BGP routing policies in ISP networks

Matthew Caesar
UC Berkeley

Jennifer Rexford
Princeton University

Abstract

The Internet has quickly evolved into a vast global network owned and operated by thousands of different administrative entities. During this time, it became apparent that vanilla shortest-path routing would be insufficient to handle the myriad operational, economic, and political factors involved in routing. ISPs began to modify routing configurations to support *routing policies*, i.e. goals held by the router's owner that controlled which routes were chosen and which routes were propagated to neighbors. BGP, originally a simple path-vector protocol, was incrementally modified over time with a number of mechanisms to support policies, adding substantially to the complexity. Much of the mystery in BGP comes not only from the protocol complexity but also from a lack of understanding of the underlying policies and the problems ISPs face which they address. In this paper we shed light on goals operators have and their resulting routing policies, why BGP evolved the way it did, and how common policies are implemented using BGP. We also discuss recent and current work in the field that aims to address problems that arise in applying and supporting routing policies.

1 Introduction

In the early days of the Internet, the problem of how to route packets to their final destination was much simpler than it is today. At the time, the requirements of the Internet's routing protocol were fairly simple, as the Internet was small by today's standards, operated by a single administrative entity (NSFNET), and shortest-path routing was typically used. Over time, as the Internet became more heavily commercialized and privatized, Internet Service Providers (ISPs) began to have vested interests in controlling the way traffic flowed for economic and political reasons. The Border Gateway Protocol (BGP) was born out of the need for ISPs to control route selection (where to forward packets) and propagation (who to export routes to).

When BGP was first introduced, it was a fairly simple path-vector protocol. Over time, many incremental modifications to allow ISPs to control routing were proposed and added to BGP. The end result was a protocol weighted down with a huge number of mechanisms that can overlap and conflict in various unpredictable ways. These modifications can be highly mysterious since many of them, including the decision process used to select routes, are not part of the protocol specification [1]. Moreover, their complexity gives rise to several key problems, including unforeseen security vulnerabilities, widespread misconfiguration, and conflicts between policies at different ISPs.

Addressing BGP's problems is difficult, as changing certain aspects of BGP (for example changing the contents of update messages or the way they are propagated) must be coordinated and simultaneously implemented in other ISPs to support the new design. Hence most modifications to the protocol have been made to the *decision process* BGP uses to choose routes. The result is a protocol where most of the complexity is in the decision process and the policies used to influence decisions, while the rest of the protocol remained fairly simple over time. Therefore, in order to understand BGP it is necessary to understand this decision process and the policies of ISPs that gave rise to its design. Understanding policies is also key to solving BGP's problems, understanding measurement data from BGP, or determining what features to support when developing a new version of BGP.

The range of policies used by operators constitutes a huge space and hence it is impossible to list them all here. Instead, we try to list common goals of network operators and the knobs of BGP that can be used to express policies. In particular, we attempt to isolate certain *design patterns* commonly used by ISPs, the motivations behind them, and how they are implemented in an ISP's network using BGP's mechanisms. We taxonomize policies into four general categories: *business relationship* policy (Section 3) arising from economic or political relationships an ISP has with its neighbor, *traffic engineering* policy (Section 4) arising from the need to control traffic flow within an ISP and across peering links to avoid congestion and provide good service quality, policies for *scalability* (Section 5) to reduce control traffic and avoid overloading routers, and *security*-related policies (Section 6) that are often used to protect an ISP against malicious or accidental attacks. We also discuss several avenues of research currently in progress related to BGP policies (Section 7). We start by giving an overview of BGP routing in the next section.

2 BGP routing in a single AS

The Internet consists of thousands of *Autonomous Systems* (ASes)—networks that are each owned and operated by a single institution. BGP is the routing protocol used to exchange reachability information across ASes. Usually each ISP operates one AS, though some ISPs may operate multiple ASes for business reasons (e.g. to provide more autonomy to administrators of an ISP's backbones in the United States and Europe) or historical reasons (e.g. a recent merger of two ISPs). Non-ISP businesses (enterprises) may also operate their own ASes so as to gain the additional routing flexibility that arises from participating in the BGP protocol.

Compared to enterprise networks, ISPs usually have more complex policies arising from the fact that they often have several downstream customers, connect to certain customers in multiple geographic locations, have complex traffic engineering goals, and run BGP on internal routers (rather than just border routers as enterprises often do). Although some of the observations we make apply to enterprise networks, our core focus in this paper is on ISP networks. In this section, we describe BGP from the standpoint of a single AS, describing first the protocol that transmits routes from one AS to another, then the decision process used to choose routes, and finally the mechanisms used at routers to implement policy.

2.1 Exchanging routing state

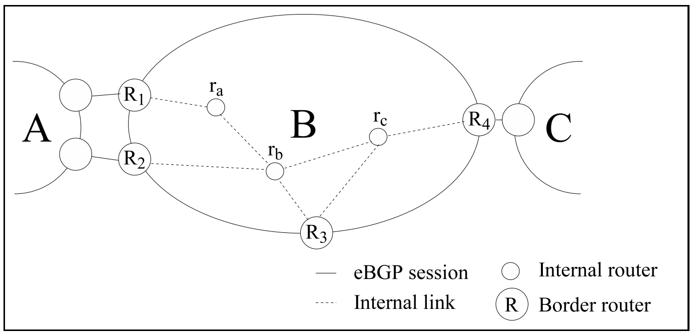

Figure 1: Example topology with three ISPs A, B, and C.

Figure 1 shows a simple BGP network. BGP sessions are established between *border routers* that reside at the edges of an AS and border routers in neighboring ASes. These sessions are used to exchange routes between neighboring ASes. Border routers then distribute routes learned on these sessions to non-border (internal) routers as well as other border routers in the same AS using internal-BGP (iBGP). In addition, the routers in an AS usually run an Interior Gateway Protocol (IGP) to learn the internal network topology and compute paths from one router to another. Each router combines the BGP and IGP information to construct a forwarding table that maps each destination prefix to one or more outgoing links along shortest paths through the network to the chosen border router.

BGP is a relatively simple protocol with a few salient features. First, BGP is an *incremental* protocol, where after a complete routing table is exchanged between neighbors, only changes to that information are exchanged. These changes may be new route advertisements, route withdrawals, or changes to route attributes. Second, BGP is a *path-vector* protocol where advertisements contain a list of ASes used to reach the destination. Third, routes are advertised at the *prefix* level, so an AS would send a separate update for each of its reachable prefixes. Fourth, BGP update messages may contain several fields, including a list of prefixes being advertised, a list of prefixes being withdrawn, and a list of route *attributes* that describe various characteristics of each advertised route. An ISP implements its policies by modifying route attributes and changing the way routers react to advertisements with certain route attributes, as discussed below.

2.2 Selecting a route at a router

A BGP router in an ISP may have several alternate routes to reach a particular destination. In the absence of policy, the router would choose the route with the minimum pathlength, with some arbitrary way to break ties between routes with the same pathlength. However, in order to give operators greater control over route selection, several additional attributes were added to advertisements, allowing a router to alter its decisions based on the values of these attributes. The end result is the BGP *decision process*, consisting of an ordered list of attributes across which routes are compared, as shown in Table 1. The router goes down the list, comparing each attribute in the list across the two routes. If the routes have different values for the attribute, the router chooses the one that has the more desirable attribute, otherwise it moves on to compare the next attribute in the list. The route that is chosen is used by the router to forward packets. The ordering of attributes allows the operator to influence various stages of the decision process. For example, the Local Preference (LocalPref) is the first step in the decision process. By changing LocalPref, an operator can force a route with a longer AS path to be chosen over a shorter one. As another example, the Multi-Exit Discriminator (MED) is typically used by two ASes connected by multiple links to indicate which peering link should be used to reach the AS advertising the attribute. MED was placed lower in the decision process as this allows an ISP to override these suggestions, e.g. by setting LocalPref. Using a strict ordering of attributes in the decision process simplifies policy expression and makes it easier to predict the outcome of making configuration changes. While some vendors allow operators to disable certain steps in the decision process, they typically do not permit the operators to put the steps in a different order. Hence some policies that violate this ordering (e.g. ignore AS path length, or first choose lowest MED then highest LocalPref) may require various hacks which can complicate router configuration and lead to unforeseen side effects.

Table 1: Steps in the BGP decision process.

Step	**Attribute**	**Controlled by local or neighbor AS?**
1.	Highest LocalPref	local
2.	Lowest AS path length	neighbor
3.	Lowest origin type	neither
4.	Lowest MED	neighbor
5.	eBGP-learned over iBGP-learned	neither
6.	Lowest IGP cost to border router	local
7.	Lowest router ID (to break ties)	neither

There are different locations where a route attribute can be set by policy: (a) *Locally*, for example LocalPref is an integer value set at and propagated throughout the local AS and filtered before sending to neighboring ISPs. (b) *Neighbor*, for example the MED attribute is typically used by two ASes connected by multiple links to indicate which peering link should be used to reach the AS advertising the MED attribute, and is not used to compare routes through two different next-hop ASes. (c) *Neither:* some attributes, for example whether the route was learned through an external BGP (eBGP) neighbor or from an internal router speaking BGP (iBGP), are set by the protocol and cannot be changed.

The collective results of the decision process across routers is to produce a set of *equally good* border routers for each prefix, where each router in the set is equivalent according to the first four steps of the decision process that compare BGP attributes. Each internal router then chooses the router in that set that is closest according to the Interior Gateway Protocol (IGP) path cost to reach that border router. For example in Figure 1, suppose prefix 6.0.1.0/24 is reachable to B via both A and C, but B's LocalPref is set higher for routes through A. The set of equally good border routers would then contain R_1 and R_2, and each router in B would select the route that was closest exit point (lowest IGP cost): r_a and R_1 would choose the route through R_1, and all other routers would choose the route through R_2.

There are three steps a router uses to process route advertisements. First *import policy* is applied to determine which routes should be filtered and hence eliminated from consideration, and may append or modify attributes. Next, the router applies the *decision process* to select the most desirable route. Finally, an *export policy* is applied which determines which neighbors the chosen route will be exported to. An ISP may implement its policy by controlling any of these three steps, i.e., by modifying import policy to filter routes it doesn't want to use, modifying route attributes to prefer some routes over others, or by modifying export policy to avoid providing routes for certain neighbors to use. In addition, an ISP can modify attributes of routes it advertises, which can influence how its neighbors perform route selection.

2.3 Configuring local policies

There are three classes of "knobs" that can be used to control import and export policies:

1. *Preference* influences which BGP route will be chosen for each destination prefix. Changing preference is done by adding/deleting/modifying route attributes in BGP advertisements. Table 1 shows which attributes can be modified during import to control preference locally, and which can be modified during export to change how much a neighbor prefers the route.

2. *Filtering* eliminates certain routes from consideration and also controls who they will be exported to. Filtering may be applied both before preference (inbound filtering) or after preference (outbound filtering). Filtering is done by instructing routers to ignore advertisements with attributes matching certain specified values or ranges.

3. *Tagging* allows an operator to associate additional state with a route, which can be used to coordinate decisions made by a group of routers in an AS, or to share context across AS boundaries. The key mechanism is the *community attribute* [2] [3], a variable-length string used to tag routes. The community attribute is a highly expressive mechanism, lending itself to support a wide variety of complex policies that are difficult to express through other means. For example, one community value might affect how the receiving router sets LocalPref, while another might cause the route to be filtered at another router. However, its expressiveness gives potential for misconfiguration, which is exacerbated by the fact that community attributes usage is not standardized.

An ISP implements its policies by applying configuration commands at routers. These configurations typically consist of a set of lists of preference, filtering, and tagging rules, one list for each *session* the router has with a neighboring BGP-speaking router. Although the configuration language differs between vendors, a key primitive that is often provided is a *route-map*, a language construct used to modify route attributes and define conditions that determine which routes are exported to peers. It consists of two parts: a set of conditions indicating when the map is to be invoked (e.g. the prefix is a specified value, or the AS path matches a specified regular expression), and the action to be taken if the advertisement matches the conditions (e.g. modify a specified attribute, or filter the route).

3 Business relationships

ISPs often wish to control next hop selection so as to reflect agreements or relationships they have with their neighbors. Three common relationships ISPs have are: *customer-provider*, where one ISP pays another to forward its traffic, *peer-peer*, where two ISPs agree that connecting directly to each other (typically without exchanging payment) would mutually benefit both, perhaps because roughly equal amounts of traffic flow between their networks, and *backup* relationships, where two ISPs set up a link between them that is to be used only in the event that the primary routes become unavailable due to failure. There are two key ways these relationships manifest themselves in policy:

Influencing the decision process (by assigning LocalPrefs): ISPs often prefer customer-learned routes over routes learned from peers and providers when both are available. This is often done because sending traffic through customers generates revenue for the ISP while sending traffic through providers costs the ISP money and sending to peers can skew the balance of power in the peering relationship and thereby give incentive to the party receiving more traffic to tear down the relationship or start charging the other party. Often an ISP will achieve this by assigning a non-overlapping range of LocalPref values to each type of peering relationship; for example LocalPref values in the range 90-99 might be used for customers, 80-89 for peers, 70-79 for providers, and 60-69 for backup links. LocalPref can then be varied within each range to do traffic engineering without violating the constraints associated with the business relationship, as described in Section 4. As another example, a large ISP spanning both North America and Europe may wish to avoid forwarding traffic generated by its customers across an expensive transatlantic link. This can be done by configuring its European routers with a higher LocalPref for routes learned from European ISPs, and giving its North American routers a lower LocalPref for these routes.

Controlling route export (by using the community attribute): Routes learned from providers or peers are usually not exported to other providers or peers, because there is no economic incentive for an ISP to forward traffic it receives from one provider or peer to another. This can be done by tagging advertisements with a community attribute signifying the business relationship of the session, and filtering routes with certain community attributes when exporting routes to peers. For example, suppose B wishes to not export routes learned from A to C (Figure 1), perhaps because it does not get paid for transitting traffic from C to A. It can do this as follows. First, for every session routers R_1 and R_2 have with routers in A, B configures an import policy that appends the community attribute X_{peer} to any route learned over these sessions, to indicate that the route was received from a peer—information which is ordinarily lost in BGP as the route propagates across the AS. After appending the community attribute, B exports the route onwards into its internal iBGP network. Second, B configures export policies at R_4 that match on this community attribute to determine which routes get exported to C. In particular, every session between R_4 and a router in C is configured with an export policy that filters any route with the community attribute X_{peer}.

4 Traffic engineering

While business relationships affect relative preferences for routes, there are often several routes available that are equally preferred. Moreover, ISPs often connect at multiple locations to reduce delay and improve reliability, increasing the number of available routes. A secondary goal for many ISPs is to engineer their traffic by modifying preference within the same business class to meet or maximize certain performance criteria (e.g., achieve desired quality and availability). An ISP can do this by modifying the import policies applied by its routers, each of which can have a different configuration. In this section we describe several common traffic engineering goals (a related topic, ensuring the selected routes are stable, is discussed in [4]).

Outbound traffic control (by changing LocalPref and IGP costs): Operators can influence outbound traffic flow either by configuring import policies that affect which routes get in the set of equally-good border routers, or by modifying IGP link costs. One common goal is *early-exit routing* (also called hot-potato routing), where the ISP forwards traffic to its closest possible exit point, so as to reduce the number of links packets traverse and hence the resulting congestion in its internal network. Although early-exit routing is known to inflate end-to-end path lengths in the Internet, ISPs often exercise early-exit routing to reduce their costs and network congestion, and because BGP does not support alternatives like determining global shortest paths across multiple ISPs.

Another common goal is to reduce congestion on outbound links to neighbors. This can be done by *load balancing* traffic over several links when possible. Outbound traffic engineering can be done by changing LocalPref. For example, suppose B wishes to shift some traffic from its links to A to its link to C as shown in Figure 1, perhaps because the link to A is overutilized or because it is planning to take the link down for maintenance. B can reduce the traffic it sends to A and increase traffic it sends to C by decreasing LocalPref for routes traversing A or increasing LocalPref for routes traversing C.

Achieving a specific level of load balance (e.g. balancing load to make spare capacity on both links equal) can be very difficult. The key challenge is to select the proper set of prefixes and change attributes for each appropriately; selecting too large a set will cause too much traffic to shift, overloading one of the links. It can also be tedious to express a long list of prefixes in a router configuration file. Some ISPs deal with this by changing preference for all prefixes whose AS path matches a regular expression, then tweaking the regular expression repeatedly to control how many prefixes match it. However, since this is done manually it is subject to misconfiguration, cannot be done in real time to adjust to changing load, and the outcome from a change can be difficult to predict. There are automated tools that an ISP can use to predict the effects of these actions [5].

Inbound traffic control (by AS prepending and MED): An ISP's internal congestion may be exacerbated by its neighbors, because its neighbors might not be aware of the ISP's traffic-engineering goals, internal topology, or load on internal links due to privacy reasons. Hence, some mechanism to allow an ISP to control how much traffic it receives from each of its peering links is essential. Unfortunately, this is a highly challenging problem, as it requires the local ISP to influence route selection in remote ISPs, which in turn might wish to limit or completely ignore the local ISP's goals. However, an ISP may convince its neighbor (perhaps through economic incentives) to allow the ISP to control how much traffic it receives on each link from the neighbor. This can be done by modifying the MED attribute, which can be used between a pair of ISPs connected via multiple peering links. For example, if B wanted to reduce the amount of traffic traversing router R_1, it could increase the value of the MED attribute R_1 advertises to A, causing the link to R_2 to become more preferred by A's routers and thereby decreasing R_1's load.

Shifting traffic between links to different neighbors is more challenging, as unfortunately BGP was not designed with a mechanism to control route selection in ASes multiple hops away. However, a workaround commonly used is for an AS to prepend multiple copies of its AS number to the AS path in order to artificially inflate the AS-path length. For example, suppose B wishes to shift some traffic from its link to A to its link to C. B can do this by prepending additional copies of its AS number onto the AS paths in BGP advertisements it sends to A. This increases the AS-path length in these advertisements, which causes routes advertised by C to other ISPs to become more desirable in comparison.

Remote control (by changing community attributes): In certain cases, an ISP may need to remotely manage a router's configuration to implement a desired policy. For example in Figure 1, suppose B wishes to have all inbound traffic routed through A, and suppose C peers with A (not shown in the figure). If C has a LocalPref to prefer the direct route to B, no change in MED or AS prepending will force C to use alternate routes through A to B. B could request C to manually change its router configurations, but this can be time consuming for human operators if B changes its policy often (e.g. for traffic engineering purposes). Instead, C can allow B to control C's routing policy with respect to B's routes by configuring its routers to map certain community attributes to certain LocalPref values [2]. If desired, C can limit the degree of B's control to prevent certain policies of its own from being subverted. For example, C can configure its routers to map community value X_1 to a LocalPref of 60, and X_2 to a LocalPref of 75, allowing B to disable the route, but not allowing B to have it chosen over routes C wants to prefer more (by setting a higher LocalPref, like 85).

Remote control has some overlapping functionality with other mechanisms to control inbound and outbound traffic. In general, remote control is typically used to allow a customer to tell its provider to perform some action on its behalf. Remote control provides more flexibility than MED because it allows control of inputs to earlier steps of the decision process like LocalPref, as shown in the example above. Moreover, MED can only change the relative preference of routes, while remote control can be configured to filter routes, or perform AS prepending. Further, MED is only used for routes with the same next-hop AS, while LocalPref is compared across routes learned from all neighbors. However, as with MED, an ISP's neighbors must agree in advance to accept community attributes from the other peer. Also, the highly expressive nature of community attributes introduces potential for misconfiguration. For example, two adjacent ISPs may use the same community attribute to mean very different things (e.g., one might use it for accounting purposes to indicate a certain customer generated the route, while another might use it to indicate the route should be filtered). If a misconfigured router allows the attribute to be passed between them without being removed, unintended consequences could ensue. ISPs typically address this by careful router configuration, and by publishing the list of communities and what actions they trigger for their customers.

In addition, there are a variety of "smart routing" tools [6] that small ASes at the edge of the Internet can use to balance outbound traffic over multiple upstream providers. However, these tools generally are not appropriate for ISPs, as dynamically changing traffic can lead to BGP routing changes that are visible to other ASes, which can trigger flap damping (a mechanism that withdraws unstable routes) if the routes become too unstable. Moreover, these tools focus on load balancing over multiple outgoing links but do not consider the effect on traffic flow inside the AS [5].

5 Scalability

Some misconfigurations and faults in neighboring ISPs can lead them to generate excessive rates of updates. Sending updates too frequently can trigger route instability, leading to poor service quality, or can overload a router's processing capability or memory capacity, which can cause outages and router failure. A properly configured set of BGP policies can improve the resilience of a network to these problems. Common goals include:

Limiting routing table size (by filtering and using the community attribute): ISPs often want to limit routing table size because overflow can cause the router to crash [7]. This can be a particularly important issue for smaller ISPs which may have less expensive routers with less memory capacity.

1. *Protection from other ISPs:* ISPs can protect themselves from excessive advertisements from neighbors by: (a) Filtering long prefixes (e.g., longer than /24) to encourage use of aggregation [8]. (b) As a safety check, routers often maintain a fixed per-session prefix limit that limits the number of prefixes a neighbor can advertise. (c) Default routing: an ISP with a small number of routes may not need the entire routing table, and may instead configure a default route through which most destinations can be reached.
2. *Protecting other ISPs:* An ISP can reduce the number of prefixes it advertises by using *route aggregation*, where instead of advertising two adjacent prefixes (e.g., 4.1.2.0/24 and 4.1.3.0/24) to a neighbor, they can be filtered in the export policy and a less specific prefix (e.g. 4.1.2.0/23) advertised [9]. However, doing this effectively may require knowledge of the neighbor's connectivity (which is not discovered or signaled by the BGP protocol and hence must be manually detected and accounted for by human operators) as illustrated in the following example.

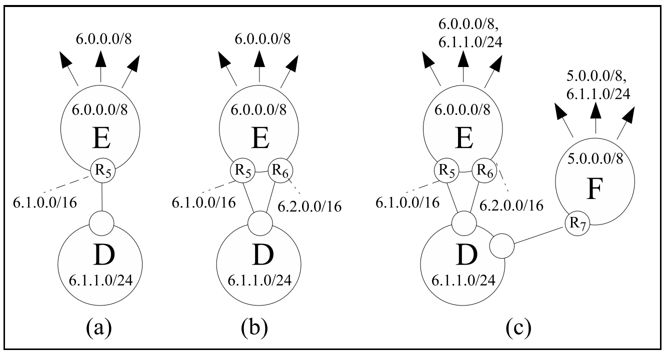

Figure 2: Example topology where adding new customer D triggers E to generate (a) no new advertisements (b) internal advertisement (c) internal and external advertisements.

Suppose E (Figure 2) owns prefix 6.0.0.0/8. E has allocated the subnet 6.1.0.0/16 to router R_5, and has allocated smaller subnets to its customers connected to R_5, including a new customer D which is allocated subnet 6.1.1.0/24. When adding D as a new customer, E may need to make changes to its routers' configuration, and the configuration it chooses impacts whether new advertisements are generated. There are three cases:

1. *No new advertisements:* Suppose D's sole provider is E, and D connects to just one router R_5 in E. In this case, R_5 is already advertising 6.1.0.0/16 within AS E, obviating the need for R_5 to advertise more specific subnets like 6.1.1.0/24. Hence, E just adds a statically configured route at R_5 to forward all traffic in 6.1.1.0/24 to D, and so no advertisements will be sent from E to its neighbors, nor will any new advertisements be sent internally within E.
2. *Internal advertisement:* Suppose instead D connects to two routers R_5 and R_6 in E. In this case, both R_5 and R_6 need to advertise the prefix 6.1.1.0/24 within E, so all routers within E know they can reach D via either R_5 or R_6. However, E can aggregate the advertisement into its address space and hence E will not send BGP advertisements for 6.1.0.0/24 to its neighbors. This is done by configuring R_5 and R_6 to tag a community attribute onto advertisements of prefix 6.1.1.0/24, and configuring all border routers to filter routes with that community attribute.
3. *External and internal advertisement:* Suppose D connects to both E and F. In this case E should not aggregate the prefix into its own address space; if it did, then F would then be advertising a longer prefix route to reach D, and since routers forward packets based on the longest prefix match, all routers in the Internet will prefer F's route

over E's route. If D wishes traffic to flow over both links, it must request that E not perform aggregation on its prefix. E can avoid aggregating the prefix by configuring its routers peering with D to append a certain community attribute, and configure its border routers to export routes containing that community attribute.

Limit the number of routing changes (by suppressing routes that flap): Routing instability is undesirable, as it can increase CPU load on routers, which can increase reaction time to important events. Also, frequent shifting of traffic to different paths can introduce jitter and packet loss in applications like Voice-over-IP and interfere with TCP's round-trip-time calculations. The key mechanism used to improve routing stability is *flap damping*. Flap damping is a mechanism that limits propagation of unstable routes. It works by maintaining a penalty value associated with the route that is incremented whenever an update is received. When the penalty value surpasses a configurable threshold, the route is *suppressed* for some time, i.e., it is made unavailable to the decision process and hence will not be selected. An ISP can lower the penalty threshold to improve route stability at the cost of worsening availability. ISPs may wish to less aggressively dampen or disable damping for certain prefixes, for example routes to the root Domain Name System servers, or routes from customers with high availability requirements. Also, ISPs sometimes more aggressively dampen longer prefixes than shorter prefixes, with the motivation that damping a shorter prefix can have a large effect on reachability [10]. This can be done by configuring a route-map that matches on the prefix length or a specific prefix and sets the flap damping parameters accordingly.

6 Security

An AS is highly vulnerable to false information in BGP updates. By sending false information, an ISP can subvert a neighbor's routing goals, cause routers to overload and fail, or degrade service quality. False information can have a significant influence on routing in an AS, even if the source of the information is several AS hops away [11]. Such information is sometimes generated by router bugs and misconfiguration. It could also be maliciously generated by an ISP's neighbor, who may be competing for customers and hence has a vested interest in making the ISP's customers dissatisfied with service. Hence an ISP may wish to exercise *defensive programming* to protect itself against attacks.

Discarding invalid routes (by import filtering): ISPs may wish to protect their customers from learning invalid routes by performing sanity checks to ensure update contents are valid before propagating them internally. For example, routes to special-use or private addresses, or address blocks that have not yet been allocated are obviously invalid [12]. Moreover, advertisements from customers for prefixes they do not own should not be propagated. ISPs can also perform certain sanity checks on the AS path; for example a Tier-1 ISP should not accept any routes from its customers that contain another Tier-1 ISP in the AS path. Also, advertisements containing private AS numbers in the AS path may be considered invalid. ISPs may configure its filters based on the contents of public repositories of routing configurations called *routing registries*, other public reports [13], or private disclosures from neighbors.

Protect integrity of routing policies (by rewriting attributes): An ISP may want to prevent a neighboring AS from having undue influence over its routing decisions, in violation of their peering agreement. Otherwise, the ISP could be duped into carrying traffic a longer distance across its backbone on the neighbor's behalf. For example, suppose the ISP peers with a neighbor in both New York and San Francisco. By advertising a prefix with a MED of 0 in New York and a MED of 1 in San Francisco, the peer could trick the ISP into having all of its routers direct traffic for this destination through the New York peering point, even if the San Francisco peering point is closer. The peer could achieve the same goal by configuring its San Francisco router to advertise the route with the next-hop attribute wrongly set to the IP address of the New York router. To defend against violations of peering agreements, the ISP can configure the import policy to delete attributes or overwrite them with the expected values. For example, the import policy could set all MED values to 0, unless the ISP has agreed in advance to honor the neighbor's MEDs. Similarly, the import policy could set the next-hop attribute to the IP address of the remote end of the BGP session, and remove any unexpected community values. Unfortunately, these techniques are not sufficient to prevent all violations of peering agreements[1].

Securing the network infrastructure (by export filtering): An ISP may wish to prevent external entities from accessing certain internal resources by configuring its *export policies* that filter BGP advertisements for destinations that should not be externally reachable. For example, the ISP may protect its own backbone infrastructure by filtering the IP addresses used to number the router interfaces. The ISP may also wish to protect certain key internal services, by filtering the addresses of the hosts running network-management software. Finally, as a courtesy to its neighbors, an ISP may also do export filtering of invalid routes (e.g., routes with invalid addresses or contents), as a preventative measure.

Blocking denial-of-service attacks (by filtering and damping): Denial-of-service attacks can degrade service by overloading the routers with extra BGP update messages or consuming excessive amounts of link bandwidth. For example, the ISP's routers could run out of memory if a neighbor sends route advertisements for a large number of destination prefixes. To protect itself, the ISP can configure each BGP session with a maximum acceptable number of prefixes, tearing down the session when the limit is exceeded; in addition, the import policy could filter prefixes with large mask lengths (e.g., longer than $/24$). As another example, a neighbor sending an excessive number of BGP update messages can easily deplete the CPU resources on the ISP's routers. Upon detecting the excessive BGP updates, the operators could modify the import policy to discard advertisements for the offending prefixes or disable the BGP session. Upon identifying the neighbor or prefix

[1]For example, many peering contracts require a peer to announce a prefix at all peering points, with AS paths of the same length [14, 15]. An ISP can detect this kind of inconsistency by comparing the BGP advertisements across all peering points [16] or collecting detailed measurements of the traffic traversing the peering links.

responsible for the excessive BGP updates, the ISP can more aggressively dampen (Section 5) or even completely filter updates it receives from these sources. In addition to BGP's own vulnerabilities to attack, an ISP (or its customers) may be subject to a denial-of-service attack where excessive data traffic is sent to victim hosts. An ISP can block the offending traffic by installing a *blackhole route* that drops traffic destined to the victim addresses. Blackhole routes may be statically configured, or operators may run a special BGP session that advertises the prefixes of the victims [17]. Routers receiving prefixes on this session then assign the next-hop to be an address associated with the "null" route (a route which drops all traffic), or the address of a monitoring system that can perform further analysis of the traffic. Using a similar technique, the ISP can advertise the address blocks of known spammers to blackhole traffic sent to these addresses. These blackhole routes prevent the spammers from establishing bidirectional communication (i.e., a TCP connection, which depends on receiving a SYN-ACK packet) with the ISP's mail servers.

7 Looking forward

BGP's rich feature set of tunable knobs and complex cross-protocol interactions make it highly subject to a variety of problems, including misconfiguration, oscillations, and protocol divergence. The challenge of supporting many different complex policies in BGP without significantly complicating the protocol or degrading its performance has led to much research activity. Three key areas of research related to BGP policy are discussed below (a wider survey of research directions is given in [18]).

Configuration checking: The complexity of Internet routing makes it difficult to predict the way policies interact, increasing the prevalence of configuration mistakes. Interdependence of policy across ISPs and within a single ISP can trigger problems like persistent route oscillations. Configuration checking tools can avoid misconfigurations by verifying certain consistency criteria hold [19], and modeling tools can predict side-effects of configuration changes on routers within an ISP [5]. Across ISPs, uncoordinated routing policy can worsen route convergence and stability. The Routing Arbiter [13] project introduced a distributed architecture for publishing and coordinating routing policies so as to avoid these problems, but was not widely deployed. Other work has attempted to coordinate route policy selection across ISPs without revealing private details of policies [20].

Language design: Routing Policy Specification Language (RPSL) [21] is a vendor-neutral language proposed to describe an ISP's policy. It was envisioned these descriptions could be bound together in a database and checked for consistency [13]. RPSL, though mature, is somewhat low-level and mechanism oriented. It may be possible to substantially improve upon RPSL by designing router configuration languages with higher level constructs that allow diverse policies while precluding certain misconfigurations, enforcing certain consistency properties to hold, simplifying configuration of certain common design patterns [22], however the design of such a language remains an open problem.

New architectures: There are several routing architectures aimed at fixing problems in and extending functionality of BGP. HLP [23] is a proposed replacement for eBGP. The design philosophy of HLP is to expose common policies that can typically be inferred in BGP today and optimize the routing protocol based on the resulting structure, with the aim to improve scalability and convergence of interdomain routes. Routing Control Platform (RCP) [24] is a logically centralized system that computes and distributes routes to routers inside an ISP. The centralization allows policies to be applied at the AS level, and the RCP applies the policies and its own decision process to select the best BGP route for each destination prefix on behalf of each router. This simplifies the configuration and application of policies and avoids misconfiguration.

8 Conclusion

Although BGP policies can be highly complex, there are a number of common design patterns that are typically used by ISPs. In this article we discussed several common patterns and how they can be realized using BGP policy mechanisms. We believe that by recognizing these patterns exist we can more efficiently develop tools that directly support them, such as analysis tools that check correctness, languages that preclude errors, or architectures that are designed for common cases.

Acknowledgments: We would like to thank Nick Feamster, Karthik Lakshminarayanan, Kobus van der Merwe, Fang Yu, Gautam Altekar, and Dilip Joseph for reading earlier drafts and providing invaluable feedback.

References

[1] Y. Rekhter, T. Li, "A border gateway protocol 4," IETF RFC 1771, March 1995.

[2] E. Chen, T. Bates, "An application of the BGP community attribute," IETF RFC 1998, August 1996.

[3] B. Quoitin, O. Bonaventure, "A survey of the utilization of the BGP community attribute," expired Internet Draft *draft-quoitin-bgp-comm-survey-00.txt*, February 2002.

[4] Y. Yang, H. Xie, H. Wang, A. Silberschatz, Y. Liu, L. Li, A. Krishnamurthy, "On route selection for interdomain traffic engineering," *IEEE Network Magazine, Special issue on Interdomain Routing*, Nov-Dec 2005.

[5] N. Feamster, J. Winick, J. Rexford, "A model of BGP routing for network engineering," in *Proc. ACM SIGMETRICS*, June 2004.

[6] J. Bartlett, "Optimizing multi-homed connections," in *Business Communications Review*, vol. 32, no. 1, pgs. 22-27, January 2002.

[7] D. Chang, R. Govindan, J. Heidemann, "An empirical study of router response to large BGP routing table load" in *Proc. Internet Measurement Workshop*, November 2002.

[8] S. Bellovin, R. Bush, T. Griffin, J. Rexford, "Slowing routing table growth by filtering based on address allocation policies," unpublished, June 2001, `http://www.cs.princeton.edu/~jrex/papers/filter.pdf`

[9] E. Chen, J. Stewart, "A framework for inter-domain route aggregation," IETF RFC 2519, February 1999.

[10] "RIPE routing-WG recommendations for coordinated flap damping parameters," October 2001, `http://www.ripe.net/ripe/docs/routeflap-damping.html`

[11] O. Nordstrom, C. Dovrolis, "Beware of BGP attacks," in *ACM SIGCOMM Computer Communications Review*, April 2004.

[12] N. Feamster, J. Jung, H. Balakrishnan, "An empirical study of "Bogon" route advertisements," in *ACM SIGCOMM Computer Communications Review*, January 2005.

[13] R. Govindan, C. Alaettinoglu, G. Eddy, D. Kessens, S. Kumar, W. Lee, "An Architecture for Stable, Analyzable Internet Routing," *IEEE Network Magazine*, Jan-Feb 1999.

[14] D. Golding, V. Gill, V. Mehta, "America Online: Settlement-free interconnection policy," web site, `http://www.atdn.net/settlement_free_int.shtml`

[15] UUNET, “MCI policy for settlement-free interconnection,” web site http://global.mci.com/uunet/peering/

[16] N. Feamster, Z. Mao, J. Rexford, “BorderGuard: Detecting cold potatoes from peers,” in *Proc. Internet Measurement Conference*, October 2004.

[17] D. Turk, “Configuring BGP to block Denial-of-Service attacks,” IETF RFC 3882, September 2004.

[18] M. Yannuzzi, X. Masip-Bruin, O. Bonaventure, “Open issues in interdomain routing: a survey,” *IEEE Network Magazine, Special issue on Interdomain Routing*, Nov-Dec 2005.

[19] N. Feamster, H. Balakrishnan, “Detecting BGP configuration faults with static analysis,” in *Proc. Networked Systems Design and Implementation*, May 2005.

[20] R. Mahajan, D. Wetherall, T. Anderson, “Negotiation based routing between neighboring domains,” in *Proc. Networked Systems Design and Implementation*, May 2005.

[21] A. Alaettinoglu, C. Villamizar, E. Gerich, D. Kessens, D. Meyer, T. Bates, D. Karrenberg, M. Terpstra, “Routing policy specification language (RPSL),” IETF RFC 2622, June 1999.

[22] T. Griffin, A. Jaggard, V. Ramachandran, “Design principles of policy languages for path vector protocols,” in *Proc. ACM SIGCOMM*, August 2003.

[23] L. Subramanian, M. Caesar, C. Ee, M. Handley, Z. Mao, S. Shenker, I. Stoica, “HLP: A next-generation interdomain routing protocol,” in *Proc. ACM SIGCOMM*, August 2005

[24] M. Caesar, D. Caldwell, N. Feamster, J. Rexford, A. Shaikh, J. van der Merwe, “Design and implementation of a routing control platform,” in *Proc. Networked Systems Design and Implementation*, May 2005.

[25] D. Golding, “Routing policy tutorial,” in *NANOG 24*, http://www.nanog.org/mtg-0202/ppt/golding/index.htm