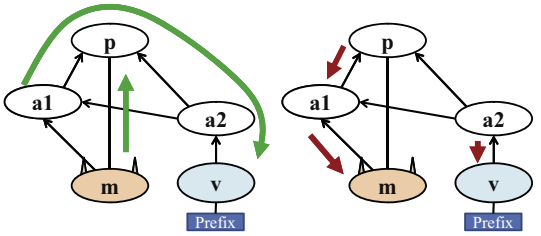


Fig. D.21. Disrupting a path through a peer.

provider path to the victim by announcing an attractive path to his peer. Assume that AS $a1$ has no customer or peer paths, nor any provider paths shorter than two hops, available to the victim v. In that case, if the manipulator *does* announce a path to his peer p, but *not* to his provider AS $a1$, the provider will prefer his two-hop provider path $(a1, m, \text{Prefix})$ over any path to the legitimate victim, and again the manipulator creates a blackhole.

Appendix E. Guidelines for interception

We prove the results marked with a ◢ in Table 2 of Section 5. That is, we provide guidelines that guarantee that a manipulator's attack strategy preserves an available path to the victim IP prefix.

To do this, we consider the *normal outcome*, where the all nodes behave normally, and the *manipulated outcome*, where a single AS m, the manipulator uses some attack strategy that *deviates* from the normal routing policies of Section 2.2. The victim IP prefix is legitimately owned by a destination AS d. Let the nodes on m's available path to d in the normal outcome be $a_1, \ldots, a_{t-1}$, so that m routes to d on the path $ma_1 \ldots a_t$ (where for convenience we will set $d = a_t$). We would like to guarantee that the manipulator's attack strategy leaves him with an available path to d through a_1 (in the manipulated outcome). That is, we want to guarantee that a_1 will not route through m in the manipulated outcome.

E.1. A useful lemma

We use the following useful concept:

Transitive customers. A node b is a *strict transitive customer* of node c if b is connected to c via a path consisting of only customer-provider links as in the right half of Fig. E.22. We also restate here a simple, useful lemma of the Gao-Rexford conditions proved by Gao, Griffin and Rexford in [16].

Lemma Appendix E.1. ([16], [Theorem VII.4]) *If either the path $P = abRc$ or the path $P' = cR'ba$ is available, and if node a is not a customer of node b, then node c is a strict transitive customer of node b over the available path.*

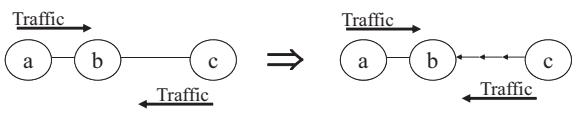


Fig. E.22. Lemma Appendix E.1.

We remark that Lemma Appendix E.1 still holds as long as all the nodes on the available path (except perhaps the last one, closest to the destination) behave normally, according the routing policies in Section 2.2.

E.2. Available path through peers/customers: may export to peers & customers

We prove the four results ◢* results in the top left corner of Table 2. The following claim that does *not* require **GR1**:

Claim Appendix E.2. *Suppose that nodes use the routing policies of Section 2.2. Suppose m's path to d in the normal outcome is a peer or customer path (i.e., a_1 is a peer or customer of m). Then m has an available path through a_1 in manipulated outcome, even if m announces any (possibly false) path to any of his neighboring peers or customers. (See Fig. E.23)*

Proof. First, notice that if m's available path in the normal outcome is a peer or customer path, then **GR2** tells us that a_1's available path in the normal outcome must be a customer path, and Lemma Appendix E.1 immediately tells that for every $i \in [t-1]$, a_{i+1} is a customer of a_i. By **NE**, it follows that every a_i hears an announcement from his customer a_{i+1}.

Let c be any neighbor node of m that heard a path announcement from m. Recall that c must be either a peer or customer of m. We now have two cases:

- Suppose that c is one of the nodes on m's available path in the normal outcome, *i.e.*, $c = a_i$ for any $i \in [t-1]$. We argued above that a_i learns a path from it's customer a_{i+1}. Now, recall that by definition m is a provider or peer of n. It follows from **LP** that for $c = a_i$, the customer path through a_{i+1} is more attractive than the peer or provider path through m, and so $c = a_i$ will prefer to route through a_{i+1}.
- Suppose that n is not one of the nodes on m's available path in the normal outcome. Repeatedly applying **GR2** tells us that the only nodes that can hear about c's path through m must be strict transitive customer of c. Suppose that some a_i for $i \in [t-1]$ hears about the path through m. It follows that a_i learns about the path through m from his provider. Again, by **NE** a_i hears an

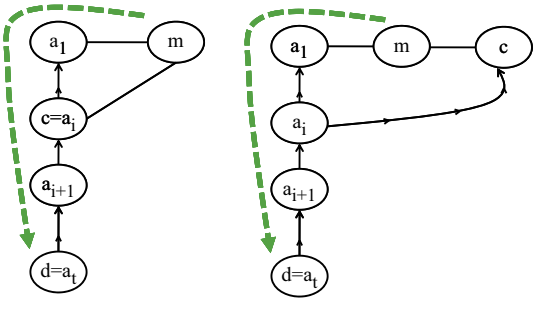


Fig. E.23. Case 1 (left) and Case 2 (right) in Claim Appendix E.2.

announcement from his customer a_{i+1}, and by **LP** this customer path through a_{i+1} is preferred over the provider path through m.

It follows that in each case, every a_i for $i \in [t-1]$ will prefer to route through a_{i+1} instead of routing through m. In particular a_1 has a path to d that does not go through m. By **NE**, a_1 will announce this path to m and the claim follows. □

E.3. Available path through customers: may export to providers

In Appendix D.1, we presented an example that proves that if a_1 is *peer* of m, then m may lose an available path through a_1 by lying to one of his neighboring providers. However, we now prove the ↙ in the top right of Table 2, showing that if a_1 is a *customer* of m, then m can even get away with lying to his neighboring providers. This claim requires **GR1**:

Claim Appendix E.3. *Suppose that* **GR1** *holds, and that nodes use the routing policies of Section 2.2. Suppose m's path to d in the normal outcome is a customer path (i.e., a_1 is a customer of m). Then m has a available path through a_1 in the manipulated outcome, even if m announces any (possibly false) path to any of its neighbors.*

Proof. Now, observe that if a_1's available path to d in the manipulated outcome is unchanged, then by **NE** a_1 announces this path to m and we are done. Thus, we suppose that the path $a_1 \dots a_t d$ is not used in the manipulated outcome. It follows that there must be some node a_i for $i \in t$ that is closest to the destination d that forwards traffic over a different path in the manipulated outcome (*i.e.*, different from the $a_i \dots a_t$ path he used in the normal outcome). The proof now follows from the following (backward) induction from $j = 1 \dots i$. (See Fig. E.24)

Base case. Let $a_{j+1} = a_{i+1}$. From the way we defined a_i, it follows that a_{i+1} uses the same customer path to d in the normal outcome and the manipulated outcome, so it follows that a_{i+1}'s available path does not go through m.

Induction step. Suppose that in the manipulated outcome a_{j+1} uses a customer path to d that does not go through m. Then in the manipulated outcome a_j also forwards along a customer path to d that does not go through m.

We now prove the induction step. First, observe that the Lemma Appendix Appendix E.1 and the fact that a_1 uses a customer path in the normal outcome immediately tells us that a_{j+1} is a customer of a_j. By **NE**, a_{j+1} must export a path to a_j in the manipulated outcome; thus, a_j has a customer path available in the manipulated outcome. By **LP**, it follows that whatever path a_j chooses in the manipulated outcome must also be a customer path. To finish the proof of the induction step, we shall show, by contradiction, that this path does not go through m: Suppose that the available path that a_j chooses in the manipulated outcome goes through m. Then, since this path is a customer path, Lemma Appendix E.1 tells us that the manipulator m as a strict transitive customer of a_j along this path. Now recall

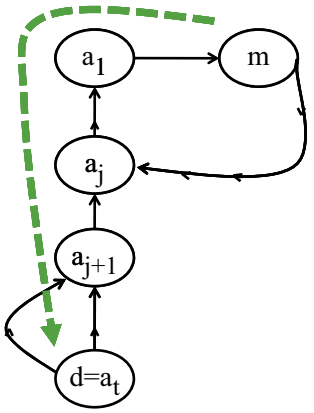


Fig. E.24. Proof of the induction step in Claim Appendix E.3.

that m uses a customer path in normal outcome, and apply Lemma Appendix E.1 again to obtain that a_j must be a strict transitive customer of m. It follows that there is a customer-provider loop in the AS-graph (between a_j and m), which violates **GR1**, and we have arrived at our contradiction.

From the induction, we learn that a_1 must use a customer path in the manipulated outcome that does not go through m. By **NE**, a_1 announces this path to m and the claim follows. □

E.4. Available path through providers may export to customers

We showed how a manipulator might disrupt an available path through a *provider* by announcing to a *provider* (Section 5.1), or a *peer* (Appendix D.2). We now show that a manipulator that wants to preserve an available path through a provider *may* export any path to his *customers*, proving the ↙ on the bottom left of Table 2. We again rely on **GR1**:

Claim Appendix E.4. *Suppose that* **GR1** *holds, and that nodes use the routing policies of Section 2.2. Suppose m's path to d in the normal outcome is a provider path (i.e., a_1 is a provider of m). Then m has a path available through a_1 in the manipulated outcome, even if m announces any (possibly false) path to any of its neighboring customers (See Fig. E.25).*

Proof. Since m only announces paths to his customers, repeated applications of **GR2** immediately tell us that the only nodes that can hear about paths through m are strict transitive customer of m. Now consider m's available path $a_1 \dots a_t$, and let a_p be the node closest to m such that m is a strict transitive customer of a_p. (We know that a_p exists since in particular a_1, a provider of m, is one such node.) We now show that no a_i will choose to route through m:

- Suppose some node a_i for $i \in [p]$ learns about the path through m. We argued above that a_i must be a strict transitive customer of m. However, by the definition of a_p, m is also a strict transitive customer of a_i! It

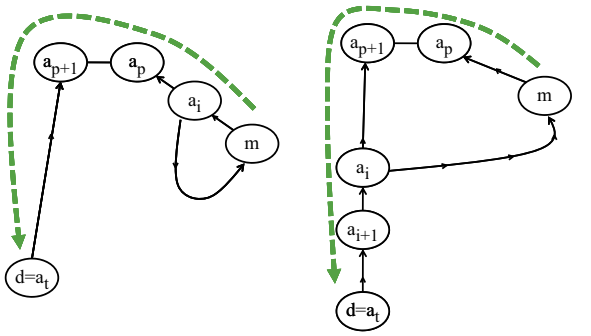


Fig. E.25. Case 1 (left) and Case 2 (right) in Claim Appendix E.4.

follows that there is a customer-provider loop in the AS graph, which violates **GR1**. It follows that no a_i for $i \in [p]$ will learn about the path through m.

- Suppose some node a_i for $i = p \ldots t-1$ learns about the path through m. Above we argued that a_i must be a strict transitive customer of m, so it follows that a_i learns about the path through m from his provider.
  Now, by the definition of a_p and **GR2** we know that a_{p+1} is either a peer or customer of a_p. Applying **GR2** again tells us that a_{i+1} is a customer of a_i for each $i = p+1 \ldots t-1$. By **NE**, we know that a_{i+1} announces a path to a_i for every $i = p \ldots t-1$. It follows that for every a_i for $i = p \ldots t-1$, the path it learns through its peer or customer a_{i+1} is more attractive than the provider-path through m.

It follows that in each case, every a_i for $i \in [t-1]$ will route through a_{i+1} instead of routing through m. In particular a_1 will have a path to d that does not go through m. By **NE**, a_1 will announce this path to m and the claim follows. □

Appendix F. Finding optimal attacks is hard

We now prove the results discussed in Section 7. We show that, from the perspective of the manipulator, finding the optimal traffic attraction attack on BGP is computationally hard. We shall then show that, in fact, not only is finding the optimal attack hard, but even finding a "reasonable" attack, that is "not far" from the optimum, *i.e.*, approximates the optimum, is computationally hard. Our hardness results are obtained via a general proof technique that can be applied to show similar impossibility results for optimal (and approximate) traffic attraction attacks on other security enhancements to BGP (*e.g.*, SBGP, soBGP, and more).

We start by presenting our proof technique. We then show how it can be used to obtain hardness results for traffic attraction attacks on BGP; these results amount to showing that its hard for the manipulator to decide which *paths* to export to which neighbors. We then move on to showing the even if the manipulator is restricted to announcing he normal paths (*e.g.*, because the network uses data-plane verification), that it is still hard for the manipulator to decide which *neighbors to export to*.

F.1. Key ideas

The DILEMMA network. Our computational hardness results rely on showing the potential existence of the following scenario (see Fig. 18): The manipulator m is directly connected to the destination d. m wishes to attract as much traffic as possible, while all other nodes behave normally. The network contains two nodes, c_u and c_v, each with many direct and indirect customers whose routes to d go only through it. The number of nodes in the trees beneath c_u and c_v, that are of equal size, is significantly bigger than the number of nodes in rest of the network. Hence, m's main goal is to attract c_u and c_v's traffic. However, in our constructions below, m shall always be able to attract *either* c_u's *or* c_v's traffic, but will be *unable* to attract *both* nodes' traffic *simultaneously*. Thus, m will have to choose which one of the two nodes to attract, inevitably losing the traffic of the other node and of all nodes in the subtree beneath it. m's inability to attract both c_u and c_v (alongside its ability to attract each of them alone) shall play a crucial role in our proofs.

Once we prove the existence of a small network as described above, that we term "DILEMMA", we use it as a building block in a reduction from the MAX-INDEP-SET problem, that is a notoriously computationally hard problem.

The MAX-INDEP-SET problem. The MAX-INDEP-SET problem is defined as follows:

Definition Appendix F.1 (*independent sets*). Let $G = (V, E)$ be a graph. A subset of the vertices $I \subseteq V$ is an *independent set* if there is no edge in E between two vertices in I.

Definition Appendix F.2 (*MAX-INDEP-SET*). In the *MAX-INDEP-SET* problem the input is a graph $G = (V, E)$ and the objective is to find an independent set I of maximum size.

The following is well known:

Theorem Appendix F.3. *MAX-INDEP-SET is NP-hard.*

Reducing from MAX-INDEP-SET. We now outline our reductions from MAX-INDEP-SET to the problem finding an optimal attack on BGP (or security enhancements to BGP), that establish the computational intractability of the latter.

Given an instance of MAX-INDEP-SET $G = (V, E)$ we construct a network such that computing the traffic-attraction-maximizing attack in the network is equivalent to computing a maximum independent set in G. The node-set in our network contains the destination node d, the manipulator m, and a node c_v for each vertex $v \in V$ (and some additional nodes, as explained below). m is directly connected to d.

We ensure that, for each edge $e = (u, v) \in E$, m shall only be able to attract either c_u's or c_v's traffic, but not both nodes simultaneously, by constructing DILEMMA for c_u and c_v (adding nodes and links appropriately). Importantly, our constructions of DILEMMA gadgets are consistent, in the sense that if the manipulator cannot attract node c_v in one such gadget (because it chose to attract the other

node in that gadget), then it also cannot attract c_v in all other DILEMMA gadgets that c_v participates in. Fig. F.26 illustrates the vertex-specific, edge-specific, and general components of each DILEMMA constructions (for each pair of neighboring nodes, c_u and c_v, that are connected by an edge (u, v) in E).

Now, consider an attack by m. Observe that because the trees beneath the c_v's constitute the vast majority of the nodes in the network, and because the nodes in the tree beneath each of the c_v's can only connect to d via that node, the success of m's attack is measured by how many of the c_v's it was able to attract. By construction, if two vertices in V, u and v, are connected by an edge in G then m cannot attract both c_u and c_v and thus the vertices corresponding to nodes that m is able to attract form an independent set in G. The converse is also true: Let $I \subseteq V$ be an independent set in G, then m can attract all the c_v's corresponding to vertices in I (because no two such nodes participate in a DILEMMA construction).

Therefore, a maximum independent set in G corresponds to a traffic-attraction-maximizing attack in our network, and vice versa. The NP-hardness of MAX-INDEP-SET (and the fact that our reduction is computationally-efficient) now implies the NP-hardness of finding an optimal attack.

On the hardness of approximating the optimal attack. In fact, the close connections, presented above, between independent sets in G and traffic attraction, when combined with the following theorem, due to Hastad, imply a stronger result.

Theorem Appendix F.4 26. *Given a graph $G = (V, E)$, finding an independent set of size at least $\frac{OPT}{|V|^{\frac{1}{2}-\epsilon}}$, where OPT is the size of the maximum independent set in G, is NP-hard.*

Using the above theorem, and the exact same construction as before, we can now show that not only is finding the optimal attack computationally-hard, but so is finding an attack that approximates (in terms of number of attracted nodes) the optimal attack within any constant factor!

F.2. Hardness of optimal attacks on BGP

We present the following theorems:

Theorem Appendix F.5. *Finding an attack on BGP, that maximizes the traffic volume that goes through that node, is NP-hard.*

Theorem Appendix F.6. *Finding an attack on BGP that approximates the optimal (traffic-volume-maximizing) attack within a constant factor C, is NP-hard for any constant C.*

Proof (*Sketch*). The proofs of both theorems follows the outline presented in Section Appendix F.1. Hence, the main ingredient of the proof is showing the existence of a DILEMMA construction. We shall now present the DILEMMA construction; here, the manipulator's dilemma will be to decide *which path* should be announced to *which neighbors*. His strategy will be similar to the "false loop prefix hijack" of Section 6.3.

The DILEMMA construction. Consider the network in Fig. F.27, called "BAT-FROM-HELL-I". m is the node that wishes to attract as much traffic as possible for the victim prefix, while all other nodes behave normally. The network is such that

1. each of the nodes c_u and c_v has a large number of (direct and indirect) customers k in the subtree below it that can only reach d through it. Let k be big enough so that m be much more concerned with attracting c_u and/or c_v than with attracting all other nodes in the drawing;

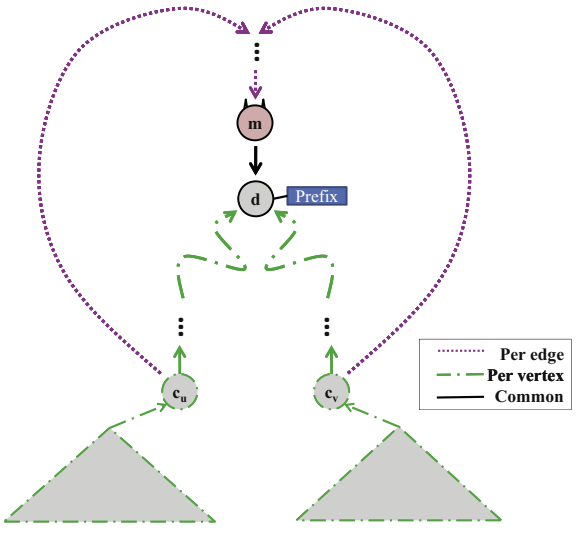


Fig. F.26. DILEMMA for proving hardness.

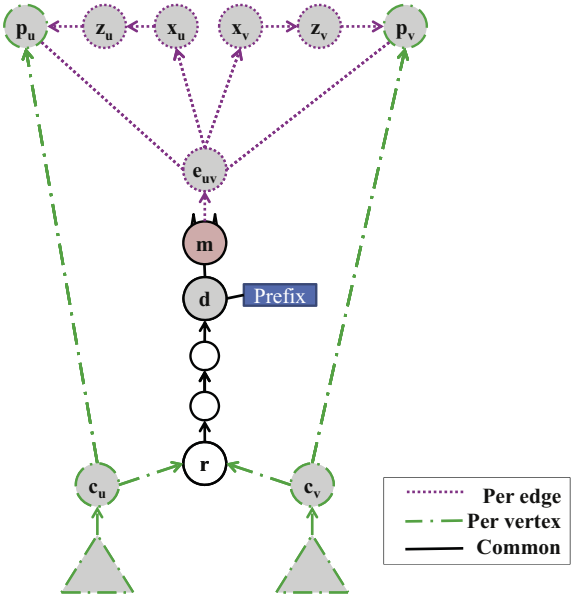


Fig. F.27. BAT-FROM-HELL-I.

2. p_u and p_v have lower AS numbers than r. Hence, if faced with a choice between the 4-hop route to d through r and a (false) 4-hop path to the prefix that has either p_u or p_v as next-hops, both c_u and c_v would prefer the latter route.

We now show that while m can attract c_u's traffic, or c_v's traffic, it cannot attract both nodes' traffic simultaneously. To see why this is true, consider node m's options. Observe that for m to attract c_u's (and its customers) traffic, it is necessary that c_u be offered a route of length 4 or less by p_u (because c_u already has an available route of length 4 through r). Recall that nodes prefer customer routes over peer routes (by **LP**) and so p_u prefers routes in which z_u is its next-hop node over routes in which the next-hop node is e_{uv}. Recall that when faced with two customer routes, they prioritize shorter routes (by **SP**). Unfortunately, observe that, no matter what m does, any route from p_u to m that has z_u as a next hop cannot be of length less than 4 (in fact, this is the case even if m hijacks d's prefix and announces it to e_{uv}). Hence, if p_u routes through z_u then c_u's available route through p_u shall consist of at least 5 hops and therefore will not be chosen by c_u.

How can m prevent p_u from routing through z_u? The easiest way is, of course, simply not to announce a route to e_{uv}. However, this will also mean that p_u will not learn of any route that goes through m. To avoid this, m must use a "false loop prefix hijack" strategy as in Section 6.3. He will announce a route to e_{uv} that contains one of the nodes x_u or z_u. By doing so m can ensure that one of these nodes shall not propagate this route further because of BGP's loop detection mechanism, and that p_u still have a loop-free route through m that is announced to it directly by e_{uv}. For example, if m announces mz_ud to e_{uv} then p_u learns the route $e_{uv}mz_u$ from e_{uv} and no route from z_u. Therefore, p_u shall make the route $p_ue_{uv}mz_u$ available to c_u, which, in turn, will choose this 4-hop route. Thus, m can attract c_u's traffic. Similarly, m can attract c_v's traffic by announcing the route mz_v to e_{uv}.

Can m attract both c_u and c_v at the same time? The answer is NO. Recall that to attract c_u m must include one of the nodes in the set $\{x_u, z_u\}$ in its announces route. Similarly, to attract c_v m must include one of the nodes in the set $\{x_u, z_u\}$. However, if m's announced route contains at least one node from each of these sets, and d, then p_u's route must be of length at least 4 and so both c_u and c_v shall not have a 4-hop route through p_u. This will result in both c_u and c_v choosing to forward traffic to r.

The reduction. We prove the correctness of the above two theorems via the arguments in Section F.1. We reduce from MAX-INDEP-SET. For every vertex $v \in V$ we create a node c_v. For every edge $e = (u, v) \in E$, we construct a BAT-FROM-HELL-I gadget to ensure that m not be able to attract both c_u and c_v simultaneously. Fig. F.27 describes the construction of BAT-FROM-HELL-I for the edge (u, v) (illustrating the per-vertex, per-edge, and common to all gadgets, parts of the construction). Observe that our constructions of BAT-FROM-HELL-I gadgets are consistent, in the sense that if the manipulator cannot attract node c_v in one such gadget (because it chose to attract the other node in that gadget), then it also cannot attract c_v in all other BAT-FROM-HELL-I gadgets that c_v participates in. The arguments in Section F.1 now imply the theorems. □

Extending to origin authentication and soBGP. The proof strategy above can easily be extended to attacks on origin authentication by adding more nodes and edges to the BAT-FROM-HELL-I. The modified BAT-FROM-HELL-I construction adds an extra node between r and d in Fig. F.27, and extra node y_i between nodes x_i and z_i with edges from y_i to nodes m and d. Then we use a similar argument as above to obtain the theorem.

F.3. It's still hard, even if the manipulator must announce normal paths

The reader might suspect that the computational task shall become much easier if the manipulator is severely constrained by security mechanisms (and hence the space of feasible attacks it must consider is significantly smaller). Surprisingly, this is not the case. We show that the above results hold even if the security mechanism (*i.e.*, S-BGP/BGPSEC/data-plane verification) forces the manipulator to announce his *normal path*! We show that finding the optimal attack is computationally hard even if the only decision the manipulator makes is *whether or not to export its normal path* (and thus, the path it actually uses).

Theorem Appendix F.7. *Even if the manipulator may only announce the normal path, finding an attack that maximizes the traffic volume through the manipulator is NP-hard.*

Theorem Appendix F.8. *Even if the manipulator may only announce the normal path, finding an attack that approximates the optimal (traffic-volume-maximizing) attack within a constant factor C, is NP-hard for any constant C.*

Proof (*Sketch*). The proofs of both theorems follows the outline presented in Section F.1. Hence, the main ingredient of the proof is showing the existence of a DILEMMA construction. We shall now present such a construction.

The DILEMMA construction. Consider the network in Fig. F.27, called "BAT-FROM-HELL-II". m is the node that wishes to attract as much traffic as possible, while all other nodes are behaving normally. The network is such that

1. each of the nodes c_u and c_v has a large number of (direct and indirect) customers k in the subtree below it that can only reach d through it. Let k be big enough so that m be much more concerned with attracting c_u and/or c_v than with attracting all other nodes in the drawing;
2. p_u and p_v have lower AS numbers than r. Hence, if faced with a choice between the 3-hop route to d through r and a 3-hop route to d that has either p_u or p_v as next-hops, both c_u and c_v would prefer the latter route.

We now show that while m can attract c_u's traffic, or c_v's traffic, it cannot attract both nodes' traffic simultaneously. To see why this is true, consider node m's options. m is forced to announce its normal path to d, md. Hence,

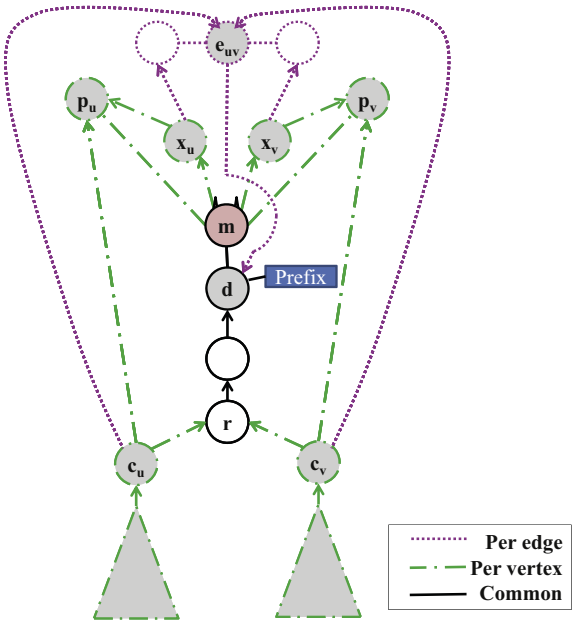


Fig. F.28. BAT-FROM-HELL-II.

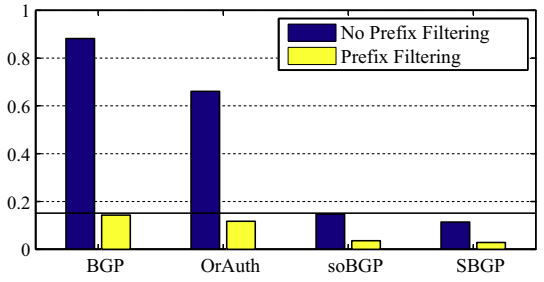


Fig. G.29. Lower bounds on the probability of attracting at least 10% of ASes in the internetwork. UCLA + IXP dataset.

m's only decision is to which neighboring nodes to announce the route md. Observe that if m announces md to x_u, then p_u will choose the customer route $p_u x_u md$ over the peer route $p_u md$ (by LP). This will result in c_u choosing the 3-hop route through r over the 4-hop route through p_u. Similarly, if m announces md to p_v this will result in the loss of c_v's traffic. Therefore, to attract c_u's traffic it is necessary that m not announce a route to x_u and, similarly, to attract c_u's traffic it is necessary that m not announce a route to x_u. Observe that if m does not announce md to both x_u and x_v then the edge e_{uv} shall be forced to choose its only available (provider-learned) route to $d, e_{uv}d$. In this case, both c_u and c_v will have a v-hop route to d through e_{uv} (and will choose it by SP). This will result in m's loss of both c_u's and c_v's traffic.

The above shows that while m can easily attract c_u's traffic alone (by not announcing md to x_u and announcing md to all other neighbors), or c_v's traffic alone (by not announcing md to x_v and announcing md to all other neighbors), it cannot attract both c_u and c_v's traffic simultaneously.

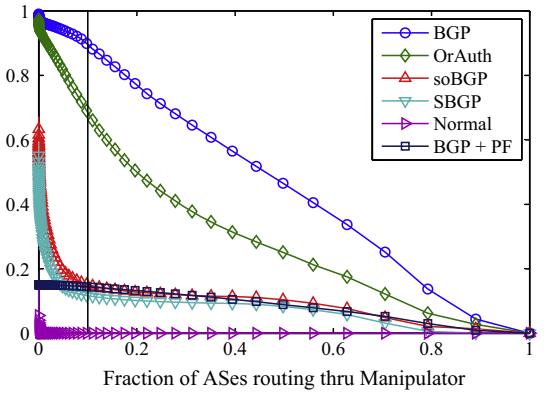


Fig. G.30. CCDF for the "Shortest-Path Export-All" attack strategy. UCLA + IXP dataset.

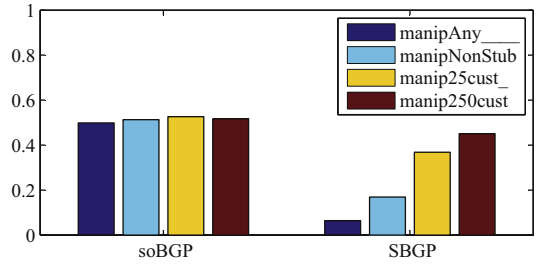


Fig. G.31. Probability of finding a shorter path. UCLA + IXP dataset.

The reduction. We prove the correctness of the above two theorems via the arguments in Section F.1. We reduce from MAX-INDEP-SET. For every vertex $v \in V$ we create a node c_v. For every edge $e = (u, v) \in E$, we construct a BAT-FROM-HELL-II gadget to ensure that m not be able to attract both c_u and c_v simultaneously. Fig. F.28 describes the construction of BAT-FROM-HELL-II for the edge (u, v) (illustrating the per-vertex, per-edge, and common to all gadgets, parts of the construction). Observe that our constructions of BAT-FROM-HELL-II gadgets are consistent, in the sense that if the manipulator cannot attract node c_v in one such gadget (because it chose to attract the other node in that gadget), then it also cannot attract c_v in all other BAT-FROM-HELL-II gadgets that c_v participates in. The arguments in Section F.1 now imply the theorems. □

F.4. Two remarks

Attraction *vs.* Interception. While our results are stated for attraction attacks (as they only discuss the amount of traffic that the manipulator can attract), the fact that in all of our DILEMMA constructions the manipulator is directly connected to d, and so always has a route available, implies that all of our hardness results extend to interception attacks.

The degree of the manipulator. Our hardness results are in the number of edges that the manipulator has (that is roughly the size of V in the MAX-INDEP-SET instance). However, the result in Section F.2 can easily be made to

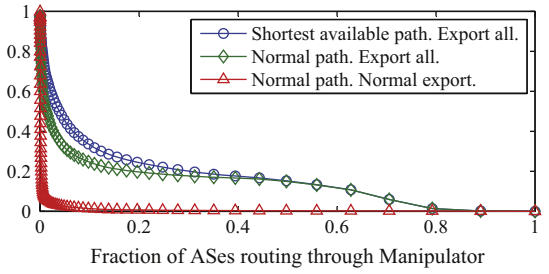


Fig. G.32. Aggressive export policies. UCLA + IXP dataset.

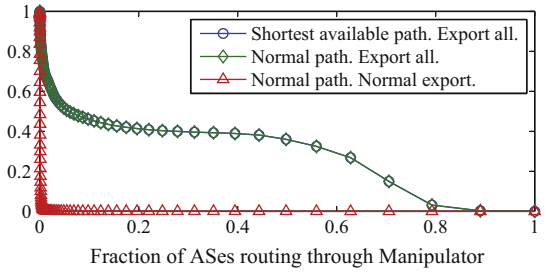


Fig. G.33. Aggressive export policies when the normal path is through a provider. UCLA + IXP dataset.

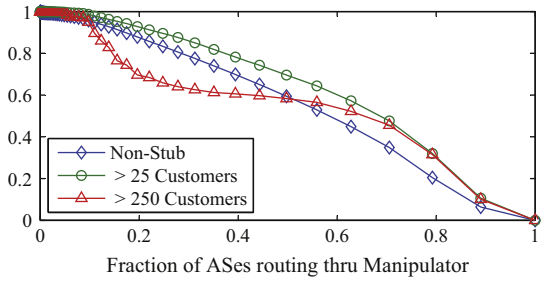


Fig. G.34. "Shortest-Path Export-All" attack strategy on BGP by different manipulators. UCLA + IXP dataset.

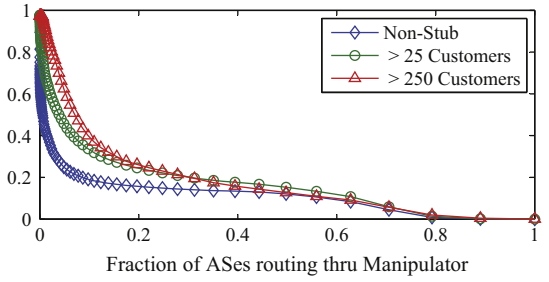


Fig. G.35. "Shortest-Path Export-All" attack strategy on S-BGP/data-plane verification by different manipulators. UCLA + IXP dataset.

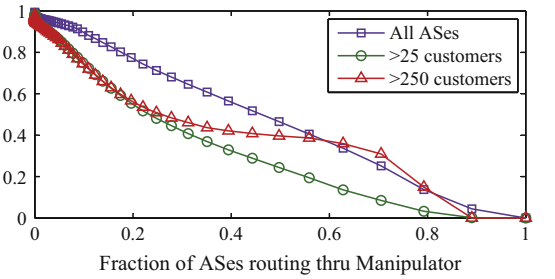


Fig. G.36. "Shortest-Path Export-All" attack strategy on BGP for different victims.

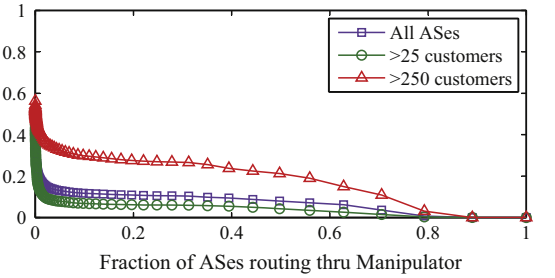


Fig. G.37. "Shortest-Path Export-All" attack strategy on S-BGP for different victims.

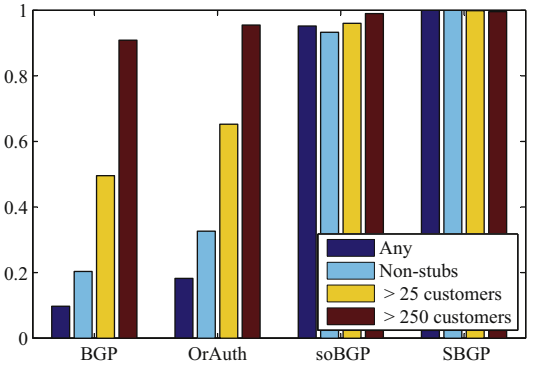


Fig. G.38. Probability that the "Shortest-Path Export-All" attack strategy does *not* create a blackhole. UCLA + IXP dataset.

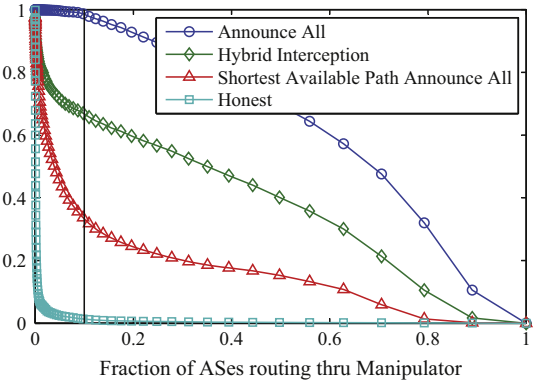


Fig. G.39. Interception attacks on BGP. UCLA + IXP dataset.

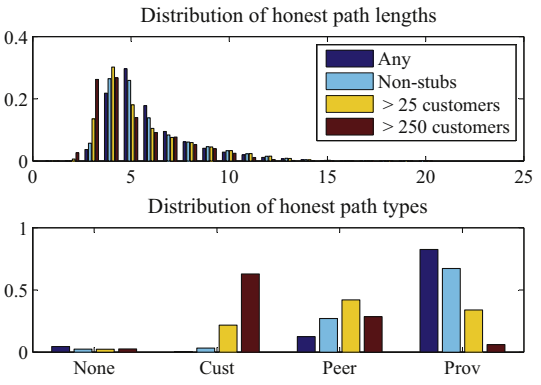


Fig. G.40. Path length and type distributions. UCLA + IXP dataset.

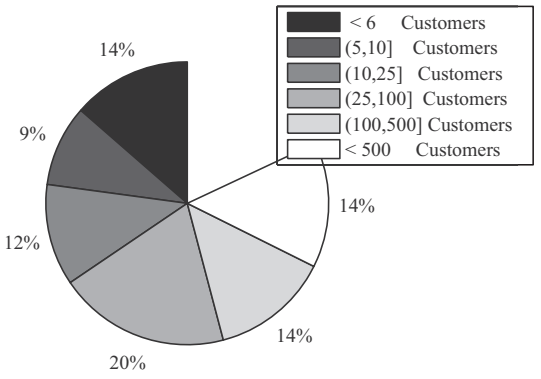


Fig. G.41. Distribution of stubs, according to the size of their smallest provider.

hold even if the manipulator only has a constant (even 2) number of neighbors. This can be achieved via the addition of intermediate nodes. In contrast, our result in Section F.3, where the manipulator only chooses whether to announce its actual path to each neighbor, is computationally easy if the manipulator has a constant number of neighbors (as it can simply go over all the possibilities).

Appendix G. Robustness: UCLA + IXP dataset

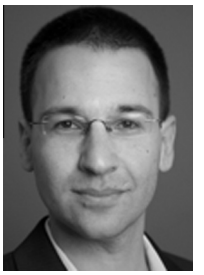
This appendix presents versions of all the graphs in this paper, computed from the 'Cyclop + IXP' AS Graph datasets [8,1]. We constructed this dataset from the November 20, 2009 UCLA dataset, by removing 276 edges connected to 4-byte ASNs, and removing 444 edges with unclassified business relationships. Then, we augmented the dataset with 21890 peer-to-peer edges from the recent IXP dataset [1], using only edges with good confidence, and ignoring edges that referred to ASes that were not in the UCLA dataset. We note that the UCLA dataset does *not* include any sibling-to-sibling edges, and is also derived using a different relationship inference algorithm than the CAIDA dataset (see G.29,G.30,G.31,G.32,G.33,G.34,G.35,G.36,G.37, G.38,G.39,G.40,G.41).

References

[1] B. Augustin, B. Krishnamurthy, W. Willinger, IXPs: mapped? in: IMC'09, 2009.
[2] I. Avramopoulos, J. Rexford, Stealth probing: data-plane security for IP routing, USENIX, 2006.
[3] H. Ballani, P. Francis, X. Zhang, A study of prefix hijacking and interception in the Internet, in: SIGCOMM'07, 2007.
[4] L. Blunk, A roa status attribute for rpsl objects, 2013. Internet-Draft: <http://tools.ietf.org/html/draft-blunk-rpsl-roa-00>.
[5] M.A. Brown, Rensys Blog: Pakistan hijacks YouTube, 2008. <http://www.renesys.com/blog/2008/02/pakistan_hijacks_youtube_1.shtml>.
[6] K. Butler, T.R. Farley, P. McDaniel, J. Rexford, A survey of BGP security issues and solutions, Proc. IEEE 98 (1) (2010) 100–122.
[7] H. Chang, D. Dash, A. Perrig, H. Zhang, Modeling adoptability of secure BGP protocol, in: SIGCOMM'06, 2006.
[8] Y.-J. Chi, R. Oliveira, L. Zhang, Cyclops: the as-level connectivity observatory, ACM SIGCOMM Comp. Comm. Rev. 38 (5) (2008) 5–16.
[9] J. Cowie. Rensys blog: China's 18-minute mystery. <http://www.renesys.com/blog/2010/11/chinas-18-minute-mystery.shtml>.
[10] A. de Beaupre, ISC Diary: Multiple Banking Addresses Hijacked, 2003. <http://isc.sans.edu/diary/BGP+multiple+banking+addresses+hijacked/16249>.
[11] B. Dickson, Route Leaks – Definitions, Internet Engineering Task Force SIDR Group, 2013. <http://tools.ietf.org/id/draft-dickson-sidr-route-leak-def-03.txt>.
[12] X. Dimitropoulos, D. Krioukov, M. Fomenkov, B. Huffaker, Y. Hyun, K.C. Claffy, AS relationships: inference and validation, SIGCOMM Comp. Comm. Rev., January 2007.
[13] J. Feigenbaum, V. Ramachandran, M. Schapira, Incentive-compatible interdomain routing, in: Conference on Electronic Commerce, 2006, pp. 130–139.
[14] J. Feigenbaum, M. Schapira, S. Shenker, Algorithmic Game Theory, Cambridge University Press, 2007. Chapter: Distributed Algorithmic Mechanism Design.
[15] L. Gao, On inferring autonomous system relationships in the Internet, IEEE/ACM Trans. Netw. 9 (6) (2001) 733–745.
[16] L. Gao, T. Griffin, J. Rexford, Inherently safe backup routing with BGP, IEEE INFOCOM, 2001.
[17] L. Gao, J. Rexford, Stable internet routing without global coordination, IEEE/ACM Trans. Netw. (TON) 9 (6) (2001) 681–692.
[18] P. Gill, S. Goldberg, M. Schapira, A survey of interdomain routing policies, NANOG'56, October 2012.
[19] P. Gill, M. Schapira, S. Goldberg, Let the market drive deployment: a strategy for transistioning to BGP security, SIGCOMM'11, 2011.
[20] P. Gill, M. Schapira, S. Goldberg, Modeling on quicksand: dealing with the scarcity of ground truth in interdomain routing data, SIGCOMM Comp. Comm. Rev. 42 (1) (2012) 40–46.
[21] S. Goldberg, S. Halevi, A.D. Jaggard, V. Ramachandran, R.N. Wright, Rationality and traffic attraction: incentives for honest path announcements in BGP, in: SIGCOMM'08, 2008.
[22] S. Goldberg, Z. Liu, The diffusion of networking technologies, in: SODA'13, 2013.
[23] S. Goldberg, M. Schapira, P. Hummon, J. Rexford, How secure are secure interdomain routing protocols? in: SIGCOMM'10, 2010.
[24] S. Goldberg, D. Xiao, E. Tromer, B. Barak, J. Rexford, Path quality monitoring in the presence of adversaries, in: SIGMETRICS, June 2008.
[25] T.G. Griffin, F.B. Shepherd, G. Wilfong, The stable paths problem and interdomain routing, IEEE/ACM Trans. Netw. (ToN) 10 (2) (2002) 232–243.
[26] J. Hastad, Clique is hard to approximate to within $n^{1-\epsilon}$, Acta Math. 182 (1999).
[27] G. Huston, Interconnection, peering, and settlements, in: Internet Global Summit, 1999.
[28] G. Huston, Peering and settlements – Part I, Internet Protocol J. (Cisco) 2 (1) (1999).
[29] G. Huston, Peering and settlements – Part II, Internet Protocol J. (Cisco) 2 (2) (1999).
[30] J. Karlin, S. Forrest, J. Rexford, Autonomous security for autonomous systems, Comput. Netw. (2008).
[31] E. Katz-Bassett, D.R. Choffnes, I. Cunha, C. Scott, T. Anderson, A. Krishnamurthy, Machiavellian routing: improving internet availability with bgp poisoning, in: HotNets-X, ACM, 2011.
[32] S. Kent, C. Lynn, K. Seo, Secure border gateway protocol (s-bgp), IEEE J. Select. Areas Commun. 18 (4) (2000) 582–592.
[33] M. Lad, D. Massey, D. Pei, Y. Wu, B. Zhang, L. Zhang, PHAS: a prefix hijack alert system, in: Proc. USENIX Security Symposium, 2006.

[34] M. Lepinski, Bgpsec protocol specification: draft-ietf-sidr-bgpsec-protocol-06, Internet-Draft, 2012.
[35] M. Lepinski, S. Kent, RFC 6480: An Infrastructure to Support Secure Internet Routing, 2012. <http://tools.ietf.org/html/rfc6480>.
[36] H. Levin, M. Schapira, A. Zohar, Interdomain routing and games, in: ACM STOC, May 2008.
[37] Y. Liao, L. Gao, R. Guérin, Z.-L. Zhang, Safe interdomain routing under diverse commercial agreements, IEEE/ACM Trans. Netw. (TON) 18 (6) (2010) 1829–1840.
[38] R. Lychev, S. Goldberg, M. Schapira, Is the juice worth the squeeze? BGP security in partial deployment, in: SIGCOMM'13, 2013.
[39] P. McDaniel, W. Aiello, K. Butler, J. Ioannidis, Origin authentication in interdomain routing, Comput. Netw. 50 (16) (2006) 2953–2980.
[40] S. Misel, "Wow, AS7007! Merit NANOG Archive, April 1997. <http://www.merit.edu/mail.archives/nanog/1997-04/msg00340.html>.
[41] P. Mohapatra, J. Scudder, D. Ward, R. Bush, R. Austein, BGP Prefix Origin Validation. Internet Engineering Task Force Network Working Group, 2012. <http://tools.ietf.org/html/draft-ietf-sidr-pfx-validate-09>.
[42] J. Naous, M. Walfish, A. Nicolosi, D. Mazieres, M. Miller, A. Seehra, Verifying and enforcing network paths with icing, in: Proceedings of the Seventh Conference on emerging Networking EXperiments and Technologies, ACM, 2011, p. 30.
[43] P. Palse, Serving ROAs as RPSL route[6] Objects from the RIPE Database. RIPE Labs, June 2010. <https://labs.ripe.net/Members/Paul_P_/content-serving-roas-rpsl-route-objects>.
[44] T. Paseka, Cloudflare blog: why google went offline today, November 2012. <http://blog.cloudflare.com/why-google-went-offline-today-and-a-bit-about>.
[45] A. Pilosov, T. Kapela, Stealing the Internet: An Internet-scale man in the middle attack, DEFCON'16, 2008.
[46] Rensys Blog, Con-Ed steals the 'Net, 2006. <http://www.renesys.com/blog/2006/01/coned_steals_the_net.shtml>.
[47] R. Steenbergen, R. Volk, W. Kumari, L. Blunk, D. McPherson, Isp route filtering: responsibilities & technical challenges, in: NANOG'43, Brooklyn, NY, June 2008.
[48] D. Wendlandt, I. Avramopoulos, D.G. Andersen, J. Rexford, Don't secure routing protocols, secure data delivery, HotNets-V, 2006.
[49] R. White, Deployment considerations for secure origin BGP (soBGP). draft-white-sobgp-bgp-deployment-01.txt, June 2003, expired.
[50] E.L. Wong, P. Balasubramanian, L. Alvisi, M.G. Gouda, V. Shmatikov, Truth in advertising: lightweight verification of route integrity, in: PODC'07, 2007.

Sharon Goldberg is an Assistant Professor in the Department of Computer Science at Boston University. Her research focuses on finding practical solutions to problems in network security. She received her Ph.D. from Princeton University in 2009, and her B.A.Sc. from the University of Toronto in 2003, and has worked as a researcher at IBM, Cisco, and Microsoft, and a telecommunication engineer at Bell Canada and Hydro One Networks. She is a Sloan Fellow and her work has been awarded two IETF/IRTF Applied Networking Research Prizes.



Michael Schapira is a Senior Lecturer at the School of Computer Science and Engineering, Hebrew University of Jerusalem. His research draws inspiration from theory (algorithmics, distributed computing, economics) to design and analyze Internet protocols (for routing, congestion control, etc.). Prior to joining the Hebrew University, he worked at Google NYC and was a postdoctoral researcher at UC Berkeley, Yale University and Princeton University. He holds a B.Sc. in Mathematics and Computer Science (2004), a B.A. in Humanities (2004), and a Ph.D. in Computer Science (2008) from the Hebrew University. He is an Allon Fellow and a Microsoft Research Faculty Fellow.

Jennifer Rexford is the Gordon Y.S.Wu Professor of Engineering in the Computer Science department at Princeton University. Before joining Princeton in 2005, she worked for eight years at AT&T Labs–Research. She received her BSE degree in electrical engineering from Princeton University in 1991, and her PhD degrees in electrical engineering and computer science from the University of Michigan in 1996. She is co-author of the book "Web Protocols and Practice" (Addison-Wesley, May 2001). She served as the chair of ACM SIGCOMM from 2003 to 2007. She was the 2004 winner of ACM's Grace Murray Hopper Award for outstanding young computer professional.