Towards a Logic for Wide-Area Internet Routing

Nick Feamster and Hari Balakrishnan
MIT Laboratory for Computer Science
200 Technology Square, Cambridge, MA 02139
{feamster,hari}@lcs.mit.edu

Abstract

Interdomain routing is a massive distributed computing task that propagates topological information for global reachability. Today's interdomain routing protocol, BGP4, is exceedingly complex because the wide variety of goals that it must meet—including fast convergence, failure resilience, scalability, policy expression, and global reachability—are accomplished by mechanisms that have complicated interactions and unintended side effects. The complexity of wide-area routing configuration and protocol dynamics requires mechanisms for expressing wide-area routing that adhere to a set of logical rules. We propose a set of rules, called the *routing logic*, which can be used to determine whether a routing protocol satisfies various properties. We demonstrate how this logic can aid in analyzing the behavior of BGP4 under various configurations. We also speculate on how the logic can be used to analyze existing configuration in real-world networks, synthesize network-wide router configuration from a high-level policy language, and assist protocol designers in reasoning about new routing protocols.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Routing protocols, Protocol verification

General Terms

Design, Performance, Reliability

1. Motivation

Interdomain routing on the Internet is staggeringly complex. Routers on the Internet participate in a massive distributed computing task that propagates topological information for path selection. The complexity of the task results from the many distinct goals that must be met: fast convergence to correct loop-free paths to all destinations under static and dynamic conditions; resilience to congestion, packet loss, and failures; scaling to large numbers of networks and end hosts; and, above all, providing global connectivity among autonomous, financially competing and mutually distrusting domains.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ACM SIGCOMM 2003 Workshops, August 25&27, 2003, Karlsruhe, Germany.
Copyright 2003 ACM 1-58113-748-6/03/0008 ...$5.00.

BGP's complexity stems not from its deceptively simple specification [35], but rather from its dynamic behavior during operation, as well as the vast possibilities for configuration. Prior work has highlighted many aspects of wide-area routing that result in the following complex, unexpected, or undesirable properties:

- *Poor integrity.* BGP is vulnerable to masquerading, denial of service, and data integrity attacks [5] and is also subject to frequent misconfiguration [29].
- *Slow convergence.* Path instability commonly results in delayed convergence [27], which is often slowed further by the unintended side effects of other performance tweaks, such as route flap dampening [30]. BGP's performance under congestion or routing instability is not well-understood.
- *Divergence.* BGP's policy-based nature can give rise to configurations that are guaranteed to diverge [20].
- *Unpredictability.* Because of the distributed, asynchronous nature of BGP, precisely predicting the effects of a configuration change is extremely challenging [14].
- *Poor control of information flow.* Certain routing policies or BGP implementations may expose information that is not intended to be public knowledge, such as peering and transit relationships. For example, BGP routing messages expose information about topology and hierarchical relationships among Internet service providers [15].

The complexity of routing configuration and protocol dynamics mandates a mechanism for understanding and manipulating Internet routing at a higher level of abstraction. We believe that the time is ripe for a reconsideration of the current approach to interdomain routing, based on a more formal approach to the problem and building on our experience as a community with BGP4 over the past several years.

Previous work in wide-area protocol design has focused on specific modifications to BGP that fix a particular problem but often spur unintended negative side effects. Furthermore, designers of new wide-area routing protocols require a mechanism that enables them to reason about the circumstances under which the protocol will behave "correctly." To help reason about modifications to existing routing protocols and to aid in the sound design of new ones in the future, we propose that routing protocols be classified in terms of the following properties, each of which expresses an important aspect of wide-area routing:

- *Validity*. The existence of a route to a destination implies that a packet sent along the corresponding path will eventually reach the intended destination.

- *Visibility*. The existence of a path to a destination from some origin implies that that origin knows about a corresponding route to the destination.
- *Safety*. Given a set of routes and a set of policies, an assignment of routes must exist such that no participant wants to change its route in response to other participants' routes (Griffin and Wilfong's Stable Paths Problem and General Stable Paths Problem [20, 22]).
- *Determinism*. Given a set of possible routes and a set of policies, the routing protocol should always arrive at the same predictable set of routes. This set of routes should be independent of the order in which the possible routes arrive.
- *Information-flow control*. Routing messages should not expose more information than is necessary to achieve the above requirements, subject to some information flow specification, such as noninterference [17].[1]

For each property, we formally define a minimal set of rules that, if satisfied, imply that the property is satisfied. We call this set of rules, together with the logic to reason about them, the *routing logic*. The routing logic helps protocol designers reason about new routing protocols and prove statements about modifications to existing ones. Our goal is to improve our understanding of the behavior of complex protocols using better tools than we currently have.

This set of rules is by no means complete; indeed, we do not consider issues like scalability or the ability to perform traffic load balancing via traffic engineering. However, the logic can be used to determine if any techniques used to achieve these goals affect the chosen properties. We stress that the logic is a set of rules for reasoning about *properties* of routing protocols, as opposed to a specification of requirements. It may be possible (and even reasonable) for a protocol to violate one or more of these properties under certain circumstances; the logic simply provides a framework for reasoning about when these violations arise.

As a simple example, observe that validity requires that some route advertisement imply reachability to a superset of that destination. A protocol violates this property when a route advertisement exists for a destination that is not reachable. One example of such a violation is delayed convergence; another is the unreachability of a subnet within an aggregated prefix.

We highlight how the routing logic can be used to reason about both BGP configuration, as well as proposed modifications to BGP itself. We show that verifying that an arbitrary route reflector configuration satisfies validity is NP-complete, and that several simple protocol modifications can guarantee route validity under certain circumstances. We use the routing logic to analyze several proposed modifications to BGP; for example, we show that the Safe Path Vector Protocol [21] modifications to BGP improve safety but can violate information flow policies.

BGP's design and implementation makes reasoning about various properties surprisingly difficult. We show that BGP's dependence on other routing protocols such as IGP and its inability to check route advertisements for consistency makes the protocol very difficult to reason about. In response, we propose that an alternative architecture based on the routing logic may be able to circumvent some of these problems.

Previous work has focused on specific problems with BGP, without considering broader implications or fundamental problems. In this paper, we present a logic that concisely describes fundamental problems in wide-area routing and provides insights for considering wide-area routing at a higher level of abstraction. We believe this logic can enable configuration analysis to catch mistakes and verify that certain properties are satisfied.

While this paper primarily focuses on the potential for the routing logic as an *analysis* tool, we also speculate on the potential uses of the logic for *synthesis* of BGP configuration and new protocol designs. We believe that the logic can be used a framework for high-level policy specification that preserves the semantics of low-level configuration but has verifiable properties. Additionally, incorporating insights from the logic into routing protocol design can speed convergence, detect routing pathologies more quickly, enforce information flow control, and facilitate the design of wide-area routing protocols that conform to high-level specifications.

2. A Routing Logic

The routing logic presented in this section is a set of rules that facilitates reasoning about whether, and under what circumstances, a routing protocol satisfies a particular property. While we believe that the logic can be used to understand other types of routing protocols (e.g., for mobility, etc.), we limit our focus to interdomain routing. We also examine the properties of the routing logic in the context of BGP.

2.1 Overview and Definitions

After providing a brief overview of the routing logic, we present the terminology for the logic and introduce the concept of hierarchical routing scopes.

2.1.1 Overview

The routing logic allows network operators and protocol designers to reason about properties of routing protocols. To determine whether a routing protocol satisfies a particular property, the routing logic requires the following inputs: (1) a specification of how the protocol behaves and (2) a specification of the protocol configuration. Protocol configuration entails both policy configuration (i.e., which routes are preferred over others) and general configuration, such as which routers exchange routing information with each other. The routing logic then determines whether the routing protocol satisfies the conditions associated with that rule.

We also believe that the logic will be useful for automated configuration analysis and generation; we explore these possibilities in Section 4. However, this type of automated reasoning requires either a thorough abstract specification of BGP's operation or a set of sufficient conditions that can be more easily tested than the rules themselves. Given the complex operations and dynamics of BGP, the most immediate benefit of the logic is providing a framework for deriving and reasoning about these sufficient conditions, as we demonstrate in Section 3. However, as we discuss in Section 4, alternative protocol designs (including simplified versions of BGP itself) may lend themselves to automated analysis more easily.

Our current version of the routing logic does not incorporate any notion of time. Because any distributed routing protocol will have invalid routes while in a transient state, it would seem that the rules for validity and visibility require a temporal dimension. However, as we will discuss in Section 2.2, the validity rule highlights the *propagation* of invalid routes (e.g., during path exploration), rather than simply the existence of invalid routes in the system during transient states (as might happen in OSPF during flooding and distributed shortest paths computation). Similarly, we apply the visibility rule to examples of BGP in the steady-state. While including time in the routing logic could facilitate reasoning about some rout-

[1] Noninterference requires that information (e.g., routing messages, peering and transit relationships) at a particular security level does not affect how the routing protocol is observed by entities at a lower security level. We describe this further in Section 2.4.

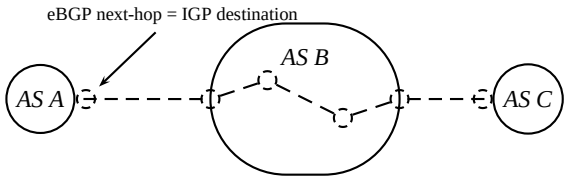


Figure 1: Routing domains are organized hierarchically. A scope i next-hop is a scope $i+1$ destination (or destination set). In this example for BGP, scope 0 (eBGP) is shown in solid lines, and scope 1 (IGP) is shown in dashed lines. The eBGP next-hop is the IGP destination.

ing protocol aspects, such as the effects of various timers, we believe that, even without a notion of time, the routing logic provides the ability to reason about many fundamental properties.

Determining whether a routing protocol configuration satisfies a particular property may sometimes require some amount of global knowledge. For example, determining whether a routing protocol can *potentially* violate the safety property merely requires a counterexample; however, determining whether an actual configuration will result in such a violation requires some knowledge about the policies of other autonomous systems. Similarly, the routing logic can be used in the design phase to determine whether a routing protocol can potentially have invalid routes; however, verifying the validity of a particular route advertisement requires knowledge about whether other autonomous systems along the advertised path actually have a corresponding route to the destination.

2.1.2 Terminology

Routing allows a *participant* in some *routing domain* to discover a *route* to a *destination*. A *participant* is an entity that advertises or receives routing messages. A *routing domain* is a group of one or more participants that behave according to one administrative policy. In BGP, an autonomous system (AS) can be thought of as a *routing domain*. A routing domain may have multiple participants; for example, in BGP, a single AS may have many BGP-speaking routers that all participate in BGP sessions with each other and with routers in other ASes.

The *route* may refer to a physical path (as in MPLS), a path at the IP layer (as in BGP), or a path in an overlay network (as in RON [2]). In addition to the destination, each route contains two fields: the *next-hop*, which names a location (e.g., by IP address or some other node identifier) to forward packets along that route; and the *next-RD*, which is the next routing domain along that route to the destination. The route contains other information, such as the source and destination for which the route is valid.

A *destination* might refer to a host (specified by an IP address or prefix), an overlay node (specified by a node identifier), or a logical host (specified by a DNS name). We use the term *destination-set* to refer to a set of nodes that share a route.[2] An IP prefix is an example of such a set, because the prefix refers to a group of nodes (specified by IP addresses) that all use the same route. When discussing certain aspects of the logic, we will also refer to paths. A *path* is a sequence of participants from one participant to a destination.

2.1.3 Hierarchical Routing Scopes

Our logic organizes routing domains into hierarchical levels called *scopes*. A routing protocol in scope i forwards packets along a path via the scope i next-hop for that path. The scope i routing protocol uses a scope $i+1$ path to reach the scope i next-hop.

Figure 1 shows an example of routing scopes. Let BGP be a protocol at scope 0 that has scope 0 routing domains (ASes), scope 0 next-hops (which is the BGP next-hop), and scope 0 destinations (the ultimate destination, since there is no higher scope). A scope 0 routing domain in turn contains a scope 1 routing protocol (an IGP, such as OSPF) with scope 1 routing domains (routers within that AS), scope 1 next-hops (the hop to the next router in the IP path) and scope 1 destinations (the first hop into the scope 0 routing domain, or the BGP next-hop). In this paper, we will often refer to BGP as a scope 0 routing domain and IGP as a scope 1 routing domain.[3]

We highlight two subtle points. First, a scope $i-1$ next-hop is a scope i destination. Second, scope i properties may depend on scope $i+1$ properties, but not vice versa; for example, scope 0 validity requires scope 1 validity.[4]

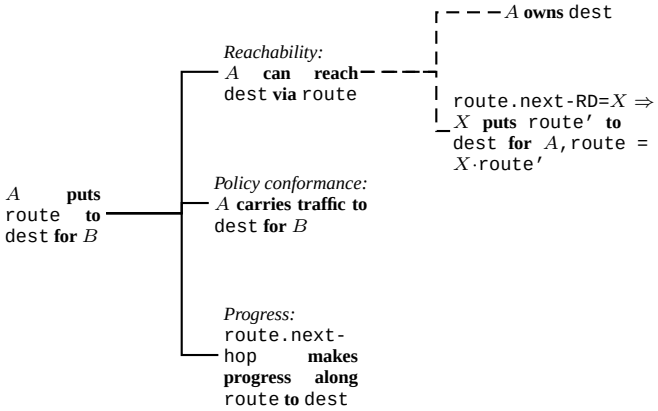


Figure 2: The validity rule. A valid route (corresponding to a "put" statement) implies that the expression shown by this tree evaluates to a true statement. Dashed lines indicate an "or" condition.

2.2 Validity and Visibility

BGP experiences slow convergence when routing faults occur due to subsequent exploration of invalid paths [27, 39]. Sometimes, policy misconfiguration, implementation bugs, or even routine maintenance, can cause a large number of invalid routes to leak into the global Internet [13, 29]. In other cases, malicious entities hijack routes to certain destinations in order to mount denial of service attacks [37]. Each of these incidents results in invalid routes leaking into the global Internet. The routing logic should define the circumstances under which BGP may end up advertising invalid routes.

2.2.1 Validity and Visibility Rules

Ideally, the routes that a participant learns about should be *valid*; that is, sending a packet along a route to its destination should result in the packet eventually reaching the intended destination (ignoring packet loss). In wide-area routing, the validity of a route implies the following three properties:

[2]Where ambiguity is not an issue, we will use "destination" when referring to destination-sets.

[3]Our routing logic could be used to evaluate wide area routing protocols running on top of BGP (e.g., overlay based protocols [2]). In this case, the overlay protocol would be scope 0, BGP would be scope 1, and so forth.

[4]This framework incorporates failures at lower scopes. A failure at scope $j > i$ violates validity for scope j, a necessary precondition for validity at scope i.

- *Reachability.* The route must transport packets to their intended destination.
- *Policy conformance.* The route must conform to routing policies, such as peering and transit agreements.
- *Progress.* The next-hop specified by a route must reduce the total distance[5] to the destination along that path.

In a wide-area routing protocol that satisfies *validity*, all advertised routes should satisfy these three properties. We say that a valid *path* exists at scope i if there exists a sequence of scope i participants to the destination that satisfy reachability, policy conformance, and progress. In other words, a routing protocol satisfies *validity* if, for all destinations, the existence of a route to a destination implies the existence of a corresponding path to that destination ($\forall d : \exists route_d \Rightarrow \exists path_d$).

If a routing protocol satisfies *visibility*, then for all destinations, the existence of a valid path to a destination implies the existence of a valid route to that destination ($\forall d : \exists path_d \Rightarrow \exists route_d$). Additionally, for a routing protocol at scope i to be valid, the routing protocols at scopes $j > i$ must satisfy validity.

The conditions for the *validity rule* are illustrated in Figure 2. Each routing announcement corresponds to a **puts...to..for**[6] statement, which has a number of implied conditions. For example, if A advertises a particular route to B, we say that A **puts** `route` **to** `dest` **for** B, where `route` is a particular route to a destination (minimally including the next-hop and next-RD), and `dest` is a destination-set. Thus, a routing domain's route advertisement corresponds to a **puts...to..for** assertion. For this assertion to be true, all of the implied conditions must hold.

The *reachability* condition (expressed with the **can reach** keyword in Figure 2) is satisfied if either participant A has ownership over the destination (which can be verified by a routing registry or cryptographic authentication, such as S-BGP [26]), or if some other participant has issued a **put** with the appropriate AS path for A. A simple inductive argument can show that the latter reachability condition also guarantees the validity of the AS path, because the condition implies that the route to a destination has been advertised (and is thus considered "valid") by all participants along the path to that destination.

Policy conformance requires A to be willing to carry traffic to the destination-set `dest` for B (e.g., if B were paying A for transit). In the absence of bugs, A would not advertise a route to B if it were not willing to carry traffic for B.

Progress says that a packet destined for `dest` forwarded to `route.next-hop` must reduce the distance to `dest` along the path specified by `route`. For BGP, distance to the (scope 0) destination is measured in ASes (i.e., scope 0 routing domains). A packet forwarded to the `next-hop` specified in the BGP message should eventually result in the packet being forwarded to the next AS specified in the AS path, as this is sufficient to reduce the distance to the scope 0 destination, thus satisfying progress.

For a scope i routing protocol to satisfy progress, no scope i route must ever traverse any scope i participant twice. Note that a routing protocol at scope $k < i$ can create a loop in a scope i *path* but still result in correct routing (as can happen with tunneling). As long as the routing protocol at all scopes is valid, then progress will be satisfied. We will see in the next section, however, that interactions between routing scopes can violate validity in a particular routing scope.

[5]The distance metric depends on the scope of the routing protocol. The distance from a scope i routing participant to a scope i destination is the sum of the costs of each hop along the path from that participant to the scope i destination. For example, scope 0 distance is the number of AS hops; scope 1 distance, however, is the IGP metric.

[6]We denote keywords of the routing logic in **boldface**.

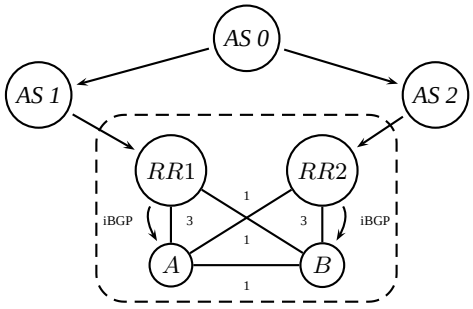


Figure 3: BGP configurations that are inconsistent with the underlying IGP can result in persistent forwarding loops [12]. Small numbers indicate IGP metrics.

2.2.2 Validity and Visibility in BGP

At the beginning of Section 2.2, we noted many cases in which BGP does not satisfy validity. To show how the validity and visibility rules can be used to prove properties about wide-area routing, we illustrate that the fundamental operation of BGP with route reflection can violate validity. In Section 3, we use the logic to prove properties about this.

Route reflection is a common technique for scalably achieving consistent BGP routes within an autonomous system. The routers within an autonomous system are either route reflectors or route reflector clients; the route reflector server reflects routes received from non-clients to all of its clients, as well as routes from a client to all other clients and non-clients. In this fashion, an autonomous system can achieve consistency without requiring pairwise BGP sessions. Nevertheless, poorly designed route reflector configurations can lead to invalid routes in the form of a persistent forwarding loop. Furthermore, certain route reflector configurations can result in a router not having a route to a destination, even when a valid path to that destination exists.

BGP's coupling with intradomain routing can easily result in accidental progress violations—for example, if a route reflector client is configured such that the route reflector is not on the shortest path to the client's egress for that route [12]; if route reflector clusters are configured such that the IGP metrics for intra-cluster links are higher than those for inter-cluster links; or in general when multiple hierarchies of route reflectors are configured carelessly [32].

In Figure 3 (adapted from [12]), routers are assigned to route reflectors that are not on the shortest IGP path to the egress for a particular destination. This situation shows a persistent routing loop. A thinks it can get to the scope 0 destination, AS 0, via the scope 0 next-hop advertised by $RR1$, but its scope 1 next-hop along its level 1 path is B, because this is the shortest IGP path to $RR1$. B tries to get to the AS 0 via a *different* scope 0 next-hop through a level 1 path through $RR2$ via A, and the process repeats.

In this example, route reflection causes BGP to violate validity. The fundamental problem with route reflectors is that they cause complex interactions between the different scopes of the routing hierarchy. In trying to reach a scope 1 destination, the scope 1 participants A and B are using a scope 0 destination to make a decision about the scope 1 next-hop! We show how the routing logic can formally express this violation of validity in Section 3 (Theorem 3.1). We also show that verifying progress for an arbitrary configuration of route reflectors is NP-complete (Theorem 3.2), and we prove that specific alternative route reflector configurations satisfy

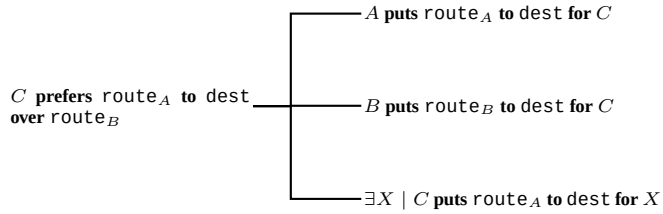


Figure 4: Preference rule for inferring participant C's preferences to destination `dest` based on available routes and path selection. "puts" assertions in this rule can alternatively be "replaces" assertions.

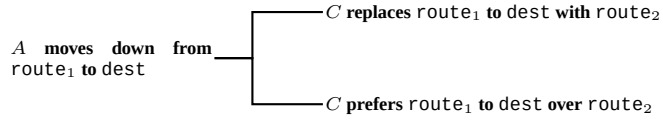


Figure 5: Selection rule for inferring realized dispute cycles.

the progress rule (Theorems 3.3, 3.4, and 3.5).

Route reflectors improve scaling within an AS, but may also violate visibility. For example, if a route reflector drops a BGP session with one of its clients, then that client will lose a route to a particular destination, even though a valid path exists (e.g., through an IGP path to another client that has a different route reflector). In Section 3, we show that for any route reflector configuration that is not a full mesh, visibility can be violated; we also discuss why the use of a full mesh in iBGP is a sufficient condition for visibility.

2.3 Safety and Determinism

Previous work has noted that because BGP makes best-path decisions based on local policy, certain sets of policies can result in routing configurations with no stable solution [16, 20]. That is, one AS may change its choice for a best route to a destination and readvertise that change to its neighbors, which will cause neighbors to change their choices for best routes, which will in turn affect the original AS's choice for a best route, and so forth. This situation is called a policy oscillation, or a *dispute cycle*. The potential for dispute cycles implies that BGP does not satisfy *safety*.

Careless router configuration can result in a router's best route depending on the order in which routes arrive or other nondeterministic factors, such as the age of the routing advertisement [9, 10]. Others have noted that the multi-exit discriminator (MED) attribute prevents one node from producing a single linear ranking of paths [14, 22]. In these cases, BGP does not satisfy *determinism*.

2.3.1 Preference and Selection Rules

The routing logic should concisely express whether a routing protocol has the potential to violate safety. In the case of policy-based wide-area routing protocols, safety means that execution of the protocol will always converge to a solution to Griffin and Wilfong's Stable Paths Problem (SPP) [20] and General Stable Paths Problem (GSPP) [22]. The absence of a dispute cycle guarantees that the protocol will always converge to a unique solution [25].

At a basic level, routing policies are expressed in terms of preferences (e.g., a router prefers routes with paths with a certain next-hop AS, a three-hop path through a certain AS, etc.). Policy cycles can be detected either statically (by analysis of the routing policies of each autonomous system) or dynamically (by observation of routing protocol behavior).

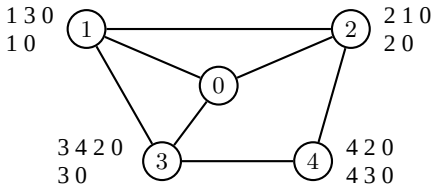


Figure 6: BAD GADGET [20]. Paths next to each node indicate ranking from most to least preferred.

A routing protocol that satisfies the following properties will satisfy safety:

- *Preference.* If a participant chooses a particular route as its best route, the participant readvertises that route. (Otherwise, the notion of preference is meaningless.)
- *No route history cycles.* Previous work has shown that the non-existence of a route history cycle is sufficient to guarantee safety [21].

The routing logic must also express certain properties with respect to route selection. If a routing protocol selects its routes independently of message arrival order and independently of other existing routes, we say that the routing protocol satisfies *determinism*. Specifically, a routing protocol that satisfies the preference requirement will also satisfy determinism, given that the additional following properties hold:

- *Time immunity.* A participant's relative ranking of two routes to a destination is independent of the order in which those routes arrive. Let σ be a participant's path selection algorithm, and let α_t and β_t be routes that the participant receives at time t. Time immunity requires that $\sigma(\{\alpha_t, \beta_{t+1}\}) = \sigma(\{\alpha_{t+1}, \beta_t\})$.
- *Set immunity.* A participant's relative ranking of two routes to a destination is independent of other routes to that destination. If $\lambda_{\mathcal{R}}$ is a participant's ranking function of routes given the set of advertised routes $\mathcal{R}$ and $\alpha, \beta \in \mathcal{R}$, then $\lambda_{\mathcal{R}}(\alpha) > \lambda_{\mathcal{R}}(\beta) \Rightarrow \lambda_{\mathcal{R}'}(\alpha) > \lambda_{\mathcal{R}'}(\beta)$, for all $\mathcal{R} \neq \mathcal{R}'$.

Determinism is an important notion because it affects certain aspects such as the ability to perform route prediction for traffic engineering [14]; additionally, it affects how easily a protocol can be debugged. Determinism can help an observer ascertain which route BGP will select as the best route, without knowledge of the arrival order of route announcements. It also assures that a participant's choice for best path is immune from any suboptimal advertisements that that participant might hear.

We use two rules from the routing logic to prove properties about safety and stability—the *preference rule* and the *selection rule*. The *preference rule* captures route preferences in this fashion by examining path selection dynamics. Figure 4 shows the preference rule; if two participants have **put** routes to a destination for a third participant C, and C selects the route from A over a specific route from B, this will either manifest itself as a re-advertisement of $\texttt{route}_A$, with C's AS prepended (since C will not readvertise a route that is not its best route), or no advertisement from C to `dest` (if the best route is filtered by export policy). From this **put** statement, we can deduce that C **prefers** $\texttt{route}_A$ **over** $\texttt{route}_B$ for that destination. This allows us to deduce route preferences simply by observing the behavior of the routing protocol without requiring a static analysis of each AS's policies.

The *selection rule* follows directly from route history attribute used to dynamically observe policy cycles [21]. If A prefers $\texttt{route}_1$ over $\texttt{route}_2$, and A **replaces** $\texttt{route}_1$ **with** $\texttt{route}_2$, then A **moves down** from $\texttt{route}_1$ (and vice versa for **moves up to**). A policy cycle is realized if a participant **moves up** to some route and later **moves down** to that same route. This rule is summarized in Figure 5.

2.3.2 Safety and Determinism in BGP

In this section, we will show how the preference and selection rules can concisely express a route history cycle (and thus the potential for unsafety) [22], given knowledge of the routing protocol messages. We also discuss how, under certain configurations, BGP violates time immunity, and, with the use of MEDs, can potentially violate set immunity. When discussing safety, we revisit the classic dispute cycle that arises in the BAD GADGET (Figure 6) configuration and describe the routing logic can express this as a contradiction. In Section 3 (Theorem 3.7), we use the logic, along with the Simple Path Vector Protocol (SPVP) [21], to formally show that safety is not satisfied. In this section, we also examine how MEDs cause BGP to violate determinism (both set immunity and time immunity).

BAD GADGET gives rise to policy oscillations because there is no stable assignment of routes for which some participant would not want to select a different route, given the route assignments to other participants [20]; this appears as a route history cycle [21]. In response, Griffin *et al.* suggest a modification to BGP—the Safe Path Vector Protocol—that causes ASes to ignore routing advertisements that are caused by policy-induced oscillation [21]. Adding a route history attribute to BGP route advertisements is but one mechanism for discovering route history cycles. While the route history attribute might seem like a reasonable solution under some circumstances, it reveals information about the route preferences (and thus the routing policies) of a particular AS. In Section 3, we use the routing logic to examine how the route history can violate information flow policies.

The routing logic can show that, under certain configurations, BGP satisfies neither set immunity nor time immunity. Because the route to some destination at a router may depend on the best route selected by some other router in the AS, route prediction must be done globally and is extremely difficult to reason about, primarily because the MED attribute prevents a simple ordering of routes [14]. This anomaly can appear when the arrival of a third route advertisement results in the AS reversing its preference between the two previously existing route advertisements. This can have the strange effect that the arrival of this third route can result in the displacement of the current best route, even if this new route is not selected as the best route. This is because BGP does not satisfy set immunity.

Consider the configuration in Figure 7. Assume AS 1 always knows about routes α and β to some destination in AS 0. If AS 3 advertises route γ to that destination, X chooses α as its best route—γ is ranked higher than β due to MED, but α is preferred over γ because it is learned via eBGP. If, however, AS 3 does not advertise γ, then X will choose β as its best route, based on the lowest router ID. Thus, the presence of the suboptimal route γ affects whether X selects α or β as its best route. We use the routing logic to show this violation of set immunity formally in Section 3 (Theorem 3.8). A slight modification fixes this problem: most implementations of BGP have an option known as `always-compare-med` that causes the MED attribute to be comparable across all routes (as opposed to just routes from the same AS). In Section 3 (Theorem 3.9), we show that BGP with the `always-compare-med` option satisfies set immunity.

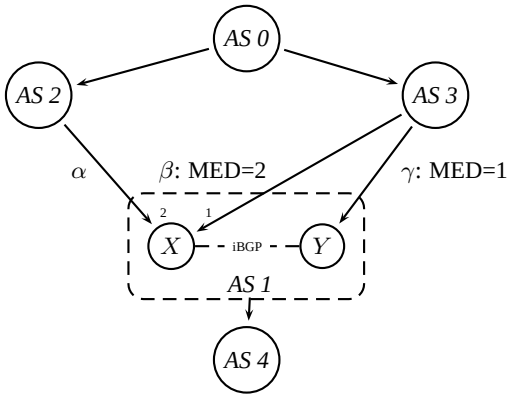


Figure 7: MEDs cause the ordering between pairs of routes to depend on the presence or absence of other routes. Depending on the presence of route γ, router X will prefer either α or β. (Small numbers near router X denote a tiebreaking metric, such as router ID.)

In the interest of space, we omit detailed discussion and proofs relating to time immunity. However, it can be shown that, while under ordinary circumstances BGP does not satisfy time immunity, the `bgp-deterministic-med` option causes BGP to satisfy this property as well. A router configured with this option selects its best route to a destination from the set of all received routes for that destination, rather than selecting it by comparing the current best route with the most recently received route. With `always-compare-med` and `bgp-deterministic-med`, BGP satisfies determinism.

2.4 Information-Flow Control

Wide-area routing requires cooperation between distinct, often competing routing domains that try to keep certain information private (e.g., peering and transit agreements and internal network topology). However, to achieve global reachability, these entities must exchange routing information, which can potentially reveal some of this sensitive information. Knowledge of AS-level topology can lend insight into peering and transit agreements [15, 36]. `set-metric-internal` [8], a Cisco IOS command that assigns MED values for route announcements according to the IGP distance from network ingress to egress, can reveal information about internal routing instability because IGP changes are reflected as changes in BGP routes. Stateless BGP implementations [28] can save router memory by not keeping track of which routes have been advertised to which neighbors; however, this optimization requires that these routers issue withdrawals for prefixes whose announcements were filtered by export policy. Other proposed modifications, such as adding a route history attribute to BGP [21], can leak information about peering and transit relationships.

2.4.1 Routing as an Information Flow Model

An information flow model consists of objects, an information flow policy, processes that cause information to flow, and a partial ordering of security levels. In the case of wide-area routing, objects include all information in the system (i.e., peering agreements, routing advertisements, policies, etc.), as well as the participants themselves. Every object is assigned a security level. The information flow policy is defined in terms of a partial ordering, which is commonly expressed as a lattice [11].

An information flow model for routing protocols should (1) specify the objects in the system, as well as the processes that cause

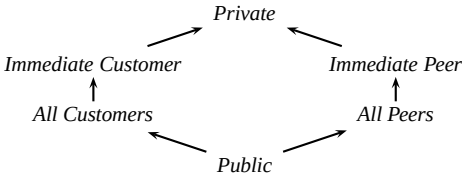


Figure 8: An example information flow lattice.

information to flow between these objects and (2) clearly and concisely express how information flow should be controlled between these parties. The information objects in wide-area routing fall into three categories:

- *Policy.* Policy includes peering and transit agreements, filtering policies and other route preferences.
- *Reachability.* Reachability incorporates any type of information related to events that affect reachability, such as internal network failures.
- *Topology.* Topology consists of internal network topology, as well as inter-AS connectivity.

Routing messages, which contain information such as prefixes, next-hop IP addresses, AS paths, MED values, and session resets, can cause the unexpected transfer of information across security levels. An administrative domain specifies an information flow policy by assigning security levels to each of these pieces of information (e.g., keep peering relationships private), as well as to other participants (e.g., assign AS A to the security level corresponding to "all of my peers"), and specifying the information flow policy itself (e.g., "don't let arbitrary participants discover information intended for this specific peer").

Figure 8 shows a simplified information flow lattice, which specifies example security levels and how information should flow between them. In a multi-level security system, a process can only read a set of objects if the security level of that process dominates the least upper bound of the security levels of these objects. For example, a sequence of routing messages with a least upper bound security level of *public* may flow into any object at the *all customers* level, but not vice versa. Similarly, information that is permitted to flow to all customers can flow into an object with the *immediate customer* designation, but not into an object with an *immediate peer* designation.

The routing logic should express whether a routing protocol satisfies the *noninterference* property. Noninterference states that actions of objects at higher security levels should not be visible to objects at lower security levels [17]. In certain cases, a routing protocol may not be able to satisfy an information flow policy and still satisfy other requirements (e.g., if the routes themselves were specified as private and the information flow policy were satisfied, validity would obviously not hold). We would like to express when policies cannot be satisfied.

2.4.2 Noninterference Rule

In BGP, information flow corresponds to the actual route announcements and withdrawals. For wide-area routing protocol to satisfy an information flow policy, no routing message must cause information at a higher security level to flow to objects at a lower security level; this property is called noninterference. The *noninterference rule* (Figure 9) specifies the conditions under which a participant's route announcement satisfies noninterference.

First, the security level of a routing message must be no higher than the security level of the participant that receives that message.

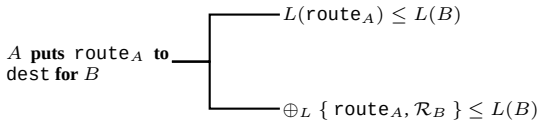


Figure 9: The noninterference rule says that a route advertisement from A to B should not allow B to learn of information at a higher level than B.

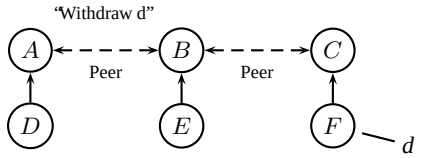


Figure 10: Stateless BGP implementations can result in information flow policy violations.

Second, the least upper bound of the security levels (denoted by $\oplus_L$ in Figure 9) of the route message and all of the other routes learned by that participant ($\mathcal{R}_B$) must be no higher than the security level of B; this requires that information not be passed implicitly (through inference or otherwise). For example, if a particular participant belongs to security level *immediate customer*, no set of routing messages should cause information belonging to security level *private* to flow to the participant at the *immediate customer* level.

2.4.3 Information-Flow Control in BGP

Consider the situation in Figure 10. Suppose that AS B wishes to keep its peering arrangements with A and C *private* (i.e., known only to itself). In this case, a reasonable routing policy is that B should tell A and C about its routes to its own destinations, as well as destinations in E, but should not tell A about routes heard from C, and vice versa [16]. However, if B sends a withdrawal to A for some destination that it never sent an advertisement for (e.g., a destination d in F), then A learns something about B's other peers that it would not otherwise have known: specifically, A learns that B peers with some AS that either contains that destination or has a customer that contains that destination. (Note that since E hears all routes from B, E does not learn any information.)

The noninterference rule expresses this violation: if B specifies that routes learned via C are at security level *immediate customer* (indicating that routes learned via C should only be re-advertised to immediate customers), then B should only be able to **put** a route **for** A (or an empty route, in the case of a withdrawal) if the security level of the relationship $A \leftrightarrow B$ is at least as high as the security level for the route in that **put** statement. Here, B has specified that advertisements heard from C are assigned to the *immediate customer* level. Since A (assigned security level *immediate peer*) is not at a security level at least as high as *immediate customer*, this advertisement violates B's information flow policy. In Section 3, we examine this formally using the routing logic. We note that in general, BGP is prone to information flow policy violations because security levels and access control for route advertisements are implicit in the BGP session itself, rather than explicitly specified.

3. Applying the Routing Logic

In this section, we demonstrate how protocol designers can use the routing logic to analyze BGP4 and various proposed modifications. We show how the logic can be useful for reasoning about the various properties of wide-area routing protocols. In addition

to formally proving properties about previously discovered anomalies and proposed modifications, we propose additional protocol modifications and highlight several previously unknown properties about BGP.

3.1 Validity and Visibility

THEOREM 3.1. *There exists a route reflector configuration for which the existence of a valid route to a destination does not imply the existence of a valid path. That is, there exists a route reflector configuration that causes BGP to violate validity.*

PROOF. Recall the stated precondition that, for a scope i routing protocol to be valid, the routing protocols for all scopes $j > i$ must also be valid. We will show that the scope 0 protocol, BGP, violates the validity property by showing that scope 1 validity is violated.

The proof is by counterexample. Consider the route reflector configuration shown in Figure 3. In this case, we will show that the precondition of valid routing at scope 1 is not satisfied, and thus, scope 0 routing is not satisfied. Consider the route from router A to AS 0; in this case, the scope 1 destination (scope 0 next-hop) is the first hop into AS 1, reached via $RR1$. However, the scope 1 path from A to $RR1$ includes router B, which has a *different* scope 1 destination of the first hop into AS 2. The scope 1 path to its destination, however, includes router A. Thus, scope 1 routing contains a loop, and, as such, scope 0 routing is not valid. ∎

An improper configuration of route reflectors violates the stated precondition that, to decide whether a scope i routing protocol satisfies validity, the routing protocol at scope $i-1$ must satisfy validity for all destination-sets. In this case, there exists an IGP (scope 1) destination-set (i.e., $RR1$, $RR2$) that does not satisfy validity. Therefore, BGP (scope 0) validity cannot be satisfied. It turns out that for an arbitrary configuration of route reflectors, verifying that routing at the intra-AS scope (scope 1) satisfies validity is NP-complete.

THEOREM 3.2. *For an arbitrary configuration of route reflectors and route reflector clients, verifying progress is NP-complete.*

PROOF. If progress at scope 0 satisfied, then progress at scope 1 must also be satisfied. We will show that verifying progress at scope 1 for an arbitrary route reflector configuration is NP-complete by reduction from the directed Hamiltonian cycle problem. That is, assume we have an algorithm A that can verify progress for an arbitrary configuration of route reflectors and route reflector clients. Then, we can use A to solve DIRECTED-HAM-CYCLE.

Define the directed graph $G = (V, E)$, where V is the set of scope 1 participants, and a directed edge E exists between from vertex v_i to v_j if and only if there exists some scope 1 destination for which v_i has v_j as a scope 1 next-hop.

Because iBGP/IGP can actually increase the distance to the destination at any hop in the scope 1 path without having an invalid route, verifying progress at scope 1 involves showing that no scope 1 participant is visited twice along the path from the starting point to a destination. That is, to show that the path from any scope 1 participant to the destination eventually reduces the distance to the destination means that the algorithm must visit the paths for all scope 1 destinations starting from every vertex $v_i \in G$. However, an algorithm that did this would be able to decide if $G \in$ DIRECTED-HAM-CYCLE, since, by the definition of G, the algorithm traverses every path that exists in the graph. ∎

A suggested modification to prevent BGP from causing persistent forwarding loops in the presence of route reflectors is to require that router reflectors be on the shortest IGP path to their clients[12]. We show that this condition results in *valid* routing, given that reachability or policy constraints are satisfied. For the following proofs, we assume that routing protocols below scope 1 satisfy validity, since this analysis is "beyond the scope" of our work.

DEFINITION **RR-IGP-Safe** A route reflector configuration is *RR-IGP-Safe* if the route reflectors within that AS are configured such that route reflectors are on the shortest IGP path to their clients.

THEOREM 3.3. *If the route reflector configuration for an AS along the path to a destination is* RR-IGP-Safe, *then BGP satisfies progress.*

PROOF. Assume that progress is not satisfied. Then it must be the case that either scope 0 progress is not satisfied, or scope 1 progress is not satisfied.

Assume that scope 1 progress is satisfied, but scope 0 progress is not. Then, it must be the case that the scope 0 route traverses the same AS twice; however, sender-side loop detection ensures that this is not the case.

Assume that scope 1 progress is not satisfied. Then there must exist a path to some scope 1 destination (or set of scope 1 destinations) that revisits the same scope 1 participant twice; thus, there must be some path and destination for which the distance does not decrease. Because we assume that IGP is valid, the path to every scope 1 destination must be distance reducing; thus, a scope 1 path that is not valid must be a path to a destination-set containing at least two scope 1 destinations for some scope 0 destination. But this is a contradiction—if the configuration is *RR-IGP-Safe*, then, for every scope 0 destination, there is exactly one scope 1 destination for every scope 0 destination. If this were not the case, then there must be some scope 1 participant which has a route reflector that is not on its shortest path. Thus, scope 1 routing must satisfy progress.

Thus, by contradiction, *RR-IGP-Safe* guarantees that BGP satisfies progress. ∎

DEFINITION **RR-Reflect-All** A route reflector configuration for an AS is *RR-Reflect-All* if all route reflectors for that AS re-advertise all routes to a particular destination (as opposed to simply the best route), and route reflectors re-advertise all routes with each other.

THEOREM 3.4. *If the route reflectors in an AS are configured according to* RR-Reflect-All, *then BGP satisfies progress.*

PROOF. There are two cases. Assume that, for some destination, a scope 0 participant hears multiple routes, where the best route is decided before the IGP tiebreak. In this case, all routers will come to select a single best route.

In the event that this is not the case, each scope 1 participant will select its best route based on the shortest IGP path to the level 1 destination. By definition of *RR-Reflect-All*, for every route, there is some reflector that is on that shortest path that re-advertised that route. Thus, the problem reduces to the *RR-IGP-Safe* configuration, and BGP satisfies progress in this case as well. ∎

THEOREM 3.5. *If an AS uses full mesh iBGP, then BGP satisfies progress.*

PROOF. Full mesh iBGP is equivalent *RR-Reflect-All*, because, in both cases, every BGP router eventually learns of all routes advertised to any other BGP speaking router. Thus, if an AS uses full mesh iBGP, then BGP satisfies progress. ∎

Step	*Paths*	*Rules and Deductions*
—		0:(0) → 1,2,3,4 *[Premise (P)]*
0	(1 0) (2 0) (3 4 2 0) (4 2 0)	1:(1 0) → 2,3 2:(2 0) → 1,4 3:(3 4 2 0) → 1 4:(4 2 0) → 3
1	(1 0) **(2 1 0)** (3 4 2 0) (4 2 0)	2:(2 0) ↛ 1 2:(2 1 0) [2 0] → 4 λ_2(2 1 0) > λ_2(2 0) *[Preference,P,0]* 2 ↑ (2 1 0) *[Selection,1]*
2	(1 0) (2 1 0) (3 4 2 0) ϵ	4:(4 2 0) ↛ 3 $\lambda_4(\epsilon)$ > λ_4(4 2 1 0) *[Preference, 1]*
3	(1 0) (2 1 0) **(3 0)** ϵ	3:(3 4 2 0) ↛ 1 3:(3 0) [3 4 2 0] → 4 λ_3(3 0) > λ_2(3 1 0) *[Preference,P,0]* 3 ↓ (3 4 2 0) *[Selection,2]*
4	(1 0) (2 1 0) (3 0) **(4 3 0)**	4:(4 3 0) → 2 λ_4(4 3 0) > λ_4(4 2 1 0) *[Preference,1,3]*
5	**(1 3 0)** (2 1 0) (3 0) (4 3 0)	1:(1 0) ↛ 3 2:(1 3 0) [1 0] → 2 λ_1(1 3 0) > λ_1(1 0) *[Preference,P,0]* 1 ↑ (1 3 0) *[Selection,5]*
6	(1 3 0) **(2 0)** (3 0) (4 3 0)	2:(2 1 0) ↛ 4 2:(2 0) [2 1 0] → 1 λ_2(2 0) > λ_1(2 1 3 0) *[Preference,P,5]* 2 ↓ (2 1 0) *[Selection,6]* **Dispute Cycle [1,6]**

Table 1: The dispute cycle in BAD GADGET**, expressed using the routing logic.**[8]

THEOREM 3.6. *For any route reflector configuration that is not a full mesh, the existence of a path to a destination does not imply the existence of a route to a destination. That is, visibility can be violated.*

PROOF. Consider an AS where a route reflector drops a BGP session to some route reflector client A; in this case, the route reflector will withdraw its route to d from A. In general, it is the case that A has a valid path to B, a client of a different route reflector that has not withdrawn its route to destination d. Thus, even though a valid route exists through B, A has no route to d. ∎

The result that visibility cannot be guaranteed is the consequence of using route reflectors for iBGP, rather than a full mesh. If, alternatively, an AS uses a full mesh iBGP configuration, then the existence of a valid path to the destination implies the existence of a route. We omit this proof in the interest of space.

3.2 Safety and Determinism

In this section, we prove BGP is not safe using the routing logic; this revisits previous work [20]. We also use the routing logic to formally express how MED causes BGP to violate determinism, and how the `always-compare-med` option can guarantee set immunity. It turns out that `always-compare-med` guarantees time immunity as well, but we omit this proof for lack of space.

THEOREM 3.7. *BGP is not safe.*

PROOF. We show that there exists some configuration in BGP that results in a route history cycle. Table 1 shows the sequence of path assignments (from [21]) for this configuration, as well as the rules used to deduce the resulting contradiction. For space considerations, we express the rules in shorthand. In step 6, the routing logic has produced a contradiction: that AS 2 **moves up** to (2 1 0) and that AS 2 **moves down** to (2 1 0); this is a *route history cycle*, exactly the condition that signifies a dispute cycle. ∎

[8]The destination is implicit in these rules for brevity. $A : r \rightarrow B \equiv A$ **puts** r **for** B; $A : r \not\rightarrow B \equiv A$ **removes** r **for** B; $A \downarrow r \equiv A$ **moves down from** r; $A \uparrow r \equiv A$ **moves up to** r; $\lambda_A(r_1) > \lambda_A(r_2) \equiv A$ **prefers** r_1 **over** r_2.

Step	*Messages*	*Rules and Deductions*
—	—	0:(0) → 2,3
0	α, β	2: $\alpha \rightarrow 1$ 3: $\beta \rightarrow 1$ 1: $\beta \rightarrow 4$ $\lambda_1(1\ \beta) > \lambda_1(1\ \alpha)$ *[Preference,0]*
1	α, β, γ	3:$\gamma \rightarrow 1$ 1: (β) [α] → 4 $\lambda_1(1\ \alpha) > \lambda_1(1\ \beta)$ *[Preference,0,1]* **Contradiction**

Table 2: The preference rules expresses how MED causes BGP to violate the rule of independent ranking.

Griffin *et al.*'s Safe Path Vector Protocol [21] would correspond to step 6 being replaced with the path assignment ((1 3 0 ϵ (3 0) (4 3 0)), in which case the rules of the routing logic would never reveal a contradiction.[9] Additionally, improperly configured route reflectors can result in divergence [32]; this class of MED-induced routing anomalies can be expressed in terms of policy cycles in the General Stable Paths Problem [22]. In the same way that Griffin *et al.* used dispute cycles to show the potential for MED-induced routing anomalies (MIRA) [22], we can realize these dispute cycles with the preference and selection rules by assigning a participant to each proxy in the General Stable Paths Problem.

The routing logic expresses dispute cycles and violations of set and time immunity. In general, constructing these proofs depends on having access to the necessary routing messages; in BGP, however, no single AS sees all of the routing messages needed to construct this proof. While the route history attribute can solve this problem, we examine how the use of this attribute can violate information flow policies in Section 3.3.

THEOREM 3.8. *BGP with the multi-exit discriminator (MED) attribute does not satisfy set immunity.*

PROOF. The proof is by counterexample. Consider the situation in Figure 7. If router X only knows routes α and β to some destination, then the best route is $\sigma_X(\alpha, \beta) = \beta$, as decided by the tiebreak (in principle this could be anything after the MED step in the decision process, such as IGP, router ID, etc.). However, $\sigma_X(\alpha, \beta, \gamma) = \alpha$, since eBGP-learned routes are always preferred over iBGP-learned routes. Table 2 shows the deductions associated from a sequence of route advertisements α, β, and γ, and the resulting nonlinearity in X's preferences. That is $\lambda_{\{\alpha,\beta\}}(1\ \beta) > \lambda_{\{\alpha,\beta\}}(1\ \alpha) \not\Rightarrow \lambda_{\{\alpha,\beta,\gamma\}}(1\ \beta) > \lambda_{\{\alpha,\beta,\gamma\}}(1\ \alpha)$. Thus, BGP does not satisfy set immunity. ∎

As shown, the routing logic will cause the preference rules to reach contradictory assertions when immunity is violated.

THEOREM 3.9. *BGP with* `always-compare-med` *satisfies set immunity.*

PROOF. The proof is by contradiction. Assume there exist two sets of route advertisements, $\mathcal{R} \neq \mathcal{R}'$, and two routes α and β, such that $\lambda_{\mathcal{R}}(\alpha) > \lambda_{\mathcal{R}}(\beta) \not\Rightarrow \lambda_{\mathcal{R}'}(\alpha) > \lambda_{\mathcal{R}'}(\beta)$.

The set $\mathcal{R}$ does not affect the ranking function itself; that is, the partial ordering $\lambda_{\mathcal{R}'}(\alpha) > \lambda_{\mathcal{R}'}(\beta)$ is still valid. Thus, in order for the rule $\lambda_{\mathcal{R}}(\beta) > \lambda_{\mathcal{R}}(\alpha)$ to be deduced, there must be some γ in $\mathcal{R}' - \mathcal{R}$ such that $\lambda_{\mathcal{R}'}(\gamma) > \lambda_{\mathcal{R}'}(\alpha)$ and $\lambda_{\mathcal{R}'}(\beta) > \lambda_{\mathcal{R}'}(\gamma)$, i.e., the partial ordering must *not* not satisfy transitivity. Because all other decision criteria for the scope 0 selection process are transitive except MED, this implies that the MED part of the BGP decision process must not be transitive. However, `always-`

[9]We omit the formal proof that this proposal suppresses route history cycles because it is described extensively in previous work [21].

`compare-med` makes this step of the decision process transitive; thus, we have a contradiction. ∎

3.3 Information Flow Control

In this section, we use the routing logic to formally describe how certain aspects of BGP (and proposed modifications to BGP) can result in unintended information leakage.

THEOREM 3.10. *A stateless BGP implementation can violate standard*[10] *information flow policy.*

PROOF. The proof is by counterexample. Consider the example in Figure 10 and the information flow lattice in Figure 8. Assume that AS B assigns security level *immediate peer* to AS A and AS C, *immediate customer* to AS E, and *all customers* to its BGP sessions with A and C. Upon a withdrawal from some destination d in AS F, which has AS C as its only upstream provider, a stateless BGP implementation at AS B will readvertise the withdrawal for d to all neighbors, including its peer A. But the security level of the BGP session with C is *all customers*, which is not at least as low as the security level *immediate peer*. Thus, information flow policy is violated. ∎

THEOREM 3.11. *The BGP route history attribute violates standard information flow policy.*

PROOF. There are many examples of policy violation; we describe one. Consider step 4 of Table 1. The selection rule states that AS 3 **moves down** to (3 4 2 0) **from** (3 0). Any participant that knows that (3 0) has not been deleted knows that $\lambda_3(3\ 0) < \lambda_3(3\ 4\ 2\ 0)$. Thus, the route history attribute, which attaches this information to the route advertisement, would violate a standard policy of keeping local preference values private. ∎

There are many other examples where BGP violates information flow policies. For example, using `set-metric-internal` exposes interior instability and IGP path changes via BGP updates with changed MED announcements.

4. Potential Applications

The previous section showed that the routing logic can be useful for reasoning about routing protocol design and configuration in the design phase. In this section, we explore potential practical applications for the routing logic. We believe that the logic will prove useful to network operators by enabling static analysis of existing network configuration and providing a framework for the design of high-level policy specification. Additionally, by designing routing protocols that adhere more closely to the routing logic, designers of new routing protocols and routing protocol modifications can reason more easily about various properties. We highlight a few practical uses of the routing logic with some illustrative examples; our ongoing work explores these possibilities in further detail and in more complicated scenarios.

4.1 Configuration Analysis

We envision the routing logic being applied to a tool that verifies properties of legacy router configurations within an autonomous system. We are currently developing such a tool. As a simple example, we show how the routing logic can be used to verify that a router configuration satisfies a specified information flow policy. Best common practice states that peers should receive routes learned from customers but not from other peers [16]. Figure 11 specifies the corresponding information flow lattice. The current version of our analysis tool verifies that the configuration for all routers in an AS conform to this policy.

[10] Most export policies should advertise routes heard from peers to customers only and keep local preference values private.

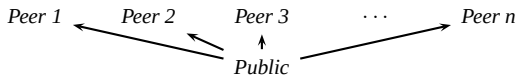


Figure 11: An information flow lattice that specifies that routes learned from one peer should not be readvertised to another peer.

To verify that a particular configuration conforms to this policy, the analysis tool first assigns a security level to each BGP session at each router in the AS. Then, for each principal in the information flow lattice (i.e., for each peer), the tool performs the following steps: (1) determine export policies for all BGP sessions to that principal, (2) for each BGP session, determine whether the routes learned at that session will be exported to the peer in question, and (3) verify that every route exported to this peer satisfies the noninterference rule.

4.2 Configuration Synthesis

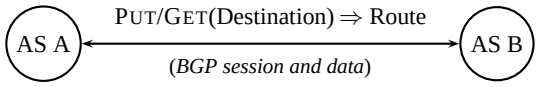
Today, network operators configure routers with low-level configuration languages; these languages unnecessarily expose complexity and are thus subject to frequent misconfiguration [29]. Operators should be able to configure networks using a high-level specification that abstracts details. Such a tool could synthesize low-level configuration by translating a high-level policy specification. Operators could benefit from assurances that the low-level configuration is a translation of the corresponding high-level specification that (1) preserves semantics and (2) satisfies various properties, such as those which can be verified using the routing logic. The routing logic can provide a useful framework for designing policy language translation that satisfies the latter design goal.

For example, configuration of peering export policy could be improved using an automated translation of high-level specification. Presently, these information flow policies are configured using a separate set of `access-list` statements and import and export policies for each peer. Import policy typically "tags" each route with a community that indicates that the route was learned via a peering session, and export policy uses the appropriate access list to ensure that no route with such a tag is advertised via an eBGP session with a different peer. Configuration synthesis that automatically generated these access lists and import and export policies could not only reduce the possibility of accidental misconfiguration, but also the likelihood of accidental route leakage due to non-atomic updates to router configuration [31]. Many other possibilities exist for configuration synthesis, including route reflector configuration, outbound traffic engineering, etc.

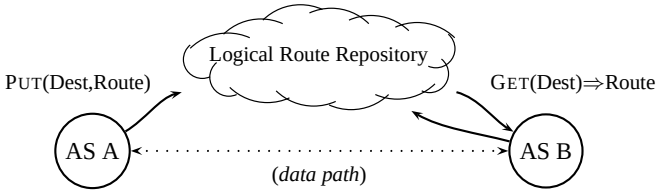
4.3 Protocol Design

Wide-area routing protocols should implement a set of protocol abstractions that, when related to the routing logic, can be used to determine if a routing protocol's behavior conforms to a particular property. BGP commonly violates many wide-area routing properties because its operation is not governed by a set of rules that guarantee that each operation satisfies these properties. In contrast, a protocol that implements abstractions that adhere more closely to the logic is much more amenable to provably satisfying properties. In this section, we briefly present examples of these abstractions. We are planning to design and implement such a protocol.

In wide-area routing, each AS informs other ASes of the routes for which it is willing to carry traffic (*route dissemination*). When



(a) BGP couples route dissemination and retrieval.



(b) These can be thought of as two separate operations.

Figure 12: Wide-area routing consists of route dissemination and retrieval. BGP's message passing architecture couples these two operations, but they can be separated into distinct operations on a logical data repository.

a routing domain sends traffic to a particular destination, it chooses the egress on which to send this traffic. This requires up-to-date, valid information about which of its neighbors are willing to forward traffic to that destination (*route retrieval*).

In BGP, these two operations are coupled. As the best route to a destination changes, an AS propagates these changes directly to its neighbors, who are always listening for changes. Alternatively, these two operations could be decoupled into a route dissemination operation (i.e., PUT) and a route retrieval operation (i.e., GET), as shown in Figure 12. In practice, the GET operation may be implemented in a triggered fashion, where updates to a route object via a PUT are propagated automatically. Route dissemination must also implement DELETE, which results in the invalidation of a previously advertised route; and REPLACE, an atomic two-command DELETE/PUT sequence for a route to the same destination.

We can map these abstractions directly to assertions in the routing logic to determine whether, and under what conditions, a routing protocol will satisfy a certain property. For example, the PUT operation corresponds directly to the **put** statement in the routing logic. If a routing protocol operation can be specified in terms of the PUT abstraction, we can say that a **put** assertion was made whenever the protocol executes that operation. Similarly, DELETE corresponds to an assertion that a previous **put** assertion is no longer valid, and REPLACE corresponds to the **replace** statement in the routing logic. Other assertions (e.g., **prefers**, **moves up/down**, etc.) are deduced using using the preference and selection rules in conjunction with **put** and **replace** statements. We can apply the routing logic to determine whether or not that operation (or some sequence of operations) violates validity by ascertaining whether the associated reachability, policy conformance, and progress rules are consistent with an assertion or set of assertions.

Although we have focused on applying the logic to BGP, we believe that a routing protocol designed with these abstractions in mind will facilitate assurances regarding whether a protocol satisfies a set of properties. For example, an improved routing protocol could enforce invariants, such as requiring that the **put** assertion corresponding to a PUT be true before permitting that operation, or ensuring that the GET for an object at a particular security level is only permitted from participants at that level or higher. Exploring alternate routing architectures that makes these types of assurances feasible is part of our ongoing work.

5. Related Work

There are several areas of research that relate to our work. Our work was inspired by the use of BAN logic for authentication protocol analysis, which lays the formal groundwork for proving properties about authentication protocols [6], and the Taos operating system, which applies BAN logic to a real-world system to analyze various properties [40]. We envision our logic developing as an analogous framework that can be used to analyze existing protocols and develop new ones. We incorporate previous work on the information flow lattice model [11] and were inspired by its application to programming languages [33], which inspired the ideas that information flow can be applied to high level policy specifications for BGP. Several algorithms have been proposed to use BGP to garner information about sub-AS topology [3], as well as inter-AS relationships and the AS hierarchy [15, 36], which provided a motivation for information flow control. Work in firewall configuration has proposed a high-level specification language that uses a high-level abstraction based on formal logic [4].

Our work builds heavily on the many specific BGP anomalies noted by previous work in routing instability [38], delayed convergence [24, 27], route reflector configuration [12, 32], route flap dampening [30], accidental misconfiguration [29], and the difficulties in route prediction for traffic engineering [14]. The work of Griffin *et al.* on the Stable Paths Problem and General Stable Paths Problem is an integral part of our routing logic rules for safety [21, 22, 23, 25].

Additionally, previous work has proposed extensions or alternatives to contemporary wide-area routing protocols that improve certain properties, such as scalability [7], the ability to withstand certain types of failures [34], and defense against malicious or incorrect route advertisements [26]. Architectures involving route servers out-of-band from BGP itself have been proposed as solutions for policy oscillation and route hijacking [1, 18, 19].

6. Conclusion

In this paper, we have presented a *routing logic*: a set of rules for proving properties about certain aspects of routing protocols. The routing logic has enabled us to understand and formally describe how certain fundamental properties of BGP can result in inadvertent violations of various properties. The logic provides a framework to evaluate future proposed modifications to BGP and can be extended to incorporate other requirements, such as traffic engineering.

The routing logic opens up many avenues for future work in wide-area routing protocols. We plan to perform a more thorough analysis of the circumstances under which BGP violates various properties, particularly information flow control, which has been understudied to date. We are exploring how the routing logic can be applied to help operators evaluate existing routing protocol configurations and motivate the design of a high-level policy specification language, both of which will facilitate reasoning about BGP configuration. Finally, we believe that it is possible to design a routing protocol that provides all necessary functionality but is easier to reason about and provably behaves according to certain properties. To explore this possibility, we intend to design and implement a wide-area routing protocol whose operation adheres more closely to the assertions and rules of the routing logic.

Acknowledgements

We thank David Andersen, Magdalena Balazinska, Jaeyeon Jung, Steven Richman, Rodrigo Rodrigues, Michael Walfish, and Xiaowei Yang for their comments on a previous draft of this paper. We also thank the anonymous reviewers for their suggestions.

7. References

[1] AGARWAL, S., CHUAH, C., AND KATZ, R. H. OPCA: Robust interdomain policy routing and traffic control. In *IEEE Openarch* (New York, NY, April 2003).

[2] ANDERSEN, D. G., BALAKRISHNAN, H., KAASHOEK, M. F., AND MORRIS, R. Resilient Overlay Networks. In *Proc. 18th ACM SOSP* (Banff, Canada, Oct. 2001), pp. 131–145.

[3] ANDERSEN, D. G., FEAMSTER, N., BAUER, S., AND BALAKRISHNAN, H. Topology Inference from BGP Routing Dynamics. In *Proc. Internet Measurement Workshop* (Marseille, France, 2002).

[4] BARTAL, Y., MAYER, A., NISSIM, K., AND WOOL, A. Firmato: A novel firewall management toolkit. In *IEEE Symposium on Security and Privacy* (Oakland, CA, May 1999), pp. 17–31.

[5] BEARD, D., ET AL. *Known Threats to Routing Protocols*. Internet Engineering Task Force, October 2002. `http://www.ietf.org/internet-drafts/draft-beard-rpsec-routing-threats-00.txt`.

[6] BURROWS, M., ABADI, M., AND NEEDHAM, R. A logic of authentication. *ACM Transactions on Computer Systems 8*, 1 (February 1990), 18–36.

[7] CANSTINEYRA, I., CHIAPPA, N., AND STEENSTRUP, M. *The Nimrod Routing Architecture*. Internet Engineering Task Force, August 1996. RFC 1992.

[8] Cisco BGP commands. `http://www.cisco.com/univercd/cc/td/doc/product/software/ios113ed/cs/csprtn1/csbgp.htm`.

[9] Cisco BGP Best Path Selection Algorithm. `http://www.cisco.com/warp/public/459/25.shtml`.

[10] How BGP Routers Use the Multi-Exit Discriminator for Best Path Selection. `http://www.cisco.com/warp/public/459/37.html`.

[11] DENNING, D. E. A lattice model of secure information flow. *Communications of the ACM 19*, 5 (May 1976), 236–243.

[12] DUBE, R. A comparison of scaling techniques for BGP. *ACM Computer Communications Review 29*, 3 (July 1999), 44–46.

[13] FARROW, R. Routing instability on the Internet. *Network Magazine* (March 4, 2002). `http://www.networkmagazine.com/article/NMG20020304S0007/2`.

[14] FEAMSTER, N., AND REXFORD, J. Network-wide BGP route prediction for traffic engineering. In *Proc. ITCOM* (Boston, MA, August 2002).

[15] GAO, L. On inferring automonous system relationships in the Internet. *IEEE/ACM Transactions on Networking 9*, 6 (December 2001), 733–745.

[16] GAO, L., AND REXFORD, J. Stable Internet routing without global coordination. *IEEE/ACM Transactions on Networking* (December 2001), 681–692.

[17] GOGUEN, J., AND MESEGUER, J. Unwinding and inference control. In *Proc. IEEE Symposium on Security and Privacy* (1984).

[18] GOODELL, G., AIELLO, W., GRIFFIN, T., IOANNIDIS, J., MCDANIEL, P., AND RUBIN, A. Working around BGP: An incremental approach to improving security and accuracy in interdomain routing. In *Proc. NDSS* (San Diego, CA, February 2003).

[19] GOVINDAN, R., ALAETTINOGLU, C., AND KANNAN VARADHAN, D. E. Route servers for inter-domain routing. *Networks and ISDN Systems 30* (1998), 1157–1174.

[20] GRIFFIN, T., AND WILFONG, G. An analysis of BGP convergence properties. In *Proc. ACM SIGCOMM* (Cambridge, MA, August 1999).

[21] GRIFFIN, T., AND WILFONG, G. A safe path vector protocol. In *Proc. INFOCOMM* (March 2000).

[22] GRIFFIN, T., AND WILFONG, G. Analysis of the MED oscillation problem in BGP. In *Proc. ICNP* (Paris, France, November 2002).

[23] GRIFFIN, T., AND WILFONG, G. On the correctness of IBGP configuration. In *Proc. ACM SIGCOMM* (Pittsburgh, PA, August 2002).

[24] GRIFFIN, T. G., AND PREMORE, B. J. An experimental analysis of BGP convergence time. In *Proceedings of the 9th International Conference on Network Protocols (ICNP 2001)* (Riverside, CA, November 2001).

[25] GRIFFIN, T. G., SHEPHERD, F. B., , AND WILFONG, G. The stable paths problem and interdomain routing. *IEEE Transactions on Networking 10*, 1 (2002), 232–243.

[26] KENT, S., LYNN, C., MIKKELSON, J., AND SEO, K. Secure border gateway protocol (S-BGP) - real world performance and deployment issues. In *Proc. NDSS 2000* (2000).

[27] LABOVITZ, C., AHUJA, A., BOSE, A., AND JAHANIAN, F. Delayed Internet Routing Convergence. *IEEE/ACM Transactions on Networking 9*, 3 (June 2001), 293–306.

[28] LABOVITZ, C., MALAN, G. R., AND JAHANIAN, F. Origins of Internet routing instability. In *Proc. Infocom* (New York, NY, March 1999), pp. 218–226.

[29] MAHAJAN, R., WETHERALL, D., AND ANDERSON, T. Understanding BGP misconfiguration. In *Proc. ACM SIGCOMM* (Aug. 2002), pp. 3–17.

[30] MAO, Z. M., GOVINDAN, R., VARGHESE, G., AND KATZ, R. Route Flap Damping Exacerbates Internet Routing Convergence. In *Prof. ACM SIGCOMM 2002* (Pittsburgh, PA, August 2002).

[31] MAUCH, J. Odd UUNet BGP announcements for interior netblocks. `http://www.merit.edu/mail.archives/nanog/2002-04/msg00653.html`, April 2002.

[32] MCPHERSON, D., GILL, V., WALTON, D., AND RETANA, A. *Border Gateway Protocol (BGP) Persistent Route Oscillation Condition*. Internet Engineering Task Force, August 2002. RFC 3345.

[33] MYERS, A. C., AND LISKOV, B. A decentralized model for information flow control. In *Proceedings of the 16th ACM Symposium on Operating Systems Principles (SOSP)* (Saint-Malo, France, October 1997), pp. 129–142.

[34] PERLMAN, R. *Network Layer Protocols with Byzantine Robustness*. PhD thesis, Massachusetts Institute of Technology, October 1988. MIT-LCS-TR-429. `http://www.lcs.mit.edu/publications/specpub.php?id=997`.

[35] REKHTER, Y., AND LI, T. *A Border Gateway Protocol 4 (BGP-4)*. Internet Engineering Task Force, 1995. RFC 1771.

[36] SUBRAMANIAN, L., AGARWAL, S., REXFORD, J., AND KATZ, R. H. Characterizing the Internet hierarchy from multiple vantage points. In *Proc. IEEE INFOCOM* (New York, NY, June 2002).

[37] TODD, J. AS number inconsistencies. `http://www.merit.edu/mail.archives/nanog/2002-07/msg00259.html`, July 2002.

[38] VARADHAN, K., GOVINDAN, R., AND ESTRIN, D. Persistent route oscillations in inter-domain routing. *Computer Networks 32*, 1 (2000), 1–16.

[39] WANG, L., ET AL. Observation and analysis of BGP behavior under stress. In *Proc. ACM SIGCOMM Internet Measurement Workshop* (Marseille, France, November 2002).

[40] WOBBER, E., ABADI, M., BURROWS, M., AND LAMPSON, B. Authentication in the Taos operating system. In *Proceedings of the 14th ACM Symposium on Operating System Principles (SOSP)* (Asheville, NC, December 1993), pp. 256–269.