Rationality and Traffic Attraction: Incentives for Honest Path Announcements in BGP

(Full version from July 20, 2009)

Sharon Goldberg[*] Princeton University

Shai Halevi IBM Research

Aaron D. Jaggard[†] Rutgers University

Vijay Ramachandran[‡] Colgate University

Rebecca N. Wright[§] Rutgers University

1. INTRODUCTION

Interdomain routing on the Internet consists of a *control plane*, where Autonomous Systems (ASes) discover and establish paths, and a *data plane*, where they actually forward packets along these paths. The control-plane protocol used in the Internet today is the Border Gateway Protocol (BGP) [37]. BGP is a path-vector protocol in which ASes discover paths through the Internet via announcements from neighboring ASes. In BGP, each AS has routing policies that may depend arbitrarily on commercial, performance, or other considerations. These policies guide the AS's behavior as it learns paths from its neighbors, chooses which (if any) neighbor it will forward traffic to in the data plane, and announces path information to its neighbors. The design of BGP seems to encourage ASes to rely on path announcement as an accurate indication for the paths that data-plane traffic follows. However, BGP does not include any mechanism to enforce that these announcements match actual forwarding paths in the data plane.

Traditional work on securing interdomain routing (*e.g.,* Secure BGP (S-BGP) [27] and the like [6,21,42]) has focused on the control plane, with the loosely-stated goal of ensuring "correct operation of BGP" [27]. However, addressing the control plane in isolation ignores the important issue of how packets are actually forwarded in the data plane. Here, we explicitly focus on the security goal of ensuring that the paths announced in the control plane match the AS-level forwarding paths that are used in the data plane; this has been implicit in many previous works (on securing BGP [21, 27, 42] and incentives and BGP [9–13, 30, 35]). This way, an AS can rely on BGP messages, *e.g.,* to choose a high-performance AS path for its traffic or to avoid ASes that it perceives to be unreliable or adversarial [3, 24, 36].

This goal has recently received some attention by works [1, 31, 38, 43] that suggest auxiliary enforcement protocols that operate in the data plane. However, because such solutions typically incur a high overhead (see Section 1.1), here we consider solutions that operate in the control plane alone. Furthermore, most works on BGP security assume ASes can be arbitrarily malicious. Here, we instead follow a different line of research where ASes are modeled as *rational*, *i.e.,* act in a self-interested manner. In our work, we define this to mean that ASes both (1) try to obtain the best possible outgoing path for their traffic, while (2) also attracting incoming traffic (see Section 1.3). We look for conditions under which rational ASes have *no incentive to lie* about their forwarding paths in their BGP path announcements. We find that protocols like S-BGP [27] are generally *not* sufficient to prove that ASes have no incentive to lie about forwarding paths; we also require unrealistically strong assumptions on the routing policies of *every* AS in the network. Our results emphasize the high cost of ensuring that control- and data-plane paths match, even if we assume that ASes are rational (self-interested), rather than arbitrarily malicious.[1]

In the rest of this section, we motivate our approach, discuss related work, outline our results and discuss their implications. The model we use is defined in Sections 2–3, and our results are detailed in Sections 4–6. Related work is discussed further in Section 7. Proofs and additional discussion can be found in the appendices.

1.1 Matching the control and data planes.

One way to enforce honest path announcements in BGP is to deploy AS-path measurement and enforcement protocols that run in the data plane. However, determining AS-level paths in the data plane is a nontrivial task even in the absence of adversarial behavior (*e.g.,* [32] discusses the difficulty of determining AS-level paths from traceroute data). When dealing with ASes that may have incentives to announce misleading paths in the control plane, we need AS-path enforcement protocols that cannot be "gamed" (*e.g.,* by ASes that send measurement packets over the path ad-

This extended version is available as Princeton University Department of Computer Science Technical Report TR–823–08.

[1] We do not consider situations when the control and data plane do not match due to malfunction or misconfiguration; we consider this *irrational* behavior. We also do not consider control- and data-plane mismatches caused by path aggregation [32], since typically only last hop of the (data-plane) AS-path is omitted from the BGP path announcement.

vertised in the control plane, while sending regular traffic over a different path). Thus, data-plane enforcement protocols [1, 31, 34, 43] must ensure that measurement packets are indistinguishable from regular traffic, resulting in high overheads that are usually proportional to the amount of traffic sent in the data plane. Also, while secure *end-to-end* data-plane protocols can robustly monitor performance and reachability, *e.g.,* [2, 20], these protocols do *not* trace the identities of the ASes on a data-plane path; securely tracing AS paths requires participation of every AS on the path [1, 31, 34, 43].

Alternatively, one could hope to ensure that control- and data-plane paths match by ubiquitously deploying S-BGP [27] and the like [6]. This provides a property called path verification [30], which ensures that no AS can announce a path to its neighbors unless that path was announced to it by one of its neighbors. While path verification defends against announcement of paths that do not exist in the Internet topology [27], it does not, by itself, ensure that control- and data-plane paths match. For example, an AS a with two different paths announced by two different neighbors can easily lie in its path announcements—announcing one path in the control plane, while sending traffic over the other path in the data plane.

While it is tempting to argue that ASes are unlikely to lie about their forwarding paths because they either fear getting caught or creating routing loops, this argument fails in many situations. The hierarchy in the Internet topology itself often prevents routing loops from forming, *e.g.,* if the lie is told to a stub AS, or see also [4]. (We analyze the effect of lies on forwarding loops in Appendix A.) Furthermore, empirical results indicate that catching lies can be difficult, because even tracing AS-level paths that packets traverse in the data plane is prone to error [32]. Finally, to minimize the likelihood of getting caught, an AS could lie only when it has a good idea about where its announcements will propagate.

1.2 The game-theoretic approach.

In this work we explore the extent to which we can use *only control-plane mechanisms*, in conjunction with assumptions on AS policies, to motivate ASes to honestly announce data-plane paths in their BGP messages. Our exploration is carried out within the context of distributed algorithmic mechanism design [10, 33], which is rooted in game theory. This paradigm asserts that ASes are *rational players* that they participate in interdomain routing because they derive utility from establishing paths and forwarding packets; ASes will do whatever they can to maximize their own utility. The task of mechanism design is to ensure that the incentives of rational players are aligned with accomplishing the task at hand, so players have no incentive to deviate from the prescribed behavior.

The paradigm of algorithmic mechanism design in the context of routing was first suggested by Nisan and Ronen [33]. Feigenbaum *et al.* [10] brought *distributed* algorithmic mechanism design to the study of incentives in routing and shifted the focus to *interdomain* routing and BGP in particular. Rather than a centralized mechanism that sets up paths, the model in [10] postulates that paths are set up in a distributed fashion by the economically interested ASes themselves. The model was further developed in a sequence of works [7, 9–13, 30, 35]. Our model builds upon the work of Levin, Schapira, and Zohar [30], who brought a fully formal game-theoretic and distributed-computational model to this line of research (Section 2 and Appendix B). When the prescribed behavior includes the requirement that ASes honestly announcing forwarding paths to their neighbors (as is the case in all prior work), and when every AS follows this behavior, then the control plane and the data plane will match. In this sense, all work within this paradigm implicitly addressed matching the control and data planes. In this work, we highlight this matching (which is strictly weaker than the goal in prior work) as a stand-alone security property that should be addressed on its own.

1.3 Modeling utility with traffic attraction.

Recent work of Levin *et al.* [30] shows that if ASes are rational, then path verification (*e.g.,* S-BGP) is sufficient for honest path announcements, even when ASes have arbitrary routing policies. This encouraging result improved on earlier work [9–13] that explored restricted classes of routing policies. For example, Feigenbaum *et al.* [11, 13] found that it is sufficient to require policy consistency, a generalization of shortest-path routing and next-hop policy that requires that the preferences of neighboring ASes regarding different paths always agree. However, these results [9–13, 30, 35] were obtained under the assumption that the utility an AS derives from interdomain routing *is entirely determined by the outgoing path that traffic takes to the destination.* In reality, however, the utility of an AS is likely to be influenced by many other factors. For example, the utility of a commercial ISP may increase when it carries more traffic from its customers [25], or a nefarious AS might want to attract traffic so it can eavesdrop, degrade performance, or tamper with packets [3, 24, 36].

Here, we use a more realistic utility model (see Section 2.3), focusing in particular on the effect of traffic attraction, where the utility of one AS increases when it transits *incoming* traffic from another AS. We consider three models of traffic attraction. In our first model, traffic-volume attractions, utility depends only the origin of the incoming traffic, but not on the path that it takes. This captures the notion that an AS may be interested in increasing the volume of its incoming traffic or that a nefarious AS might want to attract traffic from a victim AS, in order to, say, perform traffic analysis. Our second model, generic attractions, encompasses all forms of traffic attraction; the utility of an AS may depend on the path incoming traffic takes. Our third model, customer attractions, is more restrictive. This model assumes that utility increases only if an AS attracts traffic from a neighboring customer AS that *routes on the direct link* between them; this models the fact that service contracts in the Internet are typically made between pairs of neighboring ASes [25] (Section 3.3).

1.4 Overview of our results.

In this work, we want to argue that under some set of conditions, any utility that an AS can obtain by lying in BGP announcements could also be obtained with honest announcements. Unfortunately, we find that conditions from previous work do not suffice when we consider traffic attraction: neither path verification [30] nor policy consistency [11, 13] alone is sufficient. (See Figures 2, 3, and 5 for examples.) These disappointing results motivate our search for new combinations of conditions (on control-plane verification, routing policy and export rules) that ensure that ASes

Control-plane verification	Model of AS utility			
	No traffic attraction	Increase volume of incoming traffic (Section 4)	Attract customer traffic via direct link (Section 6)	Generic traffic attraction (Section 5)
None	Policy consistency Consistent export [11, 13]	Next-hop policy All-or-nothing export	No known restrictions suffice	
Loop			Policy consistency Gao-Rexford conditions	Next-hop policy All-or-nothing export
Path	Arbitrary [30]	Policy consistency Consistent export	Next-hop at attractees Consistent export	

Table 1: For each utility model and type of control-plane verification, the additional restrictions that ensure that ASes in a network with no dispute wheel have no incentive to dishonestly announce paths.

have an incentive to honestly announce paths.

In addition to path verification (*e.g.,* S-BGP), we introduce a weaker form of control-plane verification called loop verification (Section 5.3), which roughly captures the setting in which an AS is caught and punished if it falsely announces a routing loop. Loop verification can be thought of as a formalization of "the fear of getting caught," and it may be easier to deploy than path verification.

In addition to policy consistency, we also consider the more restrictive next-hop policy, which roughly requires ASes to select paths to a destination based only on the immediate neighbor that advertises the path (Section 3.2). We also consider the Gao-Rexford conditions [15] (Section 3.3). These conditions, which are believed to reflect the economic landscape of the Internet [25], assume routing policies are restricted by business relationships between neighboring ASes, *i.e.,* by customer-provider relationships (the customer pays the provider for service) and peer-to-peer relationships (peer ASes transit each other's traffic for free).

Finally, we consider several classes of export rules (Section 3.4) that dictate whether or not an AS announces paths to its neighbors. An all-or-nothing export rule requires that, for each neighbor, an AS either announces every path or no paths. We also consider a more realistic consistent export rule [11] that roughly requires that ASes' export rules agree with their routing policies.

For many combinations of the conditions discussed above, we can still find examples in which ASes have an incentive to lie about their data-plane paths. However, for some combinations we obtain positive results, as sketched in Table 1. (These results all assume a network condition called "no dispute wheel" [22]; see Section 3.1.) Furthermore, our results are "tight", in that for every combination of the considered conditions, either one of our positive results applies or one of our negative examples does (as summarized in Tables 2–4).

Our positive results show that, for *every network* satisfying some combination of conditions, any utility an AS gains by lying can equivalently be obtained if that AS had instead *honestly announced paths to only an subset of its neighbors* and announced no paths to all other neighbors. That is, we show the existence of an *export rule* for which each AS obtains its optimal utility. As in previous work [11, 13, 30], our positive results for traffic-volume attractions (Section 4) and customer attractions (Section 6.2) also explicitly define an optimal export rule. Our positive result for generic attractions (Section 5.4) shows that an optimal export *exists*, but does not explicitly state what it is (Section 5.5). We discuss the notions used for our positive results further in Appendix B.

1.5 Implications of our results.

Our results suggest that even with control-plane enforcement mechanisms, ASes may have incentive to lie in their BGP announcements, unless very strong restrictions are imposed on their policies. As sketched in Table 1, from the set of conditions we considered, we always need *every AS in the network* to obey (1) unrealistic restrictions on its preferences (such as next-hop policy) and (2) explicit restrictions on export rules. Most of our results also require (3) full deployment of either path or loop verification. Thus, our results point to a negative answer to the question that we set out to investigate—practically speaking, it is unlikely that we could use only control-plane mechanisms to remove the incentives for ASes to announce false paths in BGP.

This suggests a choice. We can either employ expensive data-plane path enforcement techniques [1,31,34,43] when it is absolutely necessary to ensure that packets are forwarded on AS-level paths that match an AS's routing policies, or dismiss this idea altogether and instead content ourselves with some weaker set of goals for interdomain routing. It is certainly possible to formulate weaker but meaningful security goals and show that certain control-plane mechanisms or data-plane protocols meet these goals. However, doing this invites the question: if we are not interested in ensuring that AS paths announced in BGP are really used in the data plane, then why use a path-vector protocol at all?

2. MODELING INCENTIVES AND BGP

We now present the formal model in support of our results in Sections 4–6. The model builds on the literature [10, 22, 30] and extends prior work by explicitly considering traffic attraction. (We also make more explicit distinctions between control- and data-plane actions.)

2.1 The AS graph.

An interdomain-routing system is modeled as a labeled, undirected graph called an AS graph (see Figure 1). For simplicity, each AS is modeled as a single node, and edges represent direct (physical) communication links between ASes. Adjacent nodes are called neighbors. We denote nodes by lowercase letters, typically a, b, c, d, m, and n. We follow [22] and assume the AS-graph topology does not change during execution of the protocol.

Because, in practice, BGP computes paths to each destination separately, we follow the literature [22] and assume that there is a unique *destination node* d to which all other nodes attempt to establish a path. (Thus, like most previous work, we ignore the issue of route aggregation [32].) We denote paths by uppercase letters, typically P, Q, and R.

2.2 The interdomain-routing game.

We extend the model of Levin *et al.* [30] that describes

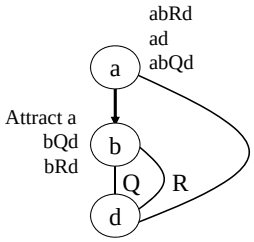


Figure 1: AS graph with traffic attraction.

interdomain routing as an infinite-round game in which the nodes of the AS graph are the strategic players. In each round, one node in the graph processes the most recent path announcements (if any) from its neighbors and then performs two *actions*: (1) it decides on an outgoing link (if any) to use in the data plane; and (2) decides on paths (if any) to announce to its neighbors.[2] Note that, just as in [30], nodes have the opportunity to announce their true data-plane path choice, but they are not forced to do so. The order in which nodes act is called the *schedule*.

We assume that path announcements sent between neighbors on direct links cannot be tampered with (by a node not on the direct link). This can be enforced via the BGP TTL Security Hack [17] or via a pairwise security association between nodes using the TCP MD5 security options [23]. We further assume that each node has the opportunity to act infinitely often—*i.e.,* the schedule is *fair*.

Game outcome and stability. The *state* of a node n at some round in the game consists of a data-plane component (the outgoing link most recently chosen by n) and a control-plane component (the announcements most recently sent by n). This state is *transient* if it occurs only finitely many times and it is *persistent* otherwise. There could be many possible sequences of states; the sequence depends on both the schedule and the actions of nodes while playing the game. When we ask whether or not there is an incentive to lie, we are interested in the more precise question: Is there a fair schedule in which a node may have an incentive, in some round, to announce a route in the control plane that is not its data-plane choice?

The global state at some round is the collection of all node states at that round. A global outcome of a game is a global state that does not contain any transient node states.We note that there could be more than one such global state; in particular, a persistent control-plane oscillation among nodes is a sequence that infinitely transitions among non-transient node states, even for a fixed schedule. Our results in this work hold regardless of which of these is taken to be the global outcome.

If the state of a node is constant after some round then this state is locally stable. A global outcome is globally stable if all node states in it are locally stable. (This definition of stability is compatible with the original definition in [22].) We typically denote global outcomes by T or M. We may use "outcome" informally to mean the control-plane or data-plane component of the outcome when the component is clear from the context.

2.3 Utility, valuation, and attraction.

[2]A node can also decide not to route on any link in the data plane, or not to announce anything to its neighbors.

A *strategy* is a procedure used by a node to determine its actions in the game. In principle, a node can make decisions in any way that it wants, but here we assume that nodes are *rational*. In particular, each node b has a utility function $u_b(\cdot)$ mapping outcomes to integers (or $-\infty$); b tries to act to obtain an outcome T that maximizes $u_b(T)$.

We assume that every node b in the graph has a utility function of the form

$$u_b(T) = v_b(T) + \alpha_b(T) \tag{1}$$

where $v_b(T)$ is the valuation function that depends only on the simple data-plane path from b to d in T, and $\alpha_b(T)$ is the attraction function that depends only on the simple data-plane paths from other nodes to b in T. (We write the utility function as a *sum* of the valuation and attraction functions; in fact, our results require only that utility increases monotonically with each of the valuation and attraction functions.) In this work, utility depends on the data-plane component of outcome alone (because the control-plane component may not correspond to actual traffic flow in the network).

The valuation function $v_b(\cdot)$ is the same as was considered in previous work on incentives and BGP [7,9–13,30,35]. It is meant to capture the intrinsic value of each outgoing path (*e.g.,* as related to the cost of sending traffic on this path, its reliability, the presence of undesirable ASes on it, *etc.*). We assume that nodes dislike disconnection, so that if node b has no data-plane path to the destination in outcome T, then $v_b(T) = -\infty$. (The implications of this are discussed further in Section 2.7.)

The attraction function $\alpha_b(T)$ is the new component of utility that we add in this work. Because we are interested in situations where nodes may want to attract traffic (and not deflect it), our most general form of the attraction function only requires that $\alpha_b(\cdot)$ does not increase when edges leading to b are removed from the data-plane outcome. Formally, for an outcome T and node b, let $T(b)$ be the set of edges along simple paths from other nodes to b in the data-plane component of T (*e.g.,* if T's data-plane links form a routing tree, then $T(b)$ is the subtree rooted at b). We assume that for every two outcomes T and T' and every node b, if $T'(b) \subseteq T(b)$, then $\alpha_b(T') \leq \alpha_b(T)$. This general condition covers many forms of traffic attraction; *e.g.,* attraction can depend on which links are traversed by incoming traffic at a node, and not just the nodes from which that traffic originates.

We also consider two specific forms of traffic attraction. First, traffic-volume attraction requires that $\alpha_b(T)$ depends only the origin of the incoming traffic, but not on the path that it takes. More formally, if $T(b)$ and $T'(b)$ include the same nodes then $\alpha_b(T) = \alpha_b(T')$. This also captures the idea of nefarious ASes who want to attract traffic for eavesdropping on or tampering with traffic (but see also Section 2.7).

Another specific form of attraction is customer attraction, in which the AS graph is assumed to have underlying business relationships, and $\alpha_b(T)$ depends only on customer nodes a that route through b on the direct a-b link between them. We further discuss this form of attraction and customer-provider relationships in Section 3.3.

We say that there is an *attraction relationship* between a and b if the attractor b increases its utility when the attractee a routes traffic through it (*e.g.,* as in Figure 1). In Figure 1, we depict the utility function of each node next to

that node: say that the attraction function of b is such that it earns 100 points of utility when it attracts traffic from a, and that the valuation function of b is such that it earns 10 points of utility when using the path bQd and only 1 point of utility when using the path bRd. Then, following Equation 1, the use of data-plane path $abRd$ earns b 101 points of utility.

2.4 BGP-compliant strategies.

Recall that we are interested in ensuring that the inter-domain-routing control and data planes match. When all nodes follow the rules prescribed by the BGP RFC [37] in their execution of the protocol, this is achieved. We call a strategy that obeys these rules a *BGP-compliant strategy*, as formalized below.

DEFINITION 2.1. A BGP-compliant strategy for node n depends on two functions: A ranking function $r_n(\cdot)$ mapping each path to an integer or $-\infty$; and, an export rule $e_n(\cdot)$ that maps each path P to the set of neighbors to which n is willing to announce the path P. A path P is admitted at n if $r_n(P) > -\infty$. Paths that include routing loops or that do not reach the destination are not admitted at any node. We require that, for any two paths P and Q admitted at n that begin with different next hops, it holds that $r_n(P) \neq r_n(Q)$. (Note that $r_n(\cdot)$ and $e_n(\cdot)$ act *only* on path announcements, rather than game outcomes (*e.g.,* data-plane paths).)

The strategy of node n is BGP-compliant, with $r_n(\cdot)$ and $e_n(\cdot)$ as defined above, if n does the following in each round in which it participates. Node n first chooses the path P such that (a) P has highest rank of all the most recently announced paths received from neighbors, and (b) the first node a of P is the neighbor that announced P to n. Then, n performs the following two actions: (1) n chooses the outgoing link to a in the data plane; and (2) n announces the path nP to all neighbors in $e_n(P)$.

This definition explicitly assumes that the all traffic to the destination is routed over a single next-hop. (We do not address here the question of modeling multipath routing.) Also, we assume that, if n does not receive any announcements with an admitted path, then n does not route on any outgoing link or announce any paths to its neighbors. (Notice that we model *ingress filtering* using the concept of admitted paths and *egress filtering* using the concept of an export rule.)

Control-plane announcements from a node executing a BGP-compliant strategy match its next-hop choices in the data-plane. Thus, if *all* nodes in the network use BGP-compliant strategies, then the control and data planes will match. (We may informally call a node executing a BGP-compliant strategy a *BGP-compliant node*, or sometimes an *honest node*.) In the positive results from previous work [11, 13, 30] included in Table 1, the prescribed strategies are examples of BGP-compliant strategies in the sense of Definition 2.1. Thus, those results also achieved agreement between the control and data planes, but contrary to the current work, they do not consider traffic attraction.

We stress that Definition 2.1 gives BGP-compliant nodes the leeway to choose their ranking and export functions in any way they want, in order to try to achieve a utility-maximizing outcome in the game. In the next subsection, we discuss the relationship between utility and the ranking and export functions in a way that encompasses earlier work (without traffic attraction) and the results in this work (with traffic attraction).

2.5 From utility to ranking and export.

To map between our model and real-world implementation of BGP [37], we can think of the actions of the game described in Definition 2.1 (*i.e.,* (1) selection of next-hop, and (2) announcements to neighbors) as being executed by nodes, in practice, through setting parameters in the ranking and export functions. In previous work [13, 30], the ranking function was set equal to the valuation function (we denote this as $r_n(\cdot) \equiv v_n(\cdot)$[3]: the larger the valuation of a path, the higher its rank. This follows from the fact that in previous work, the utility of an AS was defined to be its valuation function,[4] and thus the directly determined the ranking function. However, the direct translation from valuation to ranking does not always hold in our setting of traffic attraction: announcing an outgoing path with low valuation could be preferred because it brings incoming traffic from attractees. For example, in Figure 1, node b's valuation function ranks path bQd over path bRd; but, b has higher utility when it claims that it routes on bRd because it then attracts traffic from node a.

Although this direct translation does not always hold, we do assume that BGP-compliant ASes are able to "compile" their utility functions (which depend on both valuation and attraction as in Equation 1) into ranking and export functions that then consistently determine their actions in the game, *i.e.,* their behavior during the BGP protocol. This compilation might be viewed as transforming utilities into functions that act on path announcements by, *e.g.,* setting BGP local preference. We think of the compilation process as being done "once and for all," and we analyze the network with respect to fixed ranking and export functions. We note that this is not entirely realistic: the "compilation" can, in principle, model an ongoing process in which an AS reacts to changes in network conditions, contractual agreements, new information that ASes learn about each other, *etc.*, to better attempt to maximize its utility. However, the time scale for compilation is usually much longer than the time scale for BGP itself (say, hours versus seconds); so, a once-and-for-all modeling may still be reasonable. (See also Section 7.)

There are many conceivable ways of compiling the utility into ranking and export rules. In many cases, it makes sense to use the simple compilation $r_b(\cdot) \equiv v_b(\cdot)$ by default, and to use a different compilation only when this is advantageous in terms of traffic attraction; *e.g.,* if there is a service-level agreement that obliges b to carry a's traffic via path bRd in return for monetary compensation α, then b might decide to set $r_b(bRd) = v_b(bRd) + \alpha$. In general, we mostly sidestep the question of how to compile the utility into ranking and export policy. However, our counterexamples work for any ranking function "reasonably compiled" from the utility function, and our positive results all hold for the setting $r_b(\cdot) \equiv v_b(\cdot)$.

2.6 Incentives to lie.

Because nodes are rational (*i.e.,* acting to maximize their

[3]This is a slight abuse of notation, because r is formally defined on paths and v on outcomes. We ignore this formality from now on.

[4]Some previous work [9–12, 35] allowed utilities that depend on monetary transfers, which we do not consider here.

utility in the global outcome), they may have an incentive to follow a strategy that is not BGP-compliant. As discussed in Section 1.1, although an AS knows the outgoing link on which it forwards traffic (and the next AS at the end of that link), it may not know the AS-path that the traffic takes further downstream. For example, in Figure 1, node b could deviate from BGP-compliance by announcing the path bRd in order to attract traffic from node a, while actually sending traffic over the path bQd; as a result the control and data planes would not match, unbeknownst to a.

Hence, in this work, as in [11, 13, 30, 35], we address the following high-level question: Are there sufficient conditions on the network that ensure that all nodes are honest (*i.e.,* use BGP-compliant strategies)? The earlier work studied this question using the game-theoretic notion of "incentive compatibility." In contrast to some uses of this notion in earlier work (*e.g.,* Thm. 3.2 in [30]), our positive results give nodes some additional flexibility in choosing their strategies, as long as these strategies are BGP-compliant. (We discuss this difference in some detail in Appendix B.)

Ideally, we would like conditions that ensure that nodes have no incentive to be dishonest, no matter what the other nodes do. Unfortunately, it is extremely difficult to find such conditions; see [11,13,30,35]. Instead, we look for conditions that ensure that a node has no incentive to be dishonest *if it knows that everyone else is honest.* That is, we try to ensure that no node has an incentive to *unilaterally* deviate from using BGP-compliant strategies.

We discuss our technical formalizations after each of our positive results (Theorems 4.1, 5.1, and 6.1).

2.7 Additional remarks.

Modeling nefarious ASes. Our modeling assumes that $v_b(T) = -\infty$ implies $u_b(T) = -\infty$, so that nodes cannot derive any utility from outcomes in which they cannot reach the destination. Our negative examples do not depend on this assumption, but our positive results do. This means that our positive results do *not* hold if a manipulating node wants to attract traffic for nefarious purposes, like tampering or eavesdropping, *when it does not have a path to the destination.*

Single outgoing link. While we assume that all BGP-compliant ASes choose a single outgoing link for all their traffic, a misbehaving node m might send its outgoing traffic on more than one outgoing link. In this case, we assume that if m uses more than one path to d in T, then the valuation $v_m(T)$ is at most as high as the most valuable simple m-to-d path in the outcome T. This assumption was implicitly used in prior work, and it ensures that even for a manipulator m "the optimal strategy" is to send its outgoing traffic over a single link. This is because the valuation of the path cannot decrease if it uses only the "best outgoing link" instead of using a few of them, and the attraction function does not depend on the outgoing links that m uses.

Utility and outcomes. In this work we defined the utility function to depend on the data-plane component of outcome alone, because the control-plane component may not correspond to actual traffic flow in the network. However, this also means that an AS may be unaware of its actual utility (i.e., when its data-plane forwarding path differs from the control-plane path). An alternative approach would be to define the attraction function on the data-plane outcome and the valuation component on the control-plane outcome.

We note, however, that because in this work we consider only *unilateral* deviations (*i.e.,* the all nodes are honest except for a single manipulator), our results in this work hold just the same under this alternative approach. Since we suppose only one node can potentially deviate from honest behavior, we are assured that the data-plane forwarding path of the manipulator matches its control-plane path (since all the nodes on the manipulator's outgoing path must be honest), and so the manipulator utility can depend on either the control-plane or data-plane outcome.

3. DEFINITIONS: POLICY AND EXPORT

3.1 No dispute wheel.

Griffin, Shepherd, and Wilfong [22] described a global condition on the routing policies in the AS graph, called *"no dispute wheel,"* that ensures that BGP always converges to a unique stable outcome. Roughly, a dispute wheel is a set of nodes, each of which prefers to route through the others rather than directly to the destination. More formally, there is a dispute wheel in the valuations if there exist nodes $n_1, \ldots, n_t$ such that, for each node n_i, there exists a simple path Q_i from n_i to the destination d and a simple path R_i from n_i to n_{i+1} for which $v_{n_i}(R_iQ_{i+1}) > v_{n_i}(Q_i)$.[5] (The index i is taken modulo t.) A dispute-wheel in the ranking functions (for BGP-compliant nodes) is defined similarly with r_{n_i} replacing v_{n_i}. Following the literature [13, 30], we *always* consider networks with no dispute wheels in the valuations. The result of [22] in our terminology states that, if all nodes use BGP-compliant strategies with $r_n(\cdot) \equiv v_n(\cdot)$ and there is no dispute wheel in the valuations, then the game's outcome is unique and globally stable.

3.2 Policy consistency and next-hop policy.

Node a is *policy consistent* [11, 13] in valuations with one of its neighbors b if, whenever b prefers some path bPd over bRd (and neither path goes through a), then a prefers $abPd$ over $abRd$. Formally, for any two simple paths $abPd$ and $abRd$, if $v_b(bPd) \geq v_b(bRd)$, then $v_a(abPd) \geq v_a(abRd)$. We say that *policy consistency* holds for the problem instance if every node is policy consistent with each of its neighbors. (Policy consistency is a generalization of next-hop routing and shortest-path routing; see [11, 13].)

Next-hop policy requires that a node only care about the neighbor through which its traffic is routed and nothing else. This class of routing policies is more restrictive than policy consistency (*e.g.,* node c in Figure 3 is policy consistent but does *not* use next-hop policy with node m). Formally, a uses next-hop policy with b if for every two simple paths $abPd$ and $abRd$ it holds that $v_a(abPd) = v_a(abRd)$. Notice that if a uses next-hop policy with b then it must either admit all simple paths through b or (ingress) filter all of them (*cf.,* discussion in [8, 39]).

Similar definitions apply also to the ranking functions.

3.3 Gao-Rexford & customer attractions.

Gao and Rexford [15] described a set of conditions that are induced by business relationships between ASes [25]. In

[5]For readability, we somewhat abuse notation and use $v_n(P)$ to mean n's valuation of any outcome T in which its traffic uses the data-plane path P.

Gao-Rexford networks there are two kinds of edges: customer-provider edges (where typically the customer pays the provider for connectivity) and peer-to-peer edges (where two nodes agree to transit each other's traffic for free). A Gao-Rexford network obeys the following three conditions (GR1–GR3):

GR1. Topology. There are no customer-provider cycles in the AS graph, *i.e.,* no node is its own indirect customer.

GR2. Export. A node b only exports to node a paths through node c if at least one of nodes a and c are customers of node b.

GR3. Preferences. Nodes prefer outgoing paths where the next hop is a customer over outgoing paths where the next hop is a peer or a provider, and prefer peer links over provider links.[6]

GR3 always applies to the valuation functions of each node in a Gao-Rexford network, and can also apply to the ranking functions.

We also model customer attractions within the Gao-Rexford setting. Namely, we consider a fourth condition (AT4) that models the fact that service contracts in the Internet are made between pairs of neighboring nodes, where a customer pays its provider when it sends traffic over their shared link [25]. AT4 restricts the set of traffic attraction relationships that we allow in the AS graph, and thus does *not* model settings where, *e.g.,* an AS wants to attract traffic from ASes that are a few hops away.

AT4. Attractions. A node b may only have attraction relationships with its own customers. Furthermore, b only increases its utility if its attractee-customer a sends traffic over the direct a-b link.

When we draw Gao-Rexford networks, we represent a customer-provider relationship by a directed edge from customer to provider, and a peer-to-peer relationship by an undirected edge. We represent an AT4 attraction relationship with a **bold** arrow from attractee to attractor (*e.g.,* see Figure 2).

3.4 Export rules.

Our results about BGP-compliant strategies that achieve matching control and data planes in the setting of traffic attraction involve several types of export rules. The export-all rule (used, *e.g.,* in Thm. 3.2 of [30]) requires that a node exports all its admitted paths to all its neighbors. An all-or-nothing rule for a node n means that, for each neighbor a of n, either n exports all admitted paths to a or none at all. The consistent export rule [11] means that, if n exports to a neighbor a some path R, then it must also export every other path that is ranked at least as high as R; *i.e.,* if $r_n(Q) \geq r_n(R)$ and n exports R to a, then n must also export Q to a. Finally, in Gao-Rexford networks, the export rules used by BGP-compliant nodes satisfy GR2.

The export-all rule implies the all-or-nothing export rule, which in turn implies the consistent export rule. We emphasize that both the export-all and the all-or-nothing rules are often incompatible with the Gao-Rexford export condition GR2. As one example, the export-all rule may require an AS to export a path through one of its peers or providers to another one of its peers or providers, a violation of GR2.

[6]The original version [15] of the Gao-Rexford conditions does not require nodes to prefer peer links over provider links. To make our results as general as possible, we use this weaker version of GR3 in all our theorems, while our counterexamples do satisfy the stronger version of GR3.

3.5 Dispute wheels in Gao-Rexford networks.

As we discussed in Section 3.1, in this work we always consider AS-graphs with no dispute wheel in the *valuation* functions, *even if they obey the Gao-Rexford conditions.* Since in our model, export policy is part of the strategy from which nodes may deviate, we do not rely on GR2 to exclude paths from the valuation functions that may have caused dispute wheels; the valuation functions are only subject to GR1 and GR3. This is in contrast to other works on BGP convergence, *e.g.,* [14, 15], which relied on GR2 to remove dispute wheels, because they assumed that every node honestly follows the GR2 export rule. More generally, in the setting where nodes may deviate from (prescribed) BGP-compliant strategies in order to better their own utility, we cannot say that the Gao-Rexford conditions imply that the BGP protocol converges, as in [14, 15]. For example, it is possible to show a network in which a node unilaterally deviates from GR2 and thus causes the BGP protocol to oscillate forever. We discuss this further in Section 6.5.

4. RESULTS: VOLUME ATTRACTIONS

We start with some results for traffic-volume attractions, as defined in Section 2.3. We stress that this is a rather restricted form of traffic attraction, as it excludes the possibility of the utility depending on the path along which incoming traffic arrives. We begin with a series of counterexamples, demonstrating that even for this very restricted form of traffic attraction, ensuring that nodes have no incentive to lie is far from easy. (Most of our counterexamples are Gao-Rexford networks that obey GR1–GR3 and sometimes also AT4 from Section 3.3.) We then present a positive result (Section 4.3), showing two sets of conditions, each of which suffices to ensure that a node honestly announces paths. The results from this section are summarized in Table 2.

4.1 Path verification is not enough.

Path Verification is the focus of most traditional work on securing BGP [6]; roughly, it ensures that nodes cannot announce paths that are not in the network. More formally, path verification is a control-plane mechanism that ensures that every node a only announces a path abP to its neighbors if its neighbor b announced the path bP to a. Path verification can be guaranteed when S-BGP [27] or IRV [21] is *fully* deployed in the network. (We note, however, that soBGP [42] does *not* provide path verification; soBGP only provides information about AS-graph topology, and not about path announcements.)

For the setting of no traffic attraction, a recent result of Levin *et al.* [30] shows that, in a network with path verification and no dispute wheel, no node has an incentive to unilaterally deviate from a BGP-compliant strategy with $r_n(\cdot) \equiv v_n(\cdot)$ and an export-all rule. They also show (in [29]) that the same is true in Gao-Rexford networks, but with an export rule that exports all paths except those that would violate GR2. However, we show that when there are traffic-volume attractions, a node can have an incentive to make a

Verification?	Policy	Export	Incentive to Lie?	Result
$\star$	No restriction	$\star$	Yes	Inconsistent Policy
None / Loop	Consistent	$\star$	Yes	Nonexistent Path
Path / Loop	Next-hop	Inconsistent	Yes	Inconsistent Export
Path	Consistent	Consistent	No	Theorem 4.1
$\star$	Next-hop	All-or-nothing	No	Theorem 4.1

Table 2: Summary of our results for traffic-volume attractions. We also require no dispute wheel.

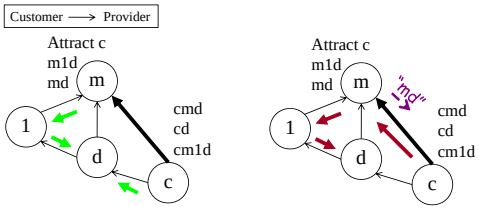


Figure 2: Inconsistent Policy

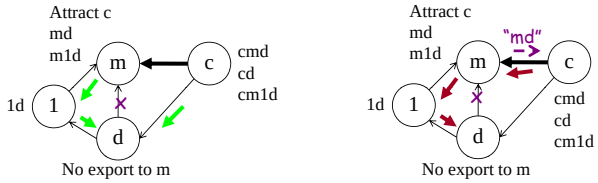


Figure 3: Nonexistent Path

dishonest announcement, even when the network has path verification:

Figure 2: Inconsistent Policy demonstrates that a policy inconsistency between a manipulator m and its customer c can give m an incentive to dishonestly announce its forwarding path in order to attract traffic from c. On the left we show the outcome T that results when each node n uses a BGP-compliant strategy with $r_n(\cdot) \equiv v_n(\cdot)$, exporting all paths except those that would violate GR2. On the right, we show the manipulated outcome M, in which only a single manipulator node m does not use a BGP-compliant strategy. Here, m has an incentive to announce the path md to node c, while actually using path $m1d$, in order to attract c's traffic. Notice that this announcement can be made even with path verification, because node 1 announced $1d$ to m. In the outcome M, node m gains not only a traffic-volume attraction (because c routes through m in M but not in T), but also an AT4 attraction (because c is a customer that routes on the direct c-m link in M). Note that Inconsistent Policy is a Gao-Rexford network with no dispute wheel that obeys AT4.

We remark that the situation in Inconsistent Policy could arise quite naturally in practice. As an example, while c is a customer of both m and d, the service contracts of c with m and d are such that usage-based billing on the m-c link is lower than billing on the d-c link. Then, c could prefer a path through m over the direct path to d as long as this path only increases AS-path length by a single hop. On the other hand, m could prefer to send traffic via 1 because 1 is, say, geographically closer to m than d.

4.2 Policy consistency alone is not enough.

Notice that, in Inconsistent Policy, node c is not policy consistent with node m (Section 3.2). It is natural to ask if requiring policy consistency is sufficient to ensure that there is no incentive to lie. Indeed, for the setting of no traffic attraction, Feigenbaum *et al.* [11, 13] proved that in a network with policy consistency and no dispute wheel, then no node has an incentive to *unilaterally* deviate from a BGP-compliant strategy with $r_n(\cdot) \equiv v_n(\cdot)$ and consistent export. Perhaps surprisingly, it turns out that policy consistency is not sufficient to ensure that nodes have no incentive to lie when we consider traffic-volume attractions:

Figure 3: Nonexistent Path demonstrates that, even in a policy consistent network, a manipulator m can have an incentive to announce a nonexistent path in order to attract traffic from its customer c. The outcome T, shown on the left, results when each node uses a BGP-compliant strategy with $r_n(\cdot) \equiv v_n(\cdot)$, where node d's export rule obeys consistent export but exports nothing to node m, and all other nodes export all paths allowed by GR2 (which implies consistent export). On the right, we show the manipulated outcome M, where only the manipulator m deviates from the BGP-compliant strategies described above. Here, the manipulator m has an incentive to announce to node c a false path "md" that *is not available to* m (because d does not export this path to m) in order to attract c's traffic. Again, node m gains both a traffic-volume attraction and an AT4 attraction in M that it could not have obtained by using a BGP-compliant strategy. Note that Nonexistent Path is a policy-consistent Gao-Rexford network with no dispute wheel that obeys AT4.

Notice that c has the same preferences in both Nonexistent Path and Inconsistent Policy. However, in Nonexistent Path, c is policy consistent with m; both prefer the nonexistent shorter path through md over the longer path through $m1d$.

4.3 But adding path verification or next-hop policy is enough!

In Nonexistent Path, the manipulator m announces a path "md" was that was not announced to it by d (which would not be possible if the network had path verification), and that announcement matters because node c does not use a next-hop policy with m. It turns out that requiring *either* path verification (on top of policy consistency) *or* next-hop policies is sufficient to ensure honesty in any network with only traffic-volume attraction functions. In these settings, if each node sets its ranking equal to its valuation and honestly exports *all* paths to all neighbors, then no node has an incentive to *unilaterally* deviate from this behavior.

Theorem 4.1. *Consider an AS graph with no dispute wheel in the valuations. Suppose that all nodes, except a single manipulator node m, use BGP-compliant strategies and set their ranking equal to their valuations ($r_n(\cdot) \equiv v_n(\cdot)$ for every node n). Suppose further that m has a traffic-volume attraction function, and that at least one of the following two conditions hold:*

a. The valuations function of all nodes are next-hop and

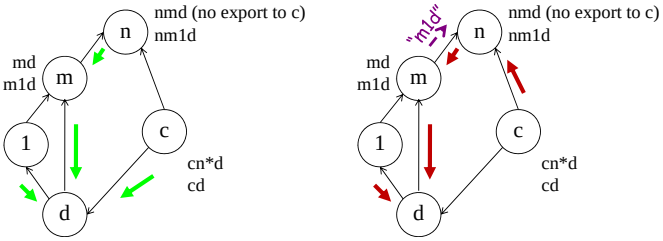


Figure 4: Inconsistent Export

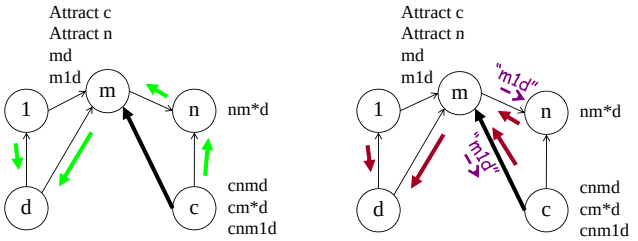


Figure 5: Bowtie

the export functions of all the nodes but m obey all-or-nothing export; or

b. The valuations function of all nodes are policy consistent, the export functions of all the nodes but m obey consistent export, and the network has path verification.

Then there is a BGP-compliant strategy for m that sets $r_m(\cdot) \equiv v_m(\cdot)$ and obeys all-or-nothing export (and therefore also consistent export), such that this strategy is optimal (utility-maximizing) for m. In particular, using the export-all rule is one such optimal strategy.

Notice that Theorem 4.1 not only establishes the *existence* of an optimal consistent export rule for m, but also asserts that export-all is one such optimal rule. Hence it actually establishes a single strategy from which no node has an incentive to deviate. This notion of a single strategy is the same notion used in prior works including [11, 13, 30, 35]. In the mechanism-design literature, this is called *incentive-compatibility in ex-post Nash equilibrium*; see [35] and Appendix B. We also comment that in a setting with path verification, the result is slightly stronger since it only requires that honest nodes use consistent export. (We do not know if consistent export suffices for the next-hop result.) The proof of Theorem 4.1 is presented in Appendix D, and makes heavy use of the result of Feigenbaum *et al.* [11, 13].

4.4 Our results need consistent export.

Theorems 4.1 required a consistent export rule. We now show that we cannot drop this requirement, by presenting a counterexample that obeys all the conditions in Theorem 4.1 (policy consistency, next-hop policy, path verification) except consistent export, where node m still has an incentive to lie about its forwarding path in order to gain a traffic-volume attraction:

Figure 4: INCONSISTENT EXPORT demonstrates that m can have an incentive to lie about its forwarding path in order to attract indirect traffic from node c, by taking advantage of the fact that some other node (n) does not use consistent export. Suppose that all nodes *except for* n use export-all rule (which implies consistent export). Now suppose that node n uses an *inconsistent* export rule; it exports the path $nm1d$ to node c, but not the more preferred path nmd. On the left we show the outcome T that results when all nodes use a BGP-compliant strategy with $r_n(\cdot) \equiv v_n(\cdot)$ and the export rules described above. In T, nodes m and n use the path nmd, but because n does not export this path to c, c routes directly to d. The manipulated outcome M is shown on the right, where only node m deviates from the BGP-compliant strategies described above. By announcing the false path "$m1d$", m manages to attract traffic from c, since now n is willing to export the path "$nm1d$" to node c. Notice that this false path can be announced even if the network has path verification, since node 1 announced "$1d$" to m. (Note that INCONSISTENT EXPORT is a Gao-Rexford network that *does not obey AT4*, where there is no dispute wheel and all nodes use next-hop policy.)

The reader might object to the fact that in INCONSISTENT EXPORT, node c prefers the long path $cnm1d$ over the short path cd. We note that this counterexample holds even we lengthen the cd path (say by replacing the c-d link by a path through four additional nodes). On the other hand, we agree that the inconsistent export rule used by node n is somewhat bizarre. Indeed, we believe that it is reasonable to require consistent export in a network that is already policy consistent.

5. RESULTS: GENERIC ATTRACTIONS

We now consider our most general notion of traffic attraction, in which the utility that nodes derive from attracting traffic can depend arbitrarily on the path that incoming traffic takes (see Section 2.3). For this general case, we show in Section 5.4 that nodes have no incentive to lie when all nodes use next-hop policy and all-or-nothing export and the network has path verification. (In fact, we show that a weaker enforcement mechanism called *loop verification* is also sufficient; see Section 5.3.) These conditions are extremely strong, but we show via a sequence of counterexamples that we cannot drop any one of these conditions without allowing an incentive to lie. The theorems and counterexamples in this section are summarized in Table 3.

5.1 Policy consistency & path verification is not enough.

In networks with only traffic-volume attraction, we were able to show that adding path verification to a policy-consistent AS graph is sufficient to ensure that nodes have no incentive to lie (Section 4.3). Unfortunately, this is not the case when we consider more general attraction relationships:

Figure 5: BOWTIE demonstrates that, even in a network that is policy consistent and has path verification, a manipulator m can have an incentive to lie about its forwarding path in order attract traffic from a customer c on the direct m-c link. Suppose node m has an attraction function such that (1) m has an AT4 attraction relationship with its customer c, and (2) m has a traffic-volume attraction with its provider n. The outcome T that results when every node uses a BGP-compliant strategy with $r_n(\cdot) \equiv v_n(\cdot)$ and exports all paths allowed by GR2, is shown on the left. The manipulated outcome M is shown on the right, where only node m deviates from the BGP-compliant strategy we de-

Verification?	Policy	Export	Incentive to Lie?	Result
None	$\star$	$\star$	Yes	FALSE LOOP
$\star$	Consistent	$\star$	Yes	BOWTIE
$\star$	Next-Hop	Consistent	Yes	GRANDMA
Path / Loop	Next-Hop	All-or-Nothing	No	Theorem 5.1

Table 3: Summary of our results for generic attractions. We also require no dispute wheel.

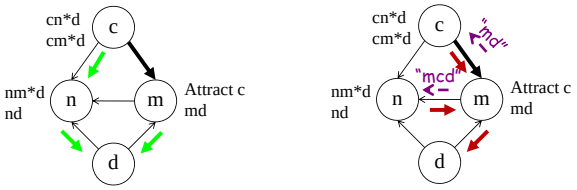


Figure 6: False Loop

scribed above.

Here, m has an incentive to dishonestly announce the path "$m1d$" to all of its neighbors in order to attract traffic from the attractee c on the direct c-m link. Node m can make this announcement, even with path verification, because node 1 announced the path $1d$ to m. Moreover, there is no BGP-compliant strategy for m that allows it to attract traffic from both c and n while maintaining its preferred data-plane forwarding path md. Note that BOWTIE is a policy-consistent, Gao-Rexford network with path verification *that does not obey AT4* and has no dispute wheel in the valuations.

We remark that even though c's traffic is routed via m in both T and M (*i.e.,* m does *not* gain a traffic-volume attraction), the manipulation in BOWTIE is quite reasonable in practice. For example, m might prefer the outcome in M over the outcome in T for load-balancing purposes, because incoming traffic from c and n is spread over two links in M. As another example, m might prefer the outcome M because it has a usage-based billing contract with c on the m-c link, whereas node m is not able to bill its provider n for carrying c's traffic (which occurs in outcome T).

5.2 Next-hop policy alone is not enough.

From BOWTIE, we learn that policy consistency is *not* sufficient to ensure honest announcements (even when using path verification). So we throw up our hands and ask if it suffices to require that every node uses next-hop policy. With next-hop policy, it is tempting to conclude that lying about an outgoing path will not help an attractor convince an attractee to 'change its mind' and route through it in a manipulated outcome. (Notice that the manipulations in INCONSISTENT POLICY, NONEXISTENT PATH and BOWTIE were of this form.) Furthermore, next-hop policy is sufficient when considering only traffic-volume attractions (Section 4.3).

Quite surprisingly, this intuition fails. We now present our most important counterexample, which shows that if the network does not have path verification, then even requiring next-hop policy is not sufficient:

Figure 6: FALSE LOOP demonstrates that, even in a network where all nodes use next-hop policies, a manipulator m can gain traffic from its customer c by falsely announcing a path through c to m's other neighbors. Suppose that m announces no paths to neighbor n and all paths to everyone else, and that all other nodes export all paths allowed by GR2. On the left is the outcome T, where each node compiles $r_n(\cdot) \equiv v_n(\cdot)$ and uses the BGP-compliant strategy with the export rules described above. The manipulated outcome M is on the right, where only m deviates from the BGP-compliant strategy above. In M, the manipulator m has an incentive to announce a false outgoing path "mcd" to n in order to attract traffic from its attractee c (on the direct c-m link). Notice that the outcome M results whenever there is no control-plane verification mechanism such as path verification, since the 'false loop' "$nmcd$" will either cause node n not to announce any path to node c, or instead cause node c to ignore the announcement. Also, m has no BGP-compliant strategy that allows it to gain an AT4 attraction from c, since c would have sent his traffic on the c-n link if m had either (a) honestly announced some path to n, or (b) announced no path to n (as in outcome T). Note that FALSE LOOP is a Gao-Rexford network with no dispute wheel that obeys AT4, in which all nodes use next-hop policies.

5.3 Introducing loop verification.

To deal with the manipulation in FALSE LOOP, we introduce loop verification, a new control-plane mechanism that deals with detecting and preventing "false loops."

BGP allows two different approaches for detecting and preventing routing loops. One is sender-side loop detection, where a node a will *not* announce path aRd to node b if b happens to be on the path R. The other is receiver-side loop detection where a *will* announce the path aRd to b, so that b will detect the loop and discard that announcement. Receiver-side loop detection has the advantage of allowing a node b to hear announcements that *falsely* include a path that b did not announce. Notice that for b to detect a "false loop," b need only perform a *local* check to see if the path it receives matches the one that b actually announced. (This local check is less onerous than the one that is required for path verification, which requires participation from all ASes on the path.)

Loop verification encourages ASes to avoid lying in BGP announcements because they should fear getting caught. We define loop verification as the use of receiver-side loop detection by *all* nodes in a network, with the additional requirement that when node b receives an announcement of a path $P = QbRd$, such that b did not announce the path bRd to its neighbors, then b "raises an alarm." Then, the first node who announced a path that includes bRd will be punished with utility reduced to $-\infty$. This punishment process models the idea that b can catch and shame the node that announced the false loop, *e.g.,* via the NANOG list.

The properties of loop verification are strictly weaker than those of path verification. Namely, if a network has path verification, then no node will raise an alarm in loop verification. This follows from the fact no node can announce a path that includes bRd unless b announces the path bRd.

5.4 Next-hop policies & loop verification is enough!

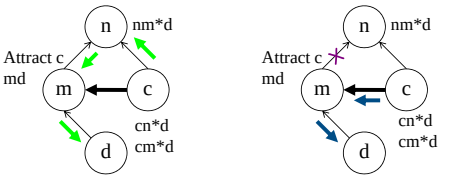


Figure 7: Access Denied.

Now that we defined loop verification, we are ready to present the main result of this section. If we add loop verification to a next-hop network with no dispute wheel, we can eliminate the manipulation performed by m in False Loop. We also require all nodes to use an all-or-nothing export rule. The following holds even if the network does *not* obey the Gao-Rexford conditions:

Theorem 5.1. *Consider an AS graph where the valuation functions are next-hop and contain no dispute wheel. Suppose that all nodes, except a single manipulator node m, use BGP-compliant strategies where they set their ranking equal to their valuations ($r_n(\cdot) \equiv v_n(\cdot)$ for every node n), and obey all-or-nothing export. Suppose further that the network uses either loop verification or path verification. Then there exists a BGP compliant strategy for m that uses $r_m(\cdot) \equiv v_m(\cdot)$ and obeys all-or-nothing export, which obtains the best possible* stable outcome *in terms of the utility function of m.*

On an intuitive level, Theorem 5.1 proves that any gains a manipulator gets from lying can be obtained by using a clever export rule.[7] That is, Theorem 5.1 shows the *existence* of an optimal all-or-nothing export rule for the manipulator; however, this optimal export rule for m depends on the export rules chosen by the other nodes in the network. Furthermore, unlike prior work or the result from Section 4, this result does *not* explicitly describe this optimal export rule.

The proof of Theorem 5.1 is quite technically involved, so we present it in Appendix E. Roughly, the proof amounts to showing that when all nodes use next-hop policy with their neighbors, the only strategically useful lie available to the manipulator is to announce a false loop. Then, we show that if the network has loop verification, some node detects the false loop and punishes the manipulator for its lie; since the utility of the manipulator drops down to $-\infty$ when it gets caught, it no longer has an incentive to announce a false loop, and the theorem follows.

5.5 Export-all is not always optimal.

Theorem 5.1 unfortunately does *not* explicitly describe the optimal export rule for the manipulator. We now show that the export-all rule (which was shown to be optimal in *e.g.,* Theorem 4.1 and [30]) is not necessarily optimal in this setting:

Figure 7: Access Denied demonstrates that m can attract traffic from its customer c over the direct m-c link by denying export to some of m's other neighbors. Here, the network has path and loop verification, next-hop policies at every node, and m is interested in attracting traffic only from c (but not from n) in an AT4 attraction. Suppose that

[7]We remark that this result only rules out the possibility of obtaining a better *stable* outcome by lying, it does not rule out the possibility of m gaining utility by inducing a non-stable outcome. See Section 2.2.

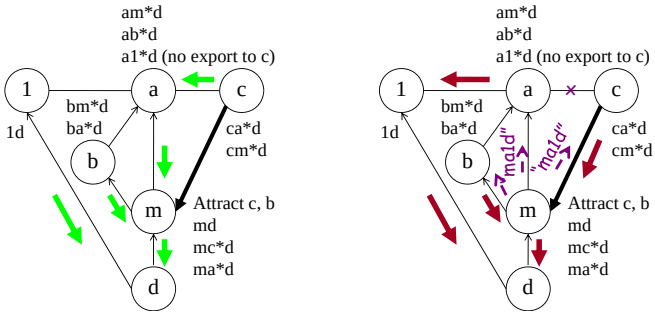


Figure 8: Grandma.

all nodes, including m, honestly announce paths. On the left we present the outcome when every node, including m, uses export-all. On the right, we illustrate the outcome when m uses a different all-or-nothing export rule: in particular, m announces all paths (honestly) to c, and no paths to n. As a result, m attracts traffic from c on the direct c-m link. If m had announced paths to n, then c would not have sent its traffic on the c-m link, as in the outcome on the left. Thus, we see that the export-all rule is not optimal for m. Note that Access Denied is a network that obeys GR1, GR3, and AT4, and has no dispute wheel.

We pause here to observe that in the outcome on the right, n has no path to the destination if node c only exports the paths allowed by GR2. We discuss this issue in Section 6.4.

5.6 Theorem 5.1 needs all-or-nothing export.

The requirement that all nodes use an all-or-nothing export policy in Theorem 5.1 is extremely strong, especially because most networks that obey the Gao-Rexford conditions (in particular GR2) violate this export rule. We now present our most devastating (and complicated) counterexample that shows Theorem 5.1 does not hold with a more realistic export rule like consistent export:

Figure 8: Grandma demonstrates that a manipulator m can have an incentive to lie in order to attract traffic from a customer c if some other node a does not use an all-or-nothing export policy. Furthermore, Grandma shows that this is possible even when all nodes use path verification and next-hop policies.

In Grandma, m has an AT4 attraction relationship with its customer c, a traffic-volume attraction relationship with its provider b, and no other attractions. Suppose now that all nodes export all paths allowed by GR2; thus, a does *not* export paths through its peer 1 to its peer c. While a uses a consistent export rule (since a filters only its lowest ranked path through 1), a does not use all-or-nothing export rule. On the left is the outcome T that results when all nodes act honestly, *i.e.,* use BGP-compliant strategies with $r_n(\cdot) \equiv v_n(\cdot)$ and the export rules above. The manipulated outcome M is shown on the right, where only the manipulator m deviates from the BGP-compliant strategies above.

In M, the manipulator m dishonestly announces the path "$ma1d$" while actually routing on md. To arrive at the outcome M on the right, node m sits quietly until node a exports "$a1d$" to it. Then m announces "$ma1d$" to all nodes, while routing on md in the data plane. Node a cannot route through m (because it thinks that m routes through it); so, a continues to route on $a1d$. Next, because a does not export paths through 1 to its peer node c, node c has no choice

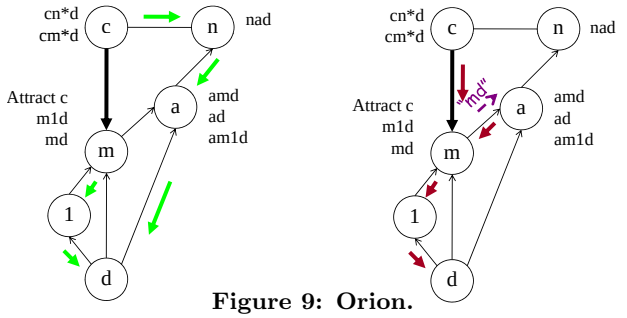


Figure 9: Orion.

but to route through node m. Meanwhile, m's machinations have no effect on b, who routes through m regardless. Notice that loop or path verification would not help, since node a is indeed routing along "$a1d$". Furthermore, m manages to retain in M its traffic-volume attraction with b and gain an AT4 attraction with customer c. Also, m has no BGP-compliant strategy that obtains as large a utility as it obtains from M. Note that Grandma is a Gao-Rexford network with no dispute wheel *that does not obey AT4*, where all nodes use next-hop policy with all their neighbors.

5.7 The need for ubiquitous participation.

Bowtie and Grandma highlight another important point; namely, that even if one node follows the conditions specified in our theorems, *e.g.,* next-hop policy, it is still possible for that node to learn a false path, if some other node in the network fails to follow the specified conditions. For example, in Bowtie (Figure 5), even though attractee node n uses next-hop policy, n still learns a false path because node c does not. Thus, we emphasize that all the theorems in this paper only hold if *every node* in the network follows the specified set of conditions.

6. RESULTS: CUSTOMER ATTRACTIONS IN GAO-REXFORD NETWORKS

We now focus on Gao-Rexford networks (see Section 3.3). In Section 5, we used Grandma (Figure 8) to show that Theorem 5.1 does not hold with consistent export in place of the unrealistic all-or-nothing export rule (which is usually not compatible with GR2). Fortunately, Grandma did not obey the AT4 attraction condition. Thus, we now weaken the assumption of all-or-nothing export by focusing on the AT4 setting, in which an attractor can increase its utility only if a *customer* routes on the direct link between them. It turns out that AT4 also allows us to weaken the next-hop-policy restrictions required in Theorem 5.1. Our results are summarized in Table 4, which also shows how dropping any one of the conditions in our positive result (Section 6.2) may create an incentive to lie.

6.1 It's not sufficient to restrict policy at attractees only.

The requirement in Theorem 5.1 that every node in the network uses a next-hop policy with all of its neighbors is very strong indeed. Ideally, we would have preferred to require only *attractees* to use next-hop policy with their attractors. Unfortunately, even requiring every attractee to use next-hop policy with *all its neighbors* may not remove the incentive to lie:

Figure 9: Orion is a Gao-Rexford network with no dispute wheel that obeys AT4. In Orion, only the attractee (node c) uses next-hop policy with all its neighbors (nodes m, n). Every other node uses next-hop policy with its peers and providers, but not necessarily with its customers. Notice that node a is not policy consistent with its customer m: node m prefers path $m1d$ to path md (say, because it is cheaper to route directly to 1), while node a prefers the path amd to the path $am1d$ (say, because it prefers shorter paths).

On the left is the outcome T that results when each node uses a BGP-compliant strategy with $r_n(\cdot) \equiv v_n(\cdot)$, exporting all paths allowed by GR2. The manipulated outcome M is shown on the right, where the manipulator m deviates from this BGP-compliant strategy. In the manipulated outcome M, m dishonestly announces the outgoing path "md" to all of its neighbors so that node a decides to route through m on the amd path. However, node n does not admit the path amd and thus is left with no path to the destination d. The attractee c has no choice but to route through m, increasing m's utility. Observe that m has no BGP-compliant strategy that obtains as large a utility as it obtains from M.

Notice that n uses a "forbidden-set policy" [9], in which it prefers using no path at all over using a path through m. Such preferences could arise in practice if node n does not trust node m to carry its traffic (say, because it perceives node m to be adversarial).

6.2 Policy consistency everywhere with next-hop policy at attractees is enough!

Earlier, we saw that, even in the Gao-Rexford setting with AT4, dropping either path or loop verification may create an incentive to lie (as in False Loop in Figure 6). Furthermore, from Orion above, we learn that policy restrictions only on attractees can leave an incentive to lie. The manipulation in Orion is possible because node a is not policy consistent with node m; we now show that requiring policy consistency, along with other conditions satisfied by Orion, is enough to ensure no incentive to lie.

Theorem 6.1. *Consider a policy-consistent, Gao-Rexford network that obeys AT4, in which there is no dispute wheel in the valuations and all attractees use next-hop policies with their providers and peers. Suppose that all nodes, except a single manipulator node m, uses a BGP-compliant strategy with $r_n(\cdot) \equiv v_n(\cdot)$ and a consistent export rule that satisfies GR2. Suppose further that the network has path or loop verification.*

Then there exists a BGP-compliant strategy for m with $r_m(\cdot) \equiv v_m(\cdot)$ and a consistent export rule obeying GR2 that obtains the best possible stable *outcome in terms of the utility function of m. In particular, exporting all paths to customers and no paths to providers and peers is one such optimal strategy.*

The proof, in Appendix F, consists of a series of technical arguments that use the Gao-Rexford conditions (GR1-GR3) and AT4 to show that if m can increase its utility in the manipulated outcome, then the network must have a customer-provider loop.

6.3 Our result needs next-hop at attractees.

We note that we cannot drop the requirement in Theorem 6.1 that all attractees use next-hop policy with all their peers and providers. To see why, recall that a manipulation

AT4	Verification	Policy Consist.	Next-hop policy	Export	Incentive to Lie?	Result
No	$\star$	$\star$	$\star$	Consist.	Yes	GRANDMA
Yes	None	$\star$	$\star$	$\star$	Yes	FALSE LOOP
Yes	$\star$	None	All nodes w. peers & providers	$\star$	Yes	ORION
Yes	None / Loop	All nodes	None	$\star$	Yes	NONEXISTENT PATH
Yes	Loop / Path	All nodes	Attractees w. peers & providers	Consist.	No	Theorem 6.1

Table 4: Summary of our results for Gao-Rexford networks (obeying GR1-GR3) with no dispute wheel.

is possible in NONEXISTENT PATH (Figure 3), which satisfies all the conditions of Theorem 6.1 (loop verification, policy consistency at all nodes, Gao-Rexford, AT4, no dispute wheel, consistent export) except that the attractee node c does not use next hop policy with its provider m. However, the manipulation in NONEXISTENT PATH would *not* be possible with path verification (instead of loop verification). Thus, in this work we have *not* ruled out the possibility that we can drop the requirement for attractees to use next-hop policy if we replace loop verification with path verification.

6.4 It's best to export only to your customers.

Observe that Theorem 6.1 not only shows the *existence* of an optimal export rule for the manipulator, but also explicitly describes one such export rule. It therefore provides a specific strategy from which no node has an incentive to unilaterally deviate.[8] However, this strategy requires that m *never announces any paths to its peers and providers*. While this export rule obeys consistent export and GR2, a network in which every node uses this "export-nothing-to-non-customers" rule would be a very sorry network indeed: Peer paths would not exist, and nodes would never transit traffic from their providers, even if that traffic is destined for their customers!

Unfortunately, there are cases in which the optimal export rule for the manipulator is to "export nothing to non-customers." For example, consider ACCESS DENIED in Figure 7 and observe that m's optimal strategy is to announce no paths to n (which means that when c's export rule obeys GR2, node n has no path to the destination). Furthermore, this network obeys the strongest conditions considered in this work (next-hop policy at all nodes and path verification). Hence, within the conditions considered here, we cannot hope to get a result where m's optimal export policy necessarily allows it to announce paths to peers and providers.

This suggests that AT4 may *not* be a reasonable model for attraction relationships; *e.g.,* a node could improve its utility by attracting traffic from a provider or peer if it *delivers* this traffic to a customer. Finding a more appropriate model for attraction relationships in Gao-Rexford networks remains open for future research.

6.5 Our result needs no dispute wheel.

Notice that in addition to obeying the Gao-Rexford conditions, Theorem 6.1 also requires that the valuation functions have no dispute wheel. As we discussed in Section 3.3, this means that in addition to obeying GR1 and GR3, the valuation functions must contain no dispute wheel *even without excluding paths that are removed by the GR2 export rule.* This is a very strong requirement indeed, since GR2 often excludes paths from the network that would have created

[8]However, as in Theorem 5.1, we add the disclaimer that this result only applies to *stable* manipulated outcomes.

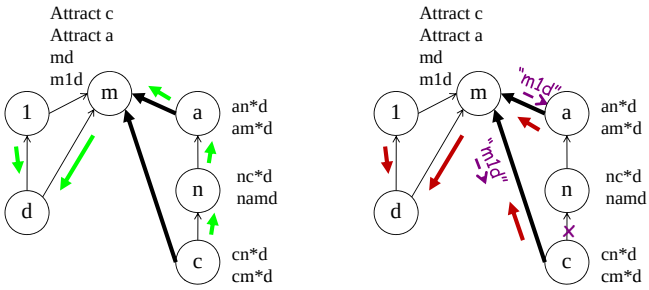


Figure 10: Disputed Path.

dispute wheels. Ideally, we would like to drop this requirement from Theorem 6.1. Unfortunately, this is not possible:

Figure 10: DISPUTED PATH demonstrates that, if a network has a dispute wheel, a manipulator m can have an incentive to falsely announce paths in order to attract traffic from a customer c. Furthermore, DISPUTED PATH shows that this is possible even if there is path verification, all nodes are policy consistent, and every attractee (nodes c, a) use next-hop policy with all their neighbors (nodes m, n).

On the left is the outcome T that results when each node uses a BGP-compliant strategy with $r_n(\cdot) \equiv v_n(\cdot)$ and exports all paths that do not violate the GR2 export condition. The manipulated outcome M is shown on the right, where only node m deviates from this strategy. In the manipulated outcome M, m announces a false outgoing path "$m1d$" to all of its neighbors. This is possible even with path verification since 1 announced the path $1d$ to m. Notice that while node n is policy consistent with all his neighbors, he does not admit the path $nm1d$. Furthermore, since c obeys GR2, he does not export any paths to n. As a result, n is left with no path to the destination, and c routes through his attractor m instead. However, the other attractee node a continues to route through m even when m announces this false path. Furthermore, m has no export rule for which he can achieve the same utility that obtained in M. Note that DISPUTE PATH is a Gao-Rexford network where all nodes are policy consistent, every attractee use next-hop policy with all neighbors, and there is path verification. DISPUTED PATH has a dispute wheel between nodes c, n; n prefers paths through its customer c over paths through its provider a, but c prefers paths through its provider n over paths through its provider m.

One way to get rid of the requirement for no dispute wheel is to change our interpretation of the Gao-Rexford conditions. Namely, we could assume instead that paths that are usually excluded by the GR2 export rule are also *not admitted by the valuation function of all nodes.* This means that paths that violate GR2 are filtered on ingress, (rather that filtered on egress, as per Section 3.3). This approach is discussed in [30]. (However, we emphasize here that Theorem 6.1 does *not* hold under this alternate interpretation

of the Gao-Rexford conditions.) While this interpretation may lead to better positive results, it may be unrealistic; for instance, in DISPUTED PATH, node c has no reason to announce the path $cnm1d$ to node n, since both m and n are providers of c and c only stands to lose money by transiting traffic from one provider to another. Thus, it seems reasonable to expect c to refuse to export this path. Meanwhile, n has no reason not to admit the path $ncm1d$, since this path is through his customer c. Furthermore, in practice, business relationships between ASes are often kept private. Thus, it is not clear how n would learn that node m is c's provider, and therefore that node n should not admit the path $ncm1d$.

7. RELATED WORK

We discussed some related work in Sections 1–2. Further discussion is below. Griffin, Shepherd, and Wilfong [22] developed a formal model of BGP which assumes ASes choose paths based on an arbitrary preference function that ranks *outgoing paths*. They used this model to initiate a study of sufficient conditions to ensure that BGP converges to a unique outcome (Section 3.1). This study was continued by many subsequent works; most relevant here are the results of Gao and Rexford [15] who considered constraints that arise due to business relationships between ASes (Section 3.3), and those of Feamster, Johari, and Balakrishnan [8] who studied the effect of filtering (Section 3.4).

In contrast to the works on BGP convergence, the game theoretic studies of BGP [7, 9–13, 30, 35], discussed in Section 1.2 and throughout this paper, looked for mechanisms that induce incentives to comply with the protocol (which, in particular, means that ASes would have no incentive to lie). These works interpret the preference function in Griffin *et al.* [22] as a measure of utility for each AS, and model ASes as rational agents who act selfishly to maximize utility. This is equivalent to assuming that utility is uniquely determined by *outgoing paths*. To our knowledge, our work is the first to model the effect of *incoming traffic* on the *incentive* to lie in BGP announcements. Earlier versions of our work appeared as [18] and [26].

Recently, the literature on BGP convergence has begun to model the effect of *incoming traffic* on BGP dynamics. These works [16, 40, 41] focus on the context of traffic engineering, and assume that ASes honestly announce paths; they do not consider ASes that lie. Gao, Dovrolis and Zegura [16] and Wang *et al.* [40] study algorithms for traffic attraction and deflection using AS-path prepending. (Our work does not model prepending.) Wang *et al.* [41] study oscillations that can occur if the BGP decision process depends on incoming traffic as well as outgoing paths. In contrast, our work allows *utility* to depend on incoming traffic (Section 2.3) but assumes that the BGP dynamics are based on *ranking* functions (Section 2.2) that depend only on outgoing paths. The ranking functions are derived from a "compilation" of the utility function (Section 2.5). Thus, in some sense, Wang *et al.* study the oscillations that can result as ASes continuously adjust their compilation. Indeed, Figure 2 of [41] shows conditions under which INCONSISTENT POLICY in our Figure 2 could experience such oscillations.

8. CONCLUSIONS

In this work, we considered control-plane mechanisms that provide incentives for rational ASes to announce their true data-plane paths in BGP messages. We find that conditions previously shown to be sufficient for honesty no longer suffice if we assume that ASes can benefit by attracting incoming traffic from other ASes. We demonstrated that, within the *control-plane* mechanisms we considered here, ensuring honesty in the face of traffic attraction requires very strong restrictions on routing policy (at the very least, policy consistency everywhere, and sometimes also next-hop policy at certain ASes), as well as control-plane verification (loop-verification or path-verification protocols like Secure BGP [27]). Thus, our results suggest that in practice, it will be difficult to achieve honesty without resorting to expensive *data-plane* protocols that verify and enforce AS-level paths. By highlighting the difficulty of matching the control and data planes, even under the assumption that ASes are rational (and not arbitrarily malicious), our results can also help inform decisions about whether security protocols should be deployed in the control plane, in the data plane, or in both.

Acknowledgments

We thank Jennifer Rexford, Michael Schapira and Joan Feigenbaum for discussions and valuable feedback that has greatly improved this work. We also thank Boaz Barak, Matthew Caesar, Andreas Haeberlen, Martin Suchara, Gordon Wilfong, and the anonymous SIGCOMM'08 reviewers for useful comments.

9. REFERENCES

[1] K. Argyraki, P. Maniatis, O. Irzak, A. Subramanian, and S. Shenker. Loss and delay accountability for the Internet. *ICNP*, 2007.

[2] I. Avramopoulos and J. Rexford. Stealth probing: Data-plane security for IP routing. *USENIX*, 2006.

[3] H. Ballani, P. Francis, and X. Zhang. A study of prefix hijacking and interception in the Internet. In *ACM SIGCOMM*, 2007.

[4] S. Balon and G. Leduc. Can forwarding loops appear when activating iBGP multipath load sharing? In *AINTEC*, 2007.

[5] S. Bradner. Key words for use in RFCs to indicate requirement levels. RFC 2119, March 1997.

[6] K. Butler, T. Farley, P. McDaniel, and J. Rexford. A survey of BGP security issues and solutions. Technical report, ATT Labs-Research, 2004.

[7] R. R. Dakdouk, S. Salihoglu, H. Wang, H. Xie, and Y. R. Yang. Interdomain routing as social choice. In *Incentive-Based Computing (IBC)*, 2006.

[8] N. Feamster, R. Johari, and H. Balakrishnan. Implications of autonomy for the expressiveness of policy routing. In *ACM SIGCOMM*, 2005.

[9] J. Feigenbaum, D. R. Karger, V. Mirrokni, and R. Sami. Subjective-cost policy routing. In X. Deng and Y. Ye, editors, *First Workshop on Internet and Network Economics*, 2005.

[10] J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenker. A BGP-based mechanism for lowest-cost routing. *Distributed Computing*, 18(1), July 2005.

[11] J. Feigenbaum, V. Ramachandran, and M. Schapira. Incentive-compatible interdomain routing. In *Conference on Electronic Commerce*, pages 130 – 139, 2006.

[12] J. Feigenbaum, R. Sami, and S. Shenker. Mechanism design for policy routing. *Distributed Computing*, 18(4):293–305, 2006.

[13] J. Feigenbaum, M. Schapira, and S. Shenker. *Algorithmic Game Theory*, chapter Distributed Algorithmic Mechanism Design. Cambridge University Press, 2007.

[14] L. Gao, T. Griffin, and R. Rexford. Inherently safe backup routing with BGP. *IEEE Infocomm*, 2001.

[15] L. Gao and R. Rexford. Stable Internet routing without global coordination. *IEEE/ACM Trans. on Network.*, 2001.

[16] R. Gao, C. Dovrolis, and E. Zegura. Interdomain ingress traffic engineering through optimized AS-path prepending. In *IFIP Networking*, 2005.

[17] V. Gill, J. Heasley, and D. Meyer. The generalized TTL security mechanism (gtsm). RFC 3682, 2004.

[18] S. Goldberg and S. Halevi. Rational ASes and traffic attraction: Incentives for honestly announcing paths in BGP. Technical Report TR-813-08, Princeton University, Dept. of Computer Science, Feb. 2008.

[19] S. Goldberg, S. Halevi, A. D. Jaggard, V. Ramachandran, and R. N. Wright. Rationality and traffic attraction: Incentives for honest path announcements in BGP. In *ACM SIGCOMM*, 2008.

[20] S. Goldberg, D. Xiao, E. Tromer, B. Barak, and J. Rexford. Path quality monitoring in the presence of adversaries. In *SIGMETRICS*, June 2008.

[21] G. Goodell, W. Aiello, T. Griffin, J. Ioannidis, P. McDaniel, and A. Rubin. Working around BGP: An incremental approach to improving security and accuracy of interdomain routing. In *Network and Distributed System Security Symposium*, 2003.

[22] T. Griffin, F. B. Shepherd, and G. Wilfong. The stable paths problem and interdomain routing. *IEEE/ACM Trans. on Network.*, April 2002.

[23] A. Heffernan. Protection of BGP sessions via the TCP MD5 signature option. RFC 2385, 1998.

[24] K. J. Houle and G. M. Weaver. Trends in denial of service attack technology. Technical report, CERT Coordination Center, October 2001.

[25] G. Huston. Interconnection, peering, and settlements. In *Internet Global Summit (INET)*, June 1999.

[26] A. D. Jaggard, V. Ramachandran, and R. N. Wright. Towards a realistic model of incentives in interdomain routing: Decoupling forwarding from signaling. Technical Report 2008-02, DIMACS, Apr. 2008.

[27] S. Kent, C. Lynn, and K. Seo. Secure border gateway protocol (S-BGP). *J. Selected Areas in Communications*, 18(4):582–592, April 2000.

[28] R. Lavi and N. Nisan. Online ascending auctions for gradually expiring items. In *ACM-SIAM Symp. on Discrete Algorithms, SODA*, 2005.

[29] H. Levin, M. Schapira, and A. Zohar. The strategic justification for BGP. Technical report, Hebrew University of Jerusalem, 2006.

[30] H. Levin, M. Schapira, and A. Zohar. Interdomain routing and games. In *ACM STOC*, May 2008.

[31] X. Liu, X. Yang, D. Wetherall, and T. Anderson. Efficient and secure source authentication with packet passports. In *SRUTI'06: Steps to Reducing Unwanted Traffic on the Internet*. USENIX, 2006.

[32] Z. Mao, J. Rexford, J.Wang, and R. H. Katz. Towards an accurate AS-level traceroure tool. In *ACM SIGCOMM*, 2003.

[33] N. Nisan and A. Ronen. Algorithmic mechanism design. *Games and Economic Behavior*, 35(1-2):166–196, 2001.

[34] V. Padmanabhan and D. Simon. Secure traceroute to detect faulty or malicious routing. *HotNets-I*, 2002.

[35] D. C. Parkes and J. Shneidman. Specification faithfulness in networks with rational nodes. In *ACM PODC*, 2004.

[36] A. Ramachandran and N. Feamster. Understanding the network-level behavior of spammers. *ACM SIGCOMM*, 2006.

[37] Y. Rekhter, T. Li, and S. Hares. A border gateway protocol 4 BGP-4. RFC 4271, January 2006.

[38] L. Subramanian, V. Roth, I. Stoica, S. Shenker, and R. H. Katz. Listen and Whisper: Security mechanisms for BGP. In *NSDI*, 2004.

[39] F. Wang and L. Gao. On inferring and characterizing Internet routing policies. In *ACM IMC '03*, pages 15–26. ACM, 2003.

[40] H. Wang, R. K. Chang, D.-M. Chiu, and J. C. Lui. Characterizing the performance and stability issues of the AS path prepending method. In *ACM SIGCOMM Asia Workshop*, 2005.

[41] H. Wang, H. Xie, Y. R. Yang, L. E. Li, Y. Liu, and A. Silberschatz. On the stability of rational, heterogeneous interdomain route selection. In *ICNP*, 2005.

[42] R. White. Deployment considerations for secure origin BGP (soBGP). draft-white-sobgp-bgp-deployment-01.txt, June 2003, expired.

[43] E. L. Wong, P. Balasubramanian, L. Alvisi, M. G. Gouda, and V. Shmatikov. Truth in advertising: Lightweight verification of route integrity. In *PODC*, 2007.

APPENDIX

A. LIES AND FORWARDING LOOPS

Our results in this work indicate that in many realistic networks, rational nodes do have an incentive to deviate from BGP in order to attract incoming traffic. Hence, we often cannot rely on path announcement to accurately reflect the paths taken by traffic. But can we still rely on BGP to ensure weaker properties of routing, even if some nodes deviate from it? At the very least, can we rely on it to prevent routing loops? .

Below we consider the following mild form of deviation, which seem realistic: we assume that every node still maintains a ranking function over paths and chooses the (first hop in the) highest-ranked path that was announced to it for forwarding its traffic. However, we allow nodes to announce to their neighbors different paths than what they choose for forwarding. We also assume that paths that do not reach the destination or have routing loops are ranked $-\infty$ (*i.e.,* nodes will not knowingly send traffic into the abyss).

In general, we cannot guarantee that a network will not have any forwarding loops if (more than one) node lies. In

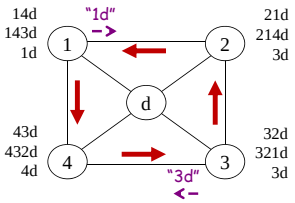


Figure 11: Forwarding Loop.

Figure 11, nodes 1 and 3 both lie about the paths they use, while nodes 2 and 4 are honest, thus causing a forwarding loop to form in the data plane. (Notice that the same forwarding loop would form in the data plane if nodes 2 and 4 lied about the paths they used.) However, we show that if the ranking functions contain no dispute wheel and the network has path verification, then no forwarding loops can occur. (This may help explain why forwarding loops are uncommon on the Internet, even though not all nodes announce their true paths.)

THEOREM A.1. *Consider an AS graph with path verification, where all nodes choose their forwarding path based on their ranking function. If a resulting outcome contains a forwarding loop in the data plane, then there are (at least two) nodes in the network that announce a path with a next-hop that is different from the next hop that they actually use, and all those nodes have a dispute wheel in their ranking functions.*

PROOF. Let T be a (not necessarily stable) outcome and assume that it has a forwarding loop in the data plane. Let the forwarding loop have the form $a_1 \rightarrow \ldots \rightarrow a_k \rightarrow a_1$ where node a_i forwards traffic to a_{i+1} and announces a path to a_{i-1}. Since we assume that nodes do not knowingly send traffic into a loop or a path that does not reach the destination (and since we have path verification), it follows that at least one node n_i that announces to a_{i-1} a path different than what it chooses for forwarding. Denote one such node by n_0 and denote the path that it announces by Q_0 and the path that it chooses for forwarding by P_0. Note that n_0P_0 reaches the destination and has no loops, since n_0 chooses it for forwarding. Note also that the first hops in Q_0 and P_0 must differ, since n_0 receives the announcement P_0 from the next hop on it, and by path-verification n_0 cannot announce a different path starting from the same next-hop.

Clearly, the next node after n_0 on P_0 is in the loop (since n_0 routes into the loop). Also, if the next node honestly announces the path that it chooses then also the node after it P_0 is in the loop, and so on. So there must be some node on P_0 that announces a path different than what it chooses (since P_0 eventually leaves the loop to reach d). Let n_1 be the first node after n_0 on the path P_0 that announces a path Q_1 that is different from what it choose for forwarding, and by the argument above n_1 must be in the loop. Also, Q_1 must be a suffix of P_0, since all the nodes between n_1 and n_0 (if any) announce honestly the path that they choose. Thus we can write $P_0 = n_0R_0n_1Q_1d$.

We similarly define $P_i = n_iR_in_{i+1}Q_{i+1}d$ for $i = 1, 2, \ldots$. (That is, P_i is the path that n_i chooses, n_{i+1} is the first node on P_i that does not announce honestly the path that is chooses, *etc.*) Repeating the arguments from above, the first-hop in P_i, Q_i must differ for all i. Note that the n_i's include all the nodes in the loop that do not announce honestly the path that they use, in the order that they appear on the loop. We must therefore eventually arrive back at n_0, namely we have $n_\ell = n_0$ (with $\ell \geq 2$).

Since the network has path verification, then the 'direct' path n_iQ_id to the destination d must exist in the graph and are available to. Still, n_i chooses the 'indirect' path $P_i = n_iR_in_{i+1}Q_{i+1}d$, which means that $r_{n_i}(n_iR_in_{i+1}Q_{i+1}d) > r_{n_i}(n_iQ_id)$. Hence, there is dispute wheel between the n_i. □

B. FORMALIZING "NO INCENTIVE TO LIE"

As we mentioned several times in the text, the formal notion of "no incentive to lie" that we use for some of our positive results is different from "incentive compatibility in ex-post Nash equilibrium" that was used in prior work; see [35]. Here we explain this difference in more detail.

B.1 Ex-Post Nash

The notion of ex-post Nash equilibrium expands upon the usual Nash equilibrium to distributed settings, where players may not have full information on each other's preferences. Below we let θ_i denote the private information of node i. (In our setting, this consists of the node's valuation and attraction functions.)

Let $s_i(\theta_i)$ be a strategy for node i; which takes as input i's private information and then describes the actions that node i takes in each round of the game. (For example, a BGP-compliant strategy was described in Definition 2.1.) A *strategy profile* $\vec{s} = (s_1, s_2, \ldots, s_k)$ is a tuple consisting of one strategy s_i for each node i. Together with the private inputs $\vec{\theta}$ of all nodes and a particular schedule t, such a strategy profile $\vec{s}$ determines a particular execution of the interdomain routing game. Below we denote by $g_t(\vec{s}(\vec{\theta}))$ the outcome of this execution. (This notation assumes that the execution converges to a stable outcome; otherwise we arbitrarily define the outcome as the first non-transient global state in this execution.)

We say that the strategy profile $\vec{s}$ is an *ex-post Nash equilibrium* if for each node i, every possible alternate strategy s_i' that i could have, every fair schedule t, and for all possible values of the private information $\vec{\theta} = (\theta_1 \ldots \theta_k)$, it holds that

$$\begin{aligned} & u_i(g_t(s_1(\theta_1), \ldots, s_i(\theta_i), \ldots, s_k(\theta_k))) \\ & \quad \geq \quad u_i(g_t(s_1(\theta_1), \ldots, s_i'(\theta_i), \ldots, s_k(\theta_k))), \end{aligned}$$

In other words, a strategy profile $\vec{s}$ is in ex-post Nash equilibrium if, regardless of the underlying private information of all other nodes, each node i obtains at least as great a utility by executing strategy s_i contained in s rather than some other strategy s_i'. This is much stronger than a Nash equilibrium, in which nodes are assumed to know the private information of other nodes, and weaker than a dominant-strategy equilibrium, in which nodes have a single strategy that is best to execute regardless of the other players' strategies (and not just their private information). Dominant-strategy equilibrium appeared in some of the initial work on mechanism design and routing [10,33]. Ex-post Nash equilibrium, as in [11, 13, 30], can be used to capture *rational specification faithfulness*. If we let the strategy profile s contain the strategies that nodes "follow a protocol as specified," then showing that s is an ex-post Nash equilib-

rium amounts to showing that nodes have no incentive to unilaterally deviate from following the protocol.

We note that ex-post Nash equilibrium does not address deviations by more than one node, although the topic of collusion-proof ex-post Nash equilibrium is addressed in [13, 30].

B.2 Partially-Specified Strategies

As defined above, ex-post Nash equilibrium requires that all nodes follow a fully-specified strategy profile. In our setting, this means in particular that all the actions of the nodes (including their filtering policies) must be spelled out in this strategy profile. We stress that this requirement goes well beyond requiring that all nodes comply with the BGP specification [37]. In particular, a BGP-compliant implementation allows node to use arbitrary ingress and egress filtering (as long as the select paths based on their ranking functions), but such arbitrary filtering is not consistent with the strategies in prior work [11, 13, 30].

Insisting that all nodes follow a fully-specified strategy-profile may not be realistic in large distributed systems, where protocols are only partially specified and many options are left for the individual implementations. (Indeed, avoiding over-specification is crucial for RFCs; see [5, §6].) We therefore describe BGP-compliance in Definition 2.1 as a *property* of a strategy (or, equivalently, as a "set of allowed strategies").

B.3 Solution Concepts

Extending the formal treatment to a set of strategy allows one to define a variety of solution concepts. Below we mention two such concepts that are used in our paper.

Ideally, one would have wanted to augment the notion of ex-post Nash, allowing also part of the *strategy* itself (*e.g.,* the export rules) and not just the valuation and attraction functions to be treated as private inputs. Namely, we would have liked to have a single (fully specified) strategy profile, such that every node i has an incentive to follow its strategy even when other nodes do not follow theirs, as long as all nodes follow "allowed strategies". Hence, this notion lies somewhere in between ex-post Nash and a dominant-strategy (and in particular it implies the standard ex-post Nash concept). We note that our positive result for traffic-volume attraction in Theorem 4.1 actually meets this strong solution concept. (The positive result for customer attraction in Theorem 6.1 achieves a similar concept, but that result is significantly weaker since it only addresses stable outcomes.)

Unfortunately, for the case of "generic attractions" in Theorem 5.1 we are not able to achieve this strong solution concept. In fact, for that case we cannot even show a standard ex-post Nash result. Instead, we settle for a very weak notion of solution, showing only that for every node there exists an "allowed strategy" that is optimal. Following Lavi and Nisan [28], this concept can be called *Set ex-post Nash*, and is defined thus:

A set profile $\vec{S} = (S_1, \ldots, S_k)$ (one set for every player) is *Set ex-post Nash equilibrium* if for every node i and every profile of fully specified strategies for the other nodes $s_1 \ldots s_{i-1}, s_{i+1} \ldots s_k$ (with $s_j \in S_j$ for all j), there exists $s_i^* \in S_i$ such that

$$u_i(g_t(s_1(\theta_1), \ldots, s_i^*(\theta_i), \ldots, s_k(\theta_k))) \geq u_i(g_t(s_1(\theta_1), \ldots, s_i'(\theta_i), \ldots, s_k(\theta_k))),$$

for every possible alternate strategy s_i' that i could have, every fair schedule t, and for all possible values of the private information $\vec{\theta} = (\theta_1 \ldots \theta_k)$.

We emphasize that this solution concept only states that the "optimal" strategy s_i^* for node i *exists* in S_i, without specifying exactly how to find it. Furthermore, this condition does not necessarily yield a single (fully-specified) strategy profile $\vec{s}$ that is an ex-post Nash equilibrium, since the optimal strategy s_i^* for node i may change depending of the strategies of the other players.

C. PROOFS: USEFUL LEMMAS

LEMMA C.1 (FALSE PATH LEMMA). *Consider an execution of the routing protocol where all the nodes in the AS graph except perhaps a single manipulator node m follow BGP-compliant strategies, and assume that this execution converges to a persistent outcome M. If any node $n \neq m$ announces a false path P in M (*i.e., *P differs from the data-plane path that n uses in M), then P must be of the form $P = nRmQd$ where nRm a true path and mQd is a false path.*

PROOF. Denote the path that n announces by $n = a_r a_{r-1} \ldots a_1 a_0 = d$. Let a_i be the closest node to n on this path that announces to a_{i+1} something other than $a_i a_{i-1} P$ where $a_{i-1}P$ is the announcement that a_i receives from a_{i-1}. Since this is not consistent with a BGP-compliant strategy, we conclude that necessarily $a_i = m$. Hence m must be on the path that n announces in this execution. Let i^* be the last occurrence of m on this path (namely $m = a_{i^*}$ and $m \neq a_j$ for $j > i^*$). Then for every $j > i^*$, a_j uses a BGP-compliant strategy so it follows that a_j announces to a_{j+1} the path $a_j a_{j-1} \ldots a_0$, and moreover a_j uses a_{j-1} as its next-hop in the data-plane path in T. It follows that the data-plane path of n begins with $n = a_r a_{r-1} \ldots a_{i^*} = m$. Thus, denoting $R = a_{r-1} \ldots a_{i^*+1}$ and $Q = a_{i^*+1} \ldots a_0$, we have that nRm is a true path, and since by assumption n announces a false path it follows that mQd must therefore be a false path. □

Next, we define a useful concept, called permitted path. Informally, a permitted path is a path that is not (ingress or egress) filtered by any node on that path.

DEFINITION C.2 (PERMITTED PATHS). Consider an AS graph where all nodes use BGP compliant strategies. We say that a path P is *permitted* if it is admitted at all the nodes in it, and moreover every node in it exports it to the next node.

Note that if all nodes use BGP compliant strategies then any data-plane path must also be a permitted path.

Our proofs rely heavily on the following lemma, due to Feigenbaum *et al.* [13].

LEMMA C.3 ([13, LEMMA 14.8]). *Consider an AS graph where all nodes use BGP-compliant strategies that obey consistent export, and where the ranking functions of all nodes are policy-consistent and contain no dispute wheels.*

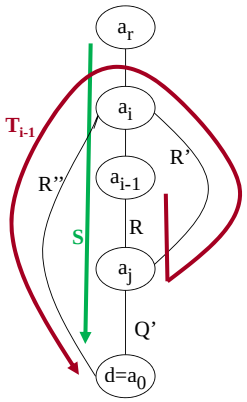


Figure 12: Case 2 of the induction step in the proof of Lemma C.3.

Then there is a unique globally stable outcome T that the protocol must converge to, and moreover T is locally optimal at all nodes in terms of the ranking functions. Namely: for any permitted path nSd in the network, the node n is assigned in T a data-plane path nRd such that $r_n(nRd) \geq r_n(nSd)$.

For self-containment, we re-prove this lemma here.

PROOF. Since the ranking contain no dispute wheel and all nodes use BGP compliant strategies, it follows from [22] that there exists a unique globally stable outcome T to which the protocol converges. It remains to show that T is locally optimal at all nodes.

Let $a_r \to a_{r-1} \to \ldots a_0 = d$ be any permitted path in the graph, and for every node a_i on this path we denote by S_i the sub-path $a_i \to \ldots a_0$. We will prove by induction over i, that each node a_i is assigned in T a path which is ranked at least as high as S_i.

Base case. The case $i = 0$ is trivially true, because the only path for $a_0 = d$ is the empty one.

Induction step. Assume that for all $j < i$ it holds that the path assigned to a_j in T (which we denote T_j) is ranked at least as high as S_j, namely $r_{a_j}(T_j) \geq r_{a_j}(S_j)$. (This implies in particular that a_j is assigned some path in T.) We now prove for a_i.

Note that a_{i-1} is willing to export S_{i-1} to a_i (since we said that S was permitted), and therefore it must also announce T_{i-1} to a_i because of consistent export. We have two cases: either the path T_{i-1} goes through a_i, or it does not.

1. If T_{i-1} does not go through a_i then from policy consistency and $r_{a_{i-i}}(T_{i-1}) \geq r_{a_{i-i}}(S_{i-1})$ we get that also

$$r_{a_i}(a_i T_{i-1}) \geq r_{a_i}(a_i S_{i-1}) = r_{a_i}(S_i)$$

   Hence a_i has an available path that is ranked at least as high as S_i, and therefore must choose one such highly-ranked path in T.

2. Assume now that the path T_{i-1} does go through a_i. We depict this case in Figure 12.

   Denote the longest common prefix of the paths T_{i-1} and S_{i-1} by $Ra_j = (a_{i-1} \ldots a_{j+1})a_j$ (note that R may be empty). Namely, we have $T_{i-1} = Ra_jQ$, $S_{i-1} = Ra_jQ'$, and the first nodes in Q, Q' differ. (In other words, the node a_j is the first node on the path S_{i-1} that uses a different next-hop in S_{i-1} and T_{i-1}.) Since T_{i-1} goes through a_i but S_{i-1} does not, it means that a_i must be somewhere on the sub-path Q, so we can rewrite T_{i-1} as $T_{i-1} = Ra_jR'a_iR''d$, where $T_j = a_jR'a_iR''d$ is the path assigned to a_j in T (and $T_i = a_iR''d$ is assigned to a_i in T).

   By the induction hypothesis we have that $r_{a_j}(T_j) \geq r_{a_j}(S_j)$, but since a_j uses different next-hops in T_j, S_j then the inequality must be strict. It must therefore be the case that $r_{a_i}(T_i) \geq r_{a_i}(S_i)$, or else we have a (2-pivot) dispute-wheel between a_i and a_j: a_i prefers $S_i = a_i \ldots a_j \ldots a_1 d$ over $T_i = a_iR''d$, and a_j prefers $T_j = a_jR'a_iR''d$ over $S_j = a_j \ldots a_1 d$.

□

D. PROOFS: VOLUME ATTRACTIONS

We now prove Theorem 4.1.

THEOREM 4.1 *Consider an AS graph where the valuation functions contain no dispute wheels. Suppose that all nodes, except a single manipulator node m, use BGP-compliant strategies and set their ranking equal to their valuations ($r_n(\cdot) \equiv v_n(\cdot)$ for every node n). Suppose further that m has a traffic-volume attraction function, and that at least one of the following two conditions hold:*

a. The valuations function of all nodes are next-hop and the export functions of all the nodes but m obey all-or-nothing export; or

b. The valuations function of all nodes are policy consistent, the export functions of all the nodes but m obey consistent export, and the network has path verification.

Then there is a BGP compliant strategy for m that sets $r_m(\cdot) \equiv v_m(\cdot)$ and obeys all-or-nothing export (and therefore also consistent export), such that this strategy is optimal for m. In particular setting $r_m(\cdot) \equiv v_m(\cdot)$ and using export-all rule is one optimal strategy.

PROOF. Consider an arbitrary strategy for m and denote by M any persistent outcome of the protocol (which need not be globally stable, see Section 3.1). We assume that $u_m(M) > -\infty$ (or else any BGP-compliant strategy for m will do).

Now consider a BGP compliant strategy for m where $r_m(\cdot) \equiv v_m(\cdot)$, and m exports-all on every edge on which it announces a simple path in M. The rest of m's export policy can be arbitrary, as long as it complies with consistent export. Clearly this strategy is BGP compliant and obeys consistent export, and moreover when m uses this strategy then the ranking functions of all nodes are policy-consistent and contain no dispute wheels (since they are set equal to the valuation functions). We can therefore apply Lemma C.3 to conclude that there is a unique globally stable outcome T, which is locally optimal at all nodes with respect to the ranking functions. We now prove that the utility of m in T is at least as high as in M. A crucial observation (that we prove in Lemma D.1 below), is that for every node n, the data-plane path of n in T has valuation at least as high as any control-plane announcement that n receives in M. We can now show that $u_m(T) \geq u_m(M)$.

- From the crucial observation Lemma D.1, we know that the valuation of m in T is at least as high as in M (since m routes in M on some path that was announced to it). Thus $v_m(M) \leq v_m(T)$.

- Next we show that every node routing through m in M must also route through it in T, and so $\alpha_m(M) \leq \alpha_m(T)$. To do this, fix some path $R = (n_r n_{r-1} \dots n_0 = d)$ that *does not go through* m in T. We prove by induction on i that each of the nodes n_i use the same path also in M. The base case $n_0 = d$ this is trivial. For the induction step, assume now that every n_j with $j < i$ uses the same path in T and M. We prove this is also the case for n_i. Denote the path that n_{i-1} uses in T and M by R_{i-1}. Since $n_{i-1} \neq m$ then we know that n_{i-1} exports the path R_{i-1} to n_i also in M. From the crucial observation Lemma D.1, we also know that R_{i-1} is at least as good as any path which is announced to n_i in M (since n_i is in a persistent state). Further, R_{i-1} must be strictly better for n_i than any path that does not have next-hop n_{i-1}. Hence n_i will choose the path $n_{i-1}R_{i-1}d$ in M as well, and we have completed the induction step.

Thus, since $u_m(\cdot) = v_m(\cdot) + \alpha_m(\cdot)$, we have that $u_m(T) \geq u_m(M)$, and Theorem 4.1 follows. □

LEMMA D.1 (CRUCIAL OBSERVATION). *Consider an AS graph where the valuation functions contain no dispute wheels, where one node m uses an arbitrary strategy and all other nodes use some BGP-compliant strategies with $r_n(\cdot) \equiv v_n(\cdot)$. Let M denote an outcome of the routing protocol in this network and assume that $u_m(M) > -\infty$ (M is a globally persistent outcome, but need not be globally stable).*

Consider further a BGP-compliant strategy for m where $r_m(\cdot) \equiv v_m(\cdot)$ and m exports-all on every edge on which it announces a simple path in M. The rest of m's export policy can be arbitrary, as long as it complies with consistent export. Let T denote the unique globally stable outcome of the protocol in this modified network.

Finally, assume that at least one of the following two conditions hold:

a. *The valuations function of all nodes are next-hop and the export functions of all the nodes but m obey all-or-nothing rule; or*
b. *The valuations function of all nodes are policy consistent, the export functions of all the nodes but m obey consistent export, and the network uses path verification.*

Then for every node n in the network, $v_n(T)$ is at least as high as the valuation of any path announcement that n receives in M.

PROOF. Let R be a path announcement that a node n receives in M, and assume that $v_n(nR) > -\infty$ (otherwise there is nothing to prove). This means that nR is a simple path that reaches the destination, so we can denote it by $R = n_{r-1} \dots n_1 n_0$ with $n_0 = d$ (and we also denote $n = n_r$). In the rest of this proof, we show that there must exists a path nS which is permitted in the network where m uses the BGP-compliant strategy above, such that $v_n(nS) \geq v_n(nR)$. Then, if we apply Lemma C.3 to the permitted path nS, it follows that the path assigned to n in T has valuation at least as high as $v_n(nS) \geq v_n(nR)$ and Lemma D.1 follows.

First, notice that if the manipulator m is *not on* R then the path nR itself is permitted in the "BGP compliant network" and we are done. Now assume that $m = n_j$ for some $j \leq r-1$. Since we assumed that $u_m(M) > -\infty$ then m has some data-plane path to the destination in M, and we denote this path by mQ. Note that mQ is a data-plane path that includes only honest nodes, so it must be permitted in the "BGP compliant network". We now consider separately the two cases in the lemma statement.

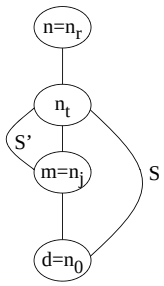


Figure 13: The proof of Theorem 4.1

Case a: next-hop policy and all-or-nothing export. There are two sub-cases: either mQ goes through n, or it does not.

- Suppose mQ does not go through n. Let t be the highest index ($j \leq t < r$) such that the path mQ goes through n_t, and denote the portion of mQ from n_t and on by $n_t S$. Thus S is a data-plane path that does not go through $n_r = n$ and does not go through $n_j = m$. (See Figure 13.) Hence $n_r \dots n_t S$ is a simple path, and by next-hop policy it holds that $v_n(n_r \dots n_t S) = v_n(n_r \dots n_t \dots n_0) = v_n(n_r R)$. Thus we have proved that the path $n_r \dots n_t S$ is ranked at least as high as nR. It remains to prove that it is permitted. We have two sub-cases: either $m = n_t$ or not.

  $m = n_t$. In this case, we have $t = j$ and $Q = S$. Then all the nodes $n_{j+1} \dots n_{r-1}$ must be honest and since n_r receives the announcement $n_{r-1} \dots n_1 n_0$ then m must have announced something to n_{j+1} in M. By construction, m must export all on this link in its BGP compliant strategy. Also the path mS is admitted at m (since m has ranking more than $-\infty$), and so $mS = nR$ is a permitted path as required.

  $m \neq n_t$. In this case m is not on the path $n_r \dots n_t S$. We prove by induction that each honest node n_i admits and exports the path $n_i n_{i-1} \dots S$ in M.

  As a base case, n_t uses the data-plane path $n_t S$ by construction, and thus $n_t S$ must be permitted. Furthermore, since n_t exports a path to n_{t+1} in M, from all-or-nothing export we have that n_t is willing to export $n_t S$ also in M. For the induction step, suppose that n_{i-1} admits and exports $n_{i-1} \dots n_t S$ to n_i. Since n_i uses next-hop policy, we have that $v_{n_{i+1}}(n_i n_{i-1} \dots n_t S) = v_{n_{i+1}}(n_i n_{i-1} \dots n_t \dots d)$. Since n_i exported a path to n_{i-1} in T, from all-or-nothing export we have that n_i is willing to export $n_i n_{i-1} \dots n_t S$ also in M.

  Thus our induction has shown that the path $n n_{r-1} \dots n_t S$ in M is permitted (since all the nodes on that path admit it and are willing to export it), and moreover that $n_r n_{r-1} \dots n_t S$ is ranked at least as high as $n_r n_{r-1} \dots n_1 d = nR$ as required.

- Suppose mQ does go through n. Then denote mQ as $mS'nS$. Now nS is permitted since it is a data-plane path, and nS must have higher ranking than nR since

(because n is in a persistent state) n received the announcement R but is routing in the data-plane over nS.

This concludes the proof for the setting of next-hop policy and all-or-nothing export.

Case b: policy-consistency and path verification. Due to path verification, we know that the path R is admitted and exported by all the "honest nodes" $n_i \neq m$ and therefore these nodes admit it and export it also in T. Also, by the way that we defined the ranking and export functions of m we know that IF $v_m(mn_{j-1} \dots n_0) > -\infty$ then also m will admit and export this path in T (and again we have that nR is permitted).

It is left to consider the case that $v_m(mn_{j-1} \dots n_0) = -\infty$, namely the case where m announces in M a path that is not admitted by its valuation function. Again, let t be the highest index ($j \leq t \leq r$) such that the data-plane path mQ that m uses in M goes through n_t, and denote the portion of mQ from n_t and on by n_tS (so S does not go through $n_j = m$). (See Figure 13.) We now show that the valuation $v_{n_t}(n_tS)$ must be at least as high as $v_{n_t}(n_tn_{t-1} \dots n_0)$.

- If $n_t = n_j = m$ (so mQ and n_tS is the same path) then this follows from the fact that $v_m(mQ) > -\infty = v_m(mn_{j-1} \dots n_1 d)$.
- If $m \neq n_t$ then we re-write the path mQ as $mS'n_tS$, and notice that we must have $v_{n_t}(n_tS) \geq v_{n_t}(n_t \dots m\ n_{j-1} \dots n_0)$, or else we have a dispute wheel between n_t and m (since $v_m(mS'n_tS) > v_m(mn_{j-1} \dots n_0) = -\infty$).

Now consider the path $n_rn_{r-1} \dots n_tS$. This is a simple path, and we just showed that $v_{n_t}(n_tS) \geq v_{n_t}(n_tn_{t-1} \dots n_0)$. From policy consistency it follows that also for each n_i, $t+1 \leq i \leq r$, the path $n_i \dots n_tS$ has ranking at least as high as $n_in_{i-1} \dots n_1$ (and therefore also valuation at least as high), since each n_i exports the path $n_in_{i-1} \dots n_1$ to n_{i+1} in T it follows from consistent export that n_i exports $n_i \dots n_tS$ in M. Hence $n_rn_{r-1} \dots n_tS$ is a permitted path with valuation in n at least as high as nR, as needed. This concludes the proof for the setting of policy consistency and path verification. □

E. PROOFS: GENERIC ATTRACTIONS

THEOREM 5.1 *Consider an AS graph where the valuation functions are next-hop and contain no dispute wheel. Suppose that all nodes, except a single manipulator node m, use BGP-compliant strategies where they set their ranking equal to their valuations ($r_n(\cdot) \equiv v_n(\cdot)$ for every node n), and obey all-or-nothing export. Suppose further that the network uses either loop verification or path verification. Then there exists a BGP compliant strategy for m that uses $r_m(\cdot) \equiv v_m(\cdot)$ and obeys all-or-nothing export, which obtains the best possible* globally stable outcome *in terms of the utility function of m.*

PROOF. Let M be a globally stable outcome that is obtained by an arbitrary (possibly cheating) strategy for m. We again assume that $u_m(M) > -\infty$, or else there is nothing to prove. In particular this implies that m has a data-plane path to d in M. Also, by the discussion in Section 2.3 we can assume without loss of generality that m has a single outgoing link in M.

Consider a BGP compliant strategy for m where $r_m \equiv v_m$ and m exports-all on every edge on which it announces a simple path in M, and exports nothing on every other edge. Clearly this strategy is BGP compliant and obeys all-or-nothing export, and moreover when m uses this strategy then the ranking functions of all nodes are next-hop (and therefore also policy-consistent) and contain no dispute wheel (since they are set equal to the valuations). This is exactly the setting of Case b of the crucial observation Lemma D.1, so we know that there is a unique globally stable outcome T such that for every node n in the network, the path assignment of n in T has valuation at least as high as any path-announcement that n receives in M. In particular, it follows that $v_m(T) \geq v_m(M)$ (because m routes in M on some path that was announced to it). Since $u_m(\cdot) = v_m(\cdot) + \alpha_m(\cdot)$, it only remains to show that $\alpha_m(T) \geq \alpha_m(M)$.

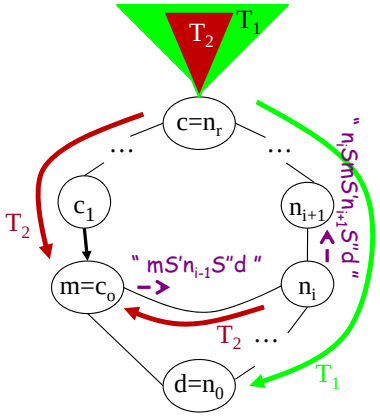


Figure 14: The proof of Theorem 5.1

Assume to the contrary that we have $\alpha_m(T) < \alpha_m(M)$. We prove a sequence of statements that imply that some other node b must have raised an alarm, because it receives a path announcement of the form QbR where b did not announce the path R, and where m is on path Q. This contradicts either path verification (since b receive an announcement containing a path through b that b did not announce) or loop verification (where the utility of m is set to $-\infty$ when such an alarm is raised).

CLAIM E.1. *There is a node c that (1) routes through m in M, (2) uses a different outgoing edge in M than in T, (3) every node that routes through c in M uses the same outgoing link in T and M.*

PROOF. We assumed towards contradiction that m gained an attraction in M, $\alpha_m(M) > \alpha_m(T)$, which implies that the subtree of m in M cannot be contained in the subtree of m in T, namely $M(m) \not\subseteq T(m)$. Hence, there exists some node that routes through m in M and uses a different next hop in M than in T.

Denoting $m = c_0$, we continue to find nodes $c_i (i \geq 1)$ as follows: For each node c_i, if there are nodes that route through c_i in M and use a different next-hop in M than in T, then we let c_{i+1} be one such node. We repeat this process until we reach a "last node" c such that every node that routes through c in M uses the same next-hop in T and in M.

Observe that we must reach such "last node" since otherwise we will eventually repeat a node, say node c_r. But since each c_i routes through c_{i-1} then repeating a node means

that we have a routing loop in M, and since all these nodes route through m and all of them (including m) have just one outgoing link, it follows that m is part of this routing loop, so in particular m does not have a path to the destination in M and $u_m(M) = -\infty$.

It follows by definition that this "last node" c satisfies items (1) through (3) in the claim assertion. □

CLAIM E.2. *Node c has a data-plane path to d in T.*

PROOF. We again use the crucial observation Lemma D.1 to establish that the path assignment of c in T is ranked at least as high as any announcement that node received in M. In particular c is routing through m so it must have received an announcement with rank higher than $-\infty$ in M, so it must have a path with rank higher than $-\infty$ also in T. □

Denote the data-plane path of c to d in T by $n_r \dots n_1 n_0$ (with $c = n_r, d = n_0$), and we distinguish two cases: either n_{r-1} has a data-plane path to d also in M or it does not.

Case 1: n_{r-1} has a data-plane path to d in M. Observe that n_{r-1} does not route through $n_r = c$ in M, since it does not route though c in T, and we chose c such that $M(c) \subseteq T(c)$ (*i.e.,* every node that routes through it in M uses the same next-hop in T as in M).

Next we claim that n_{r-1} announces some simple path to n_r in M. Observe that n_{r-1} exports some path to n_r in T. If $n_{r-1} = m$, then by construction it only exports paths in T on edges on which it announces some simple path in M, so we know that it must have announced some simple path to n_r in M. On the other hand, if $n_{r-1} \neq m$ then it uses all-or-nothing export rule, and since we assume that it has a path in M and we know that it exports a path in T, it follows that it must export some path also in M (which must be simple since only simple paths are announced by BGP-compliant strategies).

Let $n_{r-1}Rd$ be the path that n_{r-1} announces to $n_r = c$ in M. Next, we claim that the path $n_r n_{r-1}Rd$ contains a loop. Suppose it did not. Then by next-hop ranking we would get that $r_{n_r}(n_r n_{r-1}Rd) = r_{n_r}(n_r n_{r-1} \dots n_0)$. But we know that the path $n_r n_{r-1} \dots n_0$ is the T path of $n_r = c$, so from the crucial observation Lemma D.1 we know that $n_r n_{r-1}Rd$ must be ranked at least as high as any announcement that c received in M. By construction *c uses a different next-hop* than n_{r-1} in M, and thus it follows that the path the that c uses in M is ranked (strictly) lower than the path $n_r n_{r-1}Rd$. Now, since we assume that $c = n_r$ is stable in M, it follows that $c = n_r$ would have chosen to route through n_{r-1} also in M. This contradicts the fact that c indeed chose a different next-hop than n_{r-1} in M, and hence we conclude that the path $n_r n_{r-1}Rd$ contains a loop.

However, we argued above that the path $n_{r-1}Rd$ is simple. Thus, only way that $n_r n_{r-1}Rd$ could contain a loop is if $c = n_r$ itself appears somewhere on the path $n_{r-1}Rd$. But we argued above that n_{r-1} does not route through $c = n_r$ in T, so the path $n_{r-1}Rd$ is a false path. By the false-path lemma (Lemma C.1) it follows that this announced path has the form $n_{r-1}SmS'n_rS''d$ (since from the false path lemma S is a true path and $mS'n_rS''d$ is a false path, and $c = n_r$ must appear on the false path).

Next, observe that the S'' portion of the announced path cannot include m (since m appears before $c = n_r$ and $n_{r-1}SmS'n_rS''d$ is a simple path). But $c = n_r$ routes through m in M, and so invoking the false path lemma again implies that c must have announced some path that goes through m. It follows that $c = n_r$ did not announce the path $n_rS''d$, and so upon obtaining the announced path $mS'n_rS''d$ from n_{r-1}, $c = n_r$ would detect a false loop and raises an alarm.

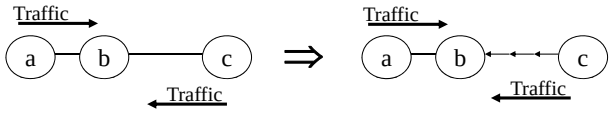


Figure 15: Lemma F.1.

Case 2: n_{r-1} has no path to d in M. Here we denote by n_i the node closest to $c = n_r$ on the T path (but not c itself) that does have a data-plane path to d also in M. We know that such n_i exists, since in particular d has the empty path to d in M. By definition of n_i, we have that n_{i+1} *does not have any data-plane path* to the destination in M. This implies (1) that $n_{i+1} \neq m$ (since m has a path to d in M), (2) that n_{i+1} does not use the same next-hop in M as it does in T, and (3) that n_i does not route through n_{i+1} in M.

Again, we argue that n_i must announce a simple path to n_{i+1} in M, since it announces some path to n_{i+1} in T. The argument is the same as in the previous case: either $n_i = m$ where this follows by construction, or $n_i \neq m$ where it follows from the all-or-nothing export and the fact that n_i has a data-plane path in M.

Also, we denote the path that n_i announces to n_{i+1} by n_iRd, and again argue that although this is a simple path, the path $n_{i+1}n_iRd$ must include a loop, or else n_{i+1} would have chosen it in M rather than having no data-plane path at all. (This follows because any path with next-hop n_i must be admitted at n_{i-1} due to next-hop policy, and from the assumption that n_{i+1} is stable in M.)

As in the previous case, we conclude that the announcement n_iRd must include n_{i+1}. However, we argued above that n_i does not route through n_{i+1} in the data plane. Thus, we have that n_iRd is a false path, and so combining this observation with the false-path lemma Lemma C.1 tells us that it is of the form $n_iSmS'n_{i-1}S''d$. But n_{i-1} did not announce the path $n_{i-1}S''d$ (since it has no data-plane path in M, and so it does not announce anything in M). Hence, n_{i+1} must raise an alarm upon receiving the announcement n_iRd from n_i. □

F. PROOFS: GAO-REXFORD NETWORKS

Before we start, we need 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 Figure 15. We also restate here a simple, useful lemma of the Gao-Rexford conditions proved by Gao, Griffin and Rexford in [14].

LEMMA F.1 (TRANSITIVE CUSTOMERS [14, THEOREM VII.4]). *If either the path $P = abRc$ or the path $P' = cR'ba$ is permitted, and if node a is not a customer of node b, then node c is a strict transitive customer of node b over the permitted path.*

We remark that even if not all the nodes in the AS graph use BGP-compliant strategies, Lemma F.1 still holds as long

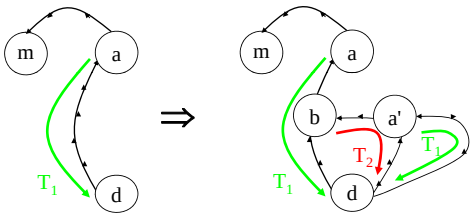


Figure 16: Proof of Lemma F.2

as all the nodes on the permitted path (except perhaps the last one, closest to the destination) use BGP-compliant strategies that obey the Gao-Rexford conditions.

We now prove the following helper lemma that we use to derive a contradiction in Theorem 6.1:

LEMMA F.2. *Consider an AS graph (that obeys GR1) where all nodes, except perhaps a single manipulator node m, use BGP-compliant strategies that obey the Gao-Rexford conditions (*i.e., *rankings obey GR3, export obeys GR2) Let T be the unique globally stable outcome when m follows some BGP-compliant strategy that obeys the Gao-Rexford conditions, and let M be a globally stable outcome that results from some other arbitrary strategy of m.*

If there is a node a in the network such that (1) a is a strict transitive customer of the manipulator m, (2) a uses a different path in M than in T, and (3) the destination d is a strict transitive customer of a along a's path in T. Then there is a different node $a' \neq a$ which is a strict transitive customer of a, such that a' also satisfies the conditions (1)-(3).

PROOF. Since a is a strict transitive customer of m, and the destination d is a strict transitive customer of a on a's T path, then the Topology condition GR1 implies that m cannot be on the path of a in T. Denote by b the node closest to the destination along a_i's path in T that uses a different path in M than in T (we know that such a b exists since in particular node a is such a node), and denote the paths of b in T and M by bQ_1d and bQ_2d, respectively.

Since all the nodes on the path Q_1d are honest and they all use that path in M, it follows that b must have received an announcement Q_1d from the first hop on that path in M, (and since M is a persistent outcome) and yet it chose a different path in M. We conclude that b's ranking has $r_b(M) > r_b(T)$. And since b's next hop in T is a customer, the Preferences condition GR3 implies that b's next hop in M must also be a customer. Applying Lemma F.1 we get that (a) node m cannot be on the path bQ_2d, or else it would have to be a strict transitive customer of b and we would have a customer-provider loop; and (b) since m is not on bQ_2d then the destination is a strict transitive customer of b along this path.

Let node a' be the node closest to the destination along the path bQ_2d that uses a different path in M than in T (again, we know it exists since b is one such node). Denote the paths of a' in T and M by $a'R_1d$ and $a'R_2d$, respectively. It follows that the path R_2d is also in the path assignment T. Notice that a' is also a strict transitive customer of the manipulator m, and that destination d is a strict transitive customer of a' along the path R_2d. Since all the nodes on the path R_2d uses this path also in T, and since a' received an announcement for this path in M (because it uses this path in M) then a' must have received an announcement R_2d in T also (since T is a globally stable outcome). Yet a' chose a different path in T. We conclude that the ranking of a' has $r_{a'}(T) > r_{a'}(M)$, which also implies that $a' \neq b$.

Since $r_{a'}(T) > r_{a'}(M)$ and since the next hop after a' on the path $a'R_2d$ in M is a customer of a', the Preferences condition GR3 implies that the next hop after a' on the path $a'R_1d$ in T must also be a customer. Then, we can apply Lemma F.1 to find that the destination is a strict transitive customer of a' along the path $a'R_1d$ in T.

We established that a' satisfies the conditions (1)-(3), and we also know that b is a transitive customer of a (or a itself), a' is a strict transitive customer of b, and $a' \neq b$. It follows that $a' \neq a$, since otherwise we would have a customer-provider loop in the graph. □

We are now ready to prove the main result of this section.

THEOREM 6.1 *Consider an AS graph where the valuations are policy consistent and contain no dispute wheels, and the valuations and attraction functions of all nodes obey the Gao-Rexford conditions and AT4, and all attractees use next-hop policy with their providers and peers. Suppose that all nodes, except a single manipulator node m, use BGP-compliant strategies that obey consistent export and GR2 export, and moreover set their ranking equal to their valuations ($r_n(\cdot) \equiv v_n(\cdot)$ for every node n). Suppose further that the network has path or loop verification.*

Then there exists a BGP compliant strategy for m that uses $r_m(\cdot) \equiv v_m(\cdot)$ and obeys GR2 and consistent export, which obtains the best possible globally stable *outcome in terms of the utility function of m. In particular, setting $r_m(\cdot) \equiv v_m(\cdot)$ and exporting all paths to customers and no paths to providers and peers is one optimal strategy.*

PROOF. Let M be a globally stable outcome that results from some arbitrary strategy for m. We assume M that $u_m(M) > -\infty$ (or else any BGP compliant strategy for m will do).

Now fix a BGP compliant strategy for m where $r_m \equiv v_m$, and where m (i) exports all paths to every *customer* that routes through it in M and (ii) exports no paths to nodes that are not its customers. (Note that this export rule obeys GR2.) The rest of m's export policy can be arbitrary, as long as it complies with consistent export and with GR2.

Clearly this strategy is BGP compliant, and when m uses this strategy then the ranking functions of all nodes contain no dispute wheels (since they are set equal to the valuation functions). The results of Griffin *et al.* [22] imply that the protocol converges to a unique globally stable outcome T. We prove next that the utility of m in T is at least as high as in M.

Our proof is by contradiction. We assume that $u_m(M) > u_m(T)$, and prove a sequence of claims that together imply that the conditions of Lemma F.2 must hold in this graph. We then repeatedly apply Lemma F.2 to show that the graph contains a customer-provider cycle, and thus violates the Topology condition GR1.

Denote the data-plane paths of m to the destination in T and M by mR_1 and mR_2, respectively.

CLAIM F.3. *The is a node c that is an attractee of m that routes directly through m in M but not in T.*

PROOF. Since the data plane path R_2 used by m in M is permitted at all nodes on R_2, and since all these nodes are honest (otherwise mR_2 would not be a simple path, and $u_m(M) = -\infty$) know that mR_2 is permitted also in T. Note that T satisfies all the conditions of Lemma C.3, since all nodes use consistent export and set their ranking equal to their valuations (so the rankings have no dispute wheel and are policy consistent). So we know that T is locally optimal everywhere. In particular, since the data-plane path of m in M is permitted also in T (since it only goes through honest nodes) then $v_m(T) \geq v_m(M)$. But we assumed that $u_m(M) > u_m(T)$, so we must have $\alpha_m(M) > \alpha_m(T)$, which means that m gained AT4 attraction in M that it did not have in T. □

CLAIM F.4. *Node c has a data-plane path to the destination in T, and moreover $r_c(T) > r_c(M)$.*

(Note that this claim *does not follow* from Lemma C.3, since there could be paths that are "permitted" in M but not in T: recall that m's export policy in T dictates that it does not announce anything to its providers and peers, whereas it is possible that m did announce something to them in M.)

PROOF. Assume toward contradiction that $r_c(T) \leq r_c(M)$. Since c was defined as a node that uses m as next-hop in M but not in T, then the inequality has to be strict. Since c is an attractee of m (and therefore its customer), then c must use next-hop policy with m. Since c is a customer that routes through m in M, then the export policy of m in T includes exporting all to c. Since m is honest in T, we know that m announces to c the path mR_1 that it uses in T.

If mR_1 was a simple path, then from next-hop policy we have that $r_c(cmR_1) = r_c(cmR_2) > r_c(T)$, which contradicts the fact that c is stable in T (it should have chosen the better available path cmR_1). So we know that mR_1 must have a loop in it, but mR_1 is a simple *path* (being the data-plane path of m), so it must be that c appears on that *path* (which in particular implies that c has a data-plane path in T). We can re-write the path that m takes in T as $R_1 = R_1' cnQ$, as depicted in Figure 17(a).

Since c is a customer of m, it follows from the Topology condition GR1 that m cannot be a strict transitive customer of c along the path $mR_1'c$. Hence there are adjacent nodes between m and c on the path R_1' (call them a, b) such that a is not a customer of b. Since the path $mR_1'cnQd$ is permitted (because it is the data plane path in T) and since all nodes behave honestly in T, we can apply Lemma F.1 to conclude that d is a transitive customer of b along this path. In particular it means that n is a customer of c. (Notice that this is true even if $n = d$.) But this violates the Preferences condition GR3, since we assumed that $r_c(M) = r_c(cmR_2) \geq r_c(cnQd) = r_c(T)$ where m is a provider of c and n is its customer. □

From now on, let us denote the path of c to the destination in T by $n_0 n_1 \dots n_t$ (where $c = n_0$ and $d = n_t$), and remember that c uses m as a next-hop in M but not in T, so $n_1 \neq m$.

From Claim F.4 we can also conclude that $n_1 \neq d$: Otherwise ($d = n \neq m$), the T-path dc would be available to c also in M, and so c would take it (since we just proved that the T path is ranked higher than then M path of c) and this would contradict the stability of c in outcome M. Next we prove that m is not on the T-path of c.

CLAIM F.5. *c does not route through m in T.*

PROOF. For the sake of contradiction, suppose that m is on the T-path of c, namely $m = n_j$ for some $1 \leq j \leq t$. This means in particular that $m = n_j$ exports some path to n_{j-1} in T, so n_{j-1} is a customer of m. (Recall that m only export paths in T to its customers.) Applying Lemma F.1 we find that c is a strict transitive customer of m along c's path in T. In particular, $c = n_0$ is a customer of n_1 and n_1 is a customer of n_2. Now since the valuations of n_1 obey GR3, we deduce that $v_{n_1}(n_1 n_2 \dots d) < v_{n_1}(n_1 c \dots d)$. However, from Claim F.4 and the fact that c uses next hop policy with all its providers, we have $v_c(cn_1 \dots d) \geq v_c(cm \dots d)$. Furthermore, the inequality is strict, since $m \neq n_1$. Hence there is a (2-pivot) dispute wheel between c and n and we have arrived at a contradiction. □

CLAIM F.6. *The node n_1 uses a different (data-plane) path for its traffic in M than in T.*

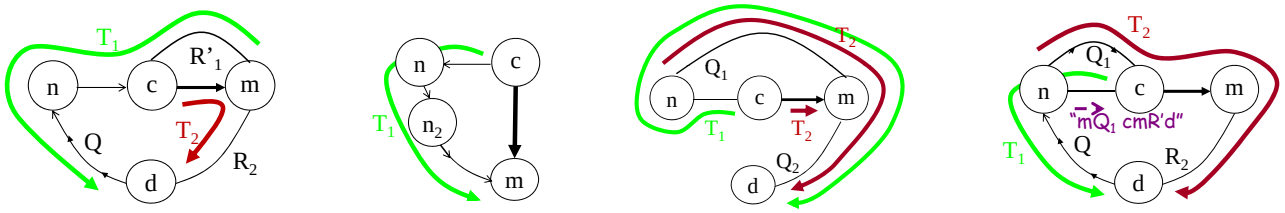
PROOF. Assume toward contradiction that n_1 uses the T-path $n_1 n_2 \dots n_t = d$ also in M. Below we also denote this path by $n_1 Q$. From Claim F.4 we know that $r_c(cmR_2) < r_c(cn_1 Q)$, so we know that n_1 does not announce $n_1 Q$ to $c = n_0$ in M (or else c would have used this path). But we know that n_1 exports the path $n_1 Q$ to c in T, and that n_1 is honest, so it would have exported this path to c in M if it had chosen it. We deduce that n_1 had chosen a different path in the control plane in M (even though it actually routes on $n_1 Q$ in the data plane). In other words, n had chosen a false path in M. From the false path lemma (Lemma C.1), we have that both the false-path in the control plane and the data-plane path must include m. But this is a contradiction, since we assume that n uses the same data-plane path in both M and T, and from Claim F.5 we know that m is not on the data-plane path of n_1 in T. □

CLAIM F.7. *Node n_1 announces a path to $c = n_0$ in M.*

PROOF. For every node n_i on the T-path $n_1 \dots n_{t-1} n_t$, we denote the control-plane path that n_i chooses in M (if any) by $n_i Q_i$. We now show by backward induction over $i = t \dots 2$ that (i) node n_i ranks $n_i Q_i$ at least as high as $n_i n_{i+1} \dots n_t$, and (ii) n_i announces the path $n_i Q_i$ to n_{i-1}. For the proof below, recall that $n_i \neq m$ for all i (due to Claim F.5), so all the n_i's use policy-consistent ranking and consistent export also in M.

The base case $n_t = d$ is obvious. For the induction case, assume that the two conditions above hold for n_{i+1} and we prove for n_i. We have two cases: either $n_{i+1} Q_{i+1}$ goes through n_i or it does not.

- If $n_{i+1} Q_{i+1}$ does not go through n_i, then from policy consistency (and since n_{i+1} prefers this path to $n_{i+1} \dots n_t$) we have that also n_i must prefer $n_i n_{i+1} Q_{i+1}$ over $n_i n_{i+1} \dots n_t$. Moreover, since the path $n_i n_{i+1} Q_{i+1}$ is available to n_i in M (as we assume that n_{i+1} announces it), and since M is a globally stable outcome, then n_i must choose a control-plane path in M that is ranked at least as high. We conclude that $r_{n_i}(n_i Q_i) \geq r_{n_i}(n_i n_{i+1} Q_{i+1}) \geq r_{n_i}(n_i n_{i+1} \dots n_t)$.



(a) Claim F.4 (b) Claim F.5 (c) Claim F.6 (d) Claim F.8

Figure 17: Pictorial representation of the proof of Theorem 6.1

- Suppose that $n_{i+1}Q_{i+1}$ does go through n_i. Then rewrite this path as $n_{i+1}Q_{i+1} = n_{i+1}R_{i+1}n_iQ'_i$. By the induction hypothesis, n_{i+1} announces this path to n_i, and also prefers it over $n_{i+1}\dots n_t$. Since n_i is honest and the network uses loop verification, it must be the case that n_i actually announces the path $n_iQ'_i$ (or else n_i would have raised an alarm, which would have set the utility of m in this outcome to $-\infty$). Hence n_i must have chosen $n_iQ'_i$ in the control plane in M, in other words we have $Q'_i = Q_i$.

  We claim that n_i must prefer n_iQ_i over $n_in_{i+1}\dots n_t$; otherwise we would have a dispute wheel between n_i and n_{i+1}, since n_{i+1} prefers $n_{i+1}R_{i+1}n_iQ_i$ over $n_{i+1}\dots n_t$.

In either case, we know that n_i prefers n_iQ_i over $n_in_{i+1}\dots n_t$. Since n_i uses consistent export, and since it announces $n_in_{i+1}\dots n_t$ to n_{i-1} in T, then it has to announce also n_iQ_i to n_{i-1} in M. □

Claim F.8. *The node n_1 is a strict transitive customer of m, and the destination d is a strict transitive customer of n_1 over the data-plane path of n_1 in T.*

Proof. Recall that we denote the data-plane path of n_1 in T by n_1Q. If n_1 is a direct customer of c then the first part of the lemma follows trivially (since c is a customer of m), and the second part follows by applying Lemma F.1 to the permitted path cn_1Q in T.

If n_1 is not a customer of c, then c must use next hop policy with n_1. From Claim F.7, we know that n_1 announces a path to c in M. Let n_1Q' be that path that n_1 announces to c in the manipulated outcome M. If the path n_1Q' does not go through c, then we have

$$r_c(cn_1Q') = r_c(cn_1Q) > r_c(cmR_2)$$

where the equality follows from next-hop policy and the inequality is from Claim F.4. But this is impossible, since if this was the case then c would have chosen n_1 as its next-hop also in M. Thus, the path n_1Q' must go through c.

Next denote by cmR' the control-plane path that c chooses in M. By loop-verification, it must be the case that cmR' is a suffix of n_1Q' (or else c would have raised an alarm and the utility of m would be set to $-\infty$). So re-write n_1Q' as $n_1Q'_1cmR'$. The path Q'_1 does not include m, or else n_1 wouldn't have chosen this path since it would contain a routing loop through m. Hence the partial path $n_1Q'_1cm$ must be the data-plane path that is used in M (and in particular it must be a permitted path). Since c is a customer of m, then we can apply Lemma F.1 to conclude that n_1 is a strict transitive customer of c (and therefore also of m).

Moreover, since n_1 is a strict transitive customer of c then the Topology condition GR1 says that it cannot be a provider of c. We assumed that n_1 is also not a customer of c, so they must be peers. We can now apply Lemma F.1 to the permitted T path cn_1Q, to conclude that the destination d is a strict transitive customer of n_1 over this path. □

Claims F.6 and F.8 established the existence of a node $a_0 = n_1$ which is (1) a strict transitive customer of the manipulator m, and where (2) a_0 uses a different path in M than in T, and (3) the destination d is a strict transitive customer of a_0 along its data-plane path in T. Lemma F.2 asserts that there must be another node $a_1 \neq a_0$ which is a strict transitive customer of a_0, where a_1 also satisfies the conditions (1)-(3). Repeated applications of this lemma thus give us a sequence of nodes $a_1, a_2, \dots$ such that for all i $a_i \neq a_{i-1}$ and a_i is a strict transitive customer of a_{i-1} (and they all satisfy the same conditions). Since there are a finite number of nodes in the AS graph, eventually one of the nodes in the sequence will repeat, resulting in a customer-provider cycle and violating the Topology condition GR1.

We see that our assumption that $u_m(M) > u_m(T)$ leads to a contradiction, thus concluding the proof of Theorem 6.1. □