Locating Internet Routing Instabilities

Anja Feldmann
TU-München
anja@in.tum.de

Olaf Maennel
TU-München
olaf@maennel.net

Z. Morley Mao
Univ of Michigan
zmao@eecs.umich.edu

Arthur Berger
MIT/Akamai Technologies
arthur@akamai.com

Bruce Maggs
CMU/Akamai Technologies
bmm@cs.cmu.edu

ABSTRACT

This paper presents a methodology for identifying the autonomous system (or systems) responsible when a routing change is observed and propagated by BGP. The origin of such a routing instability is deduced by examining and correlating BGP updates for many prefixes gathered at many observation points. Although interpreting BGP updates can be perplexing, we find that we can pinpoint the origin to either a single AS or a session between two ASes in most cases. We verify our methodology in two phases. First, we perform simulations on an AS topology derived from actual BGP updates using routing policies that are compatible with inferred peering/customer/provider relationships. In these simulations, in which network and router behavior are "ideal", we inject inter-AS link failures and demonstrate that our methodology can effectively identify most origins of instability. We then develop several heuristics to cope with the limitations of the actual BGP update propagation process and monitoring infrastructure, and apply our methodology and evaluation techniques to actual BGP updates gathered at hundreds of observation points. This approach of relying on data from BGP simulations as well as from measurements enables us to evaluate the inference quality achieved by our approach under ideal situations and how it is correlated with the actual quality and the number of observation points.

Categories and Subject Descriptors C.2.2 [**Computer Communication Networks**]: Routing Protocols
General Terms: Measurement, Analysis, Simulation
Keywords: BGP, root cause analysis, routing instability, instability origin

1. INTRODUCTION

Although routing dynamics and especially BGP [1] dynamics have been extensively studied within the last few years, *e.g.,* [2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17], they are still poorly understood. In his work on developing a signal propagation model for BGP updates, T. Griffin [2] observed that "In practice, BGP updates are perplexing and interpretation is very difficult".

Despite the difficulties in working with BGP updates, we propose a methodology for locating the origins of routing instabilities. Our approach is to correlate information from updates across observation points (views) and across prefixes. In contrast to others [4, 3, 5] we propose to first correlate across time, then views, and finally prefixes. Each instability (any change of BGP advertisement over an EBGP session) implies that some BGP attribute changes for some prefixes are propagated via BGP updates throughout the autonomous system (AS) topology. *Our main insight is that if there is an AS path change, then some instability has to have occurred on one of two AS paths, the previous best path or the new best path.* Furthermore, if there is only an attribute change, then it has to be on the AS path. Using multiple vantage points, one can then pinpoint an instability. Under certain conditions (see Section 2.3) this instability corresponds to the original cause of the routing change.

We verify our approach in a novel way. In particular, we use a simulator (Section 6) which is based on an AS topology derived from actual BGP updates, and which uses BGP policies that are compatible with the inferred peering/customer/upstream relationships among the ASes. In the simulation, network and protocol behavior are ideal. Through simulation we learn what inference quality is achievable and how it is correlated with the number of observation points and the location of the observation points.

We then apply our methodology and evaluation technique to the same actual BGP updates gathered at more than $1,100$ observation points to more than 650 ASes including the ones from RIPE RIS [18], University of Oregon RouteView [19], and more than 700 ASes from Akamai Technologies (Section 5). To cope with the complexity of BGP (Section 3), we develop several heuristics (Section 4) to deal with the limitations of real BGP updates, such as update propagation, AS path exploration, MRAI timer, route-flap damping, absent updates, multiple instability events, as well as missing information. Overall we find (Sections 6 and 7) that we can pinpoint a likely origin of instability to a single AS or a session between two ASes in most cases even without correlating across prefixes. For further validation, we correlated the inferred instability with router syslog data from a tier-1 ISP. We are able to confirm that 75% of the inferences where this ISP is identified to be responsible for originating an instability coincides with a BGP session reset.

To summarize our contributions: We present a methodology for identifying origin of instability visible in BGP routing changes along three dimensions: time, prefix, and view. Our approach is thorough, as we take into consideration complex BGP operational issues, but it is simple and intuitive. It is based on how BGP path selection operates – some routing instability lies on either the previous or the changed stable paths. To show improvement over previous work, we also illustrate through detailed examples that sim-

SIGCOMM'04, Aug. 30–Sept. 3, 2004, Portland, Oregon, USA.

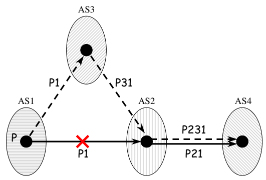


Figure 1: Example AS topology.

foreach instability event of prefix p
    foreach observation point o
        if route change with path change: from r_p to r_n
            r_b = best_path(r_p, r_n)
            candidate_set c_o = candidates(r_b)
        if route change without path change: $r_p == r_n$
            r_b = best_path(r_n)
            candidate_set c_o = candidates(r_b)
        if route r_s is stable (no BGP update)
            candidate_set s_o = candidates(r_s)
    instability candidates = $\cap c_o - \cup s_o$

Figure 2: Per prefix – ideal meth. for locating instabilities.

plified assumptions do not hold in practice. Furthermore, a main distinction of our work is that we use simulation as a validation methodology on an accurate, fairly complete AS topology to understand when we can and cannot narrow down the instability origin and the effect of vantage points on the inference. Finally, we apply our inference methodology on a large set of BGP data from diverse vantage points.

2. IDEAL METHODOLOGY

The goal of this Section is to propose a methodology for inferring the origin of routing instabilities from their effects – the results of the BGP convergence process. Each instability may cause BGP updates to propagate through the Internet which can be observed at various monitoring points throughout the network. We use these updates to identify the instability origins. Moreover, the methodology is applicable to any other path vector routing protocols.

We refer to the location of a routing instability, either internal to an AS or between two ASes with BGP peering session(s), as an *instability origin*, and refer to the resulting sequence of BGP update messages as an *instability burst*. The specific instability burst observed at particular points on the Internet via monitoring sessions differs according to the location of the observation point (also referred to as view), the instability origin, the policies of the ASes along the AS path, the effects of timing imposed on the message ordering, and the AS topology itself. When the instability cause is due to an event internal to a given AS, excluding EBGP sessions to its neighbors, we say the cause is located in the given AS. When the cause of the instability is due to an event at an EBGP session between two given ASes, we say the cause is located at the edge between the two given ASes. (Note that in the latter case we do not try to determine which router at either end of the EBGP session initiated the event.)

2.1 Basic methodology

Let us consider the example AS topology in Figure 1 where AS1 is originating a route to prefix P. Assume the single link between AS1 and AS2 fails. In this case, the best BGP route at AS2 and AS4 changes from the solidly marked one to the dashed one. Given EBGP monitoring sessions to AS2, AS3 and AS4 (not shown in the figure), one will observe BGP updates at AS2, similar to the ones propagated to AS4, but none at AS3. The best path propagated by AS2 changes from P:21 to P:231. This is the kind of information that we take advantage of. In this case we can narrow the cause of the routing instability to AS2, or the edge between AS1 and AS2. The main idea is that when there is a change in the best BGP path, the origin of instability is either on the new path or on the old path or induces another instability on either of the two paths. Furthermore, the original or the induced instability must be on whichever of these two paths is "better" when compared head-to-head. To see this, notice that if the old path was better, then there would be no path change without instability on the old path. On the other hand, if the new path is better, then there must have been some instability on the new path. While it is not always obvious to an outside observer which of the two paths (old and new) is better, it is possible to derive a set of candidates for the instability by taking the union of the two paths. Alternative heuristics that are more aggressive are presented in Section 4. For example, here one may presume that the best path is the one with shorter path length: P:21. Similarly, the EBGP monitor at AS4 sees previous and new paths of P:421 and P:4321, and may presume that the best path is P:421.

Using information from multiple monitoring sessions helps narrow down the origin of the instability. Assume that the instability under consideration is the only instability during some time period. Then all path changes for the prefix P are due to this instability. This implies that the instability is visible at each observation point receiving BGP updates for P, which means that it is present in the intersection of the corresponding candidate sets. In the present example, the intersection from the EBGP monitoring sessions at AS2 and AS4 yields the candidate set of AS1, AS2, and the edge between AS1 and AS2.

The lack of a BGP update is another information source. It indicates a stable best path which implies that the current path does not suffer any instabilities. Thus, the candidate set can be further reduced by *excluding* the union of the candidate sets from those observation points without BGP updates. In this example, the lack of updates at the EBGP monitoring session at AS3 implies that the instability is neither within AS1, AS3 nor at the edge AS1 to AS3. Thus, AS1 can be removed as a possible cause of the instability, and the resulting candidate set is AS2 and the edge between AS1 and AS2. This basic approach is summarized in Figure 2.

This ideal methodology assumes the following:

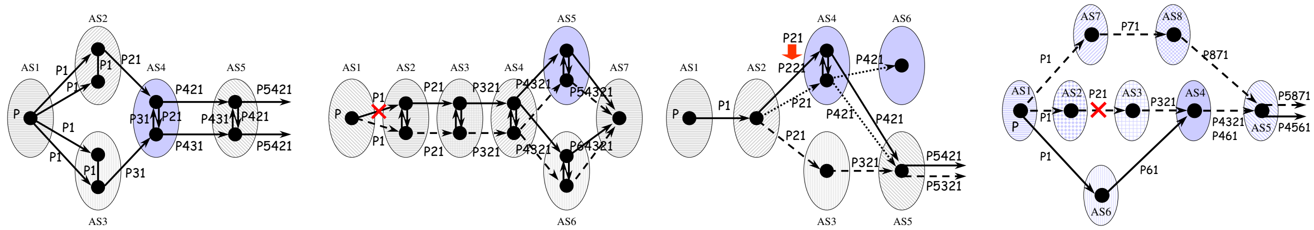
1. All updates caused by an instability event are identifiable.
2. At any time each prefix is only hit by one instability event.
3. BGP convergence finishes within some time period.
4. We can determine which paths are stable.
5. We can determine which of two BGP paths is better.
6. There are no *induced* instabilities (see Section 2.2).

While any of the above may not apply with actual BGP update data, it is possible to develop heuristics to deal with each violation of these assumptions as described in Section 4. Furthermore, it is possible to evaluate the methodology using simulations, see Section 6. This enables us to calibrate our expectations.

2.2 Cautions

Next we illustrate using simple examples why the details and the assumptions matter when trying to locate routing instabilities. For a related discussion see [11].

Caution on excluding candidate ASes: Suppose at a given observation point and for a given prefix P one sees previous and new stable paths of P:7,6,5,4,3,2,1 and P:7,6,5,8, 3,2,1 respectively. One might think that AS 7 or 6, or 2 or 1 could not be the cause of the



(a) Example topology 1 (b) Example topology 2 (c) Example topology 3 (d) Example topology 4

Figure 3: Example AS topologies

routing change, and thus these shared segments of the two paths could be excluded from the candidate set. However, such inference can be erroneous and is not made in the "ideal methodology" of Figure 2. The following two examples show that if the previously best and the new best path share segments, it can be important to include the shared segments in the candidate set. This is where our methodology differs from the approach by Chang et al. [4] which can incorrectly exclude some instability originators.

Consider the example shown in Figure 3(a). Customer, AS1, is multi-homed to two providers, AS2 and AS3. Both providers, AS2 and AS3, have the same upstream provider AS4. And AS4 peers with AS5 at two peering points. AS4 learns of the given prefix from AS2, and may propagate the path P:421 to AS5 on one of the two peering sessions, the top one in Figure 3(a). Likewise, AS4 also learns of the given prefix from AS3, and may propagate the path P:431 to AS5 on the second of the two peering sessions, the bottom one in Figure 3(a). With cold potato routing [20], AS5 chooses to announce the single route P:5421 to other ASes, including an EBGP session with an observation point (not shown in the figure). Now of key interest in the present example, due to say an internal failure within AS5 or an operator in AS5 intentionally changing IGP costs, the route announced by AS5 to other ASes changes from P:5421 to P:5431. Thus AS5 is the instability originator even though the AS path change is at a different location, *i.e.,* from 421 to 431. Thus the tie breakers cause situations in which changes within a remote AS can lead to AS path changes in the initial (closer to the origin AS) segment of the AS path.

Furthermore, consider the example shown in Figure 3(b). Here the customer, AS1, is again multi-homed but to only a single provider and originates prefix P. AS2, as well as AS3, and AS4 all use cold potato routing in the sense that they use the IGP metrics to initialize the MED values within the BGP updates. In this case it might well be that AS5 uses a route which is propagated along the solidly marked path while AS6 is using the dashed one. Lastly, based on received MED values, next hop IP's and IGP costs, AS7 chooses to announce path P:754321 on an EBGP session to some observation point (not shown in the figure). Now assume that the solid link between AS1 and AS2 fails. In this case the next hop of the BGP update from AS2 to AS3 together with the MED value will change. This causes all routers within AS3 to change their best path to the dashed path. This implies that the BGP update from AS3 to AS4 will have a different next hop and a different MED value. This will cause AS5 and AS6 to announce new next hop and different MED values to AS7. Since the next hop and MED values received by AS7 have changed, AS7 may change its preference from AS5 to AS6, and announce a new path of P:764321 to the observation point. Thus, a link failure in or between some AS near the origin AS, *i.e.,* AS1 to AS2, can cause an AS path change at a subsequent location on the path, *i.e.,* from P:754 to P:764. If one imagines a slightly more complex internal topology, even changes to IGP metrics within an AS can have such an effect. Using IGP metrics as MED values creates a link between internal changes and external effects and therefore between distant ASes. Similar effects are possible using communities.

Caution on instability propagation: Figure 3(c) shows the danger of assuming that all instabilities are propagated. In this specific case, AS6 uses the dotted route to prefix P, P:6421, while AS5 uses the solid one P:5421. Now suppose that AS2 does AS path prepending on one of the EBGP sessions with AS4, and that this causes AS4 to switch its best route for P to the dotted one. This change has no impact on AS6 since its route does not change. AS5 will receive updates since the IGP/MED values within AS4 changed. This may cause AS5 to switch to the dashed path via AS3, P:5321. Hence we have a situation where the best path of AS6, in the sense of AS-level path P:6421, has an instability, but AS6 will not receive a corresponding BGP update. This can be achieved via IGP/MED coupling and filters, *e.g.,* using communities. In essence this problem corresponds to the previous problem.

Caution regarding induced updates: Figure 3(d) shows the danger of assuming that the origin of all instabilities is in either on the new or on the old path. In this specific case, AS4 prefers the route P:321 for prefix P instead of the route P:61. Accordingly it advertises the route P:4321 to AS5, and AS5 advertises P:5871 to an observer. If the link between AS2 and AS3 goes down, AS4 revises its advertisement to AS5 to the route P:461. If now AS5 prefers the route P:461 over the route P:871 it will advertise the route P:5461 to an observer. Thus the observer sees the route to P change from path P:5871 to P:5461, even though the original failure is the link between AS2 and AS3. In this case the original failure *induced* or triggered a route change at AS5. While the routing decision at AS4 may seem unorthodox, it is nevertheless coherent in the sense that AS4 uses a consistent ranking of the paths. Induced updates can occur if the ranking of routes differs between providers. Our methodology is capable of locating the AS where the route change is induced, but may not be able to locate the original cause of the instability. On the one hand this is disappointing, yet on the other hand locating the induced instability already reduces the problem and is valuable in itself. The problem introduced by induced updates is that the intersections can be empty, if a subset of the observers point towards the original instability and another subset to an induced update, or even incorrect, *e.g.,* if A is the instability origin AS, B the AS at which an update is induced, and A, C is the subset that one subset of the observer identifies, and B, C is identified by another subset. Nevertheless it is the case that each set of observers can be partitioned in such a way that the intersection of the union of the AS paths will include either one AS at which an update is induced or the original instability.

2.3 Identifying link changes

The cautionary examples highlight that the union and the intersection rules are only heuristics. Yet these are sensible heuristics and we now provide some formal justification. In particular, we analyze the effectiveness of the union heuristic in the simplified model of BGP that is realized by our simulator. Each of the theorems in this section relies on one or more of the following assumptions.

ASSUMPTION 2.1. *The simplified BGP model assumes:*

a) The only events in the network are link failures and link restorations, and the network fully converges to new routes between successive events.

b) Routes are chosen based on the AS Path attribute only. Other attributes such as MED and next hop are not considered in calculating local preferences. There is at most one peering session between any pair of ASes.

c) For each destination and for each AS, routes are chosen based on a total order over all possible AS paths to the destination. Although the list of paths available to an AS may change over time, the total order over all possible paths never changes.

The following theorems relate to the union rule. Suppose that an event has occurred, and that as a result, the AS path from an observer (AS O) to a destination (AS D) has changed.

THEOREM 2.2. *Suppose Assumptions 2.1 hold. If observer O sees its path to destination D change, then on either the old path or the new path, at least one AS changes the advertisement for D that it sends to its predecessor on the path.*

Proof: If no AS on either the old or new paths changes its advertisement, then both paths were already available to O, and remain available. Since, by Assumption 2.1(c), paths are chosen according to a fixed total ordering, the old path remains preferred over the new path. □

Observe that, by Theorem 2.2, on either the old path or the new path from O to D, there is a maximal prefix of ASes such that every AS on the path changed its advertisement to its predecessor on the path. By definition, the last AS on the prefix did not receive a new advertisement for D from its successor on this path. Call this last AS on the prefix Y, its successor Z, and its predecessor X.

THEOREM 2.3. *Suppose Assumptions 2.1 hold. If Y is on the old path, then it either observed a link failure on the old path or received a new advertisement for a path to D from outside the old path. If Y is on the new path, then it either observed a link restoration on the new path, or it received an advertisement withdrawing a path to D from outside the new path.*

Proof: Suppose that X, Y, and Z lie on the old path, and Y changed its advertisement to its predecessor X on the old path. Since Y is receiving the same advertisement for D from its successor Z on the old path, then either the link between X and Y failed (and hence Y could no longer advertise across it), or Y must have learned of a new path to D from outside the old path that it prefers over the old path. If, on the other hand, X, Y, and Z lie on the new path, then either the link between Y and Z was restored, or a path to D that Y prefers over the new path was withdrawn from outside the new path. □

THEOREM 2.4. *Suppose Assumptions 2.1 hold. Consider the common prefix of the old and new paths from O to D. If Y appears on this prefix, it can only appear as the AS closest to D.*

Proof: The proof is by contradiction. If Y is on the common prefix with respect to the old path (but not the AS closest to D), then it must appear (in the same position) with respect to the new path, and vice versa. Since the link between Y and X has neither failed nor been restored (it appears on both the old and new paths), it must be that Z either learned of a new path from outside old path, or saw a path withdrawn from outside the new path. But both the old path and new path were available to Y before the event, and are still available to Y after the event, and (by Assumption 2.1(c)) no event observed by Y can change its preference of the old path over the new path. □

THEOREM 2.5. *Suppose Assumptions 2.1 hold. Consider the common suffix of the old and new paths from O to D. If Y lies on this suffix, then it must appear as the AS farthest from D.*

Proof: The proof is by contradiction. If Y appears on the common suffix but is not farthest from D, then Y appears in the same position with respect to both the old and new paths. On both paths, Y receives the same advertisement from its successor Z, and sends the same advertisement to its predecessor X. This contradicts the definition of Y. □

2.4 Consideration of multiple prefixes

So far we have only considered two of the possible three dimensions [3] for inferring the origin of routing instabilities: *time, views,* but not multiple *prefixes*. Since it is quite likely that multiple prefixes use the same BGP session/same link/same AS on their AS path, a failure to any of the latter will cause changes to multiple best paths. This implies that if a prefix is affected by only a single instability during some time, then we can identify correlated events. One approach is to use a **Greedy heuristic** which starts with a set $\mathcal{P}$ that includes all prefixes with instabilities during this time window. For all prefixes within the set $\mathcal{P}$, count how often each AS topology component appears across the instability candidate sets associated with these prefixes. Choose the most frequented AS topology component E as the most likely instability cause/origin, and subtract the prefixes from $\mathcal{P}$ that include E in their candidate set. The algorithm continues until $\mathcal{P}$ is empty which means that all prefixes have been assigned an instability "origin".

3. ORIGINS OF INSTABILITIES IN BGP

To apply the above ideal methodology to actual BGP updates we have to use a number of heuristics to deal with the assumptions. Furthermore, to assess the sensibility of the proposed Greedy heuristic one has to have a better understanding of the dependence across BGP updates for multiple prefixes. Accordingly, we delve into some BGP details and discuss what kind of instability creators exist, how instabilities propagate through the actual network and what kind of updates may be visible at an observation point.

3.1 Instability creators

A BGP instability is an event that impacts inter-AS routing, see Figure 4. We exclude from the notion of an "event" the receipt of an EBGP update message. Rather, we consider the EBGP update message as a consequence of some instability. That is, in response to a BGP instability, a BGP speaking router initiates a BGP update that propagates an attribute change from one BGP peer to another. Before reviewing the types of BGP instabilities, we review the main steps of how BGP chooses its routes. For each BGP session, the input filter policies, which can rewrite the BGP attributes,

Instability	Examples
BGP session availability	session establishment/teardown/reset
BGP session filters	filter changes and/or BGP attribute manipulations usually imply session (soft-)reset or graceful restart
IGP costs changes	IGP metric changes, link or node failures/repairs
IP address changes	renumbering, link or node failures/repairs
link/node availability	link failures/repairs, node failures/repairs may cause BGP session availability changes and IGP cost changes
originator changes	addition/deletion of network prefixes
route flap damping	delay of the propagation of updates

Figure 4: BGP instability and their typical causes.

are applied first. Then the BGP decision process considers a priority list of attributes to select the best path. If the best route changes, then the routing table is updated, and the new best route is passed through the output filter policies, which can again rewrite the BGP attributes. Finally the updates are propagated to the BGP peer.

Note that changes to filter policies can originate BGP instabilities, since each BGP configuration change implies that the BGP peers have to reconcile their databases. Accordingly, updates that were previously filtered may now be considered during the best path selection step or updates that were previously selected as best path may now be filtered. Accordingly, BGP instabilities can have their origin at the source of the prefix, in the input filters, the decision process, the output filters, or through the availability of BGP sessions. While the filters are limited to the BGP attributes, the decision process also uses the following other resources: link availability, node reachability, IGP cost, and next hop IP addresses.

Accordingly, the BGP instabilities can be initiated by: changes to the availability of BGP sessions, the BGP session filters, the link and/or node availability, introduction of withdraw of prefixes including aggregation, IGP cost changes, or IP address changes. Typical examples for each of these are given in Figure 4. Note that one kind of change, *e.g.,* a node failure, may imply other failures, *e.g.,* multiple link failures, which can in turn imply other changes, *e.g.,* IGP cost changes.

Next we consider what kind of BGP updates these BGP instabilities impose. Here the first question is which prefixes will see any updates, referred to as *relevant prefixes*. An instability is relevant to a prefix if an attribute of its best path or if its filter policy is changed. In terms of blaming an AS for an instability, one has to distinguish between changes within an AS, called *internal changes*, and between ASes, *called external* changes. Typical internal changes are those associated with IBGP sessions. Others are IGP traffic engineering operations changing IGP metrics. Typical external changes are changes to EBGP sessions, *e.g.,* for the purpose of traffic engineering and may include subaggregation, or aggregation of prefixes, changing filter rules, AS path prepending, *etc.* The difference between internal and external changes is that the latter usually only impacts the prefixes whose best path includes that session and therefore both ASes. Internal changes can impact prefixes with diverse next hop and previous hop ASes.

The next question is what kind of updates will a prefix experience. One important factor is the diversity of routes available to the best path selection process. A link or BGP session failure can disrupt the connectivity between two ASes and affect many prefixes. The existence of an alternative route causes the selection of a new best path, which will not be propagated if it has the same attribute values or if it is caught in the output filter. Otherwise it announces the existence of an alternative route. This route may differ from the old one in either the AS path, the next hop, or other attribute changes. An AS path change is necessary if reachability via the old AS path is no longer given, *e.g.,* if two ASes have a single EBGP session and it fails, or if an internal link failure causes a network split, or if the reachability via the new AS path is more attractive.

Yet, by design not all instabilities impact reachability, *e.g.,* peering usually requires BGP sessions at at least three diverse locations and ASes usually have multiple upstream providers. This diversity implies that the addition of a new route or the withdraw of a route usually just adds one more variant to the best path selection process, *e.g.,* if two EBGP sessions exist between two ASes one may expect to learn two routes to each prefix routed via these sessions which, if a consistent routing policy is used, will have the same AS path. Accordingly, the BGP decision process may choose between multiple routes with the same AS path. In addition, if the prefix is reachable via another AS, further alternatives are available to the decision process. If the routes have the same length AS path, the decision about which route is best depends on the MED values, the IGP distance metrics, and the next hop IP addresses. Accordingly, the addition of a new route or the retraction of the best route can, ignoring steps above AS path change in the BGP decision process, *e.g.,* local preference, either lead to a AS path change with and without next hop change, a next hop IP address change without AS path change, or no change at all if there are multiple peerings between the same routers. In this case, each router may make a different decision which implies that AS path changes are likely to occur for only a subset of the relevant ASes.

Since the IGP metric and the IP addresses are used as tie breakers in the BGP decision process, changes to these cause instabilities to those prefixes using this AS as a transit AS. While IP address changes are expected to be rare, intra-domain traffic engineering is more widespread especially since the introduction of tools, such as Bravo [21]. Accordingly, a sizable number of prefixes may see AS path and/or next hop changes. Inter-domain traffic engineering [22] takes advantage of the full spectrum of options that BGP provides including but not limited to AS path prepending, filtering, local preference, prefix deaggregation, prefix aggregation, IGP metric changes, MED changes, EBGP session parameter changes. But since automatic tools are still a rarity, most of the tuning is still done by hand.

In summary, most instability events cause BGP updates to a number of prefixes at about the same time. But not all of these have to result in an AS path change. Other instability events are of concern to only individual prefixes. Note that human misconfigurations of BGP [23] are either unintended changes impacting single prefixes, IGP costs, or whole BGP sessions. Figure 5 tabulates the various possibilities.

3.2 Instability propagation

Next we consider the effects of routing instability in terms of how these BGP updates propagate through the Internet, and where they are observable. While some BGP updates change almost all attributes, quite a few only change a single attribute. We classify updates according to the attribute change with the biggest impact with regard to how far the update are propagated, see Figure 6.

Next we define an abstraction of the actual AS topology that we use in our arguments below. Each reasonably sized AS consists of a number of routers that have between them full IBGP connectivity, either via a full IBGP mesh, route reflectors, or confederations. Accordingly, we model each AS as a clique, one node for each router and an edge for each node pair. Each EBGP session between AS A and B corresponds to an edge between a node of the clique of AS

Instability/Condition	BGP updates	expected number of effected prefixes	responsible AS	comment
EBGP session availability				
single session	AS path changes	all relevant prefixes	both ASes	
multiple sessions	next hop changes	subset of rel. prefixes	both ASes	no alt. AS path of equal length
multiple sessions	AS path/next hop changes	subset of rel. prefixes	both ASes	alt. AS path of equal length
IBGP session availability				
reachability impacted	AS path changes	all relevant prefixes	AS	
reachability not impacted	next hop changes	subset of rel. prefixes	AS	
EBGP session filter	attribute changes	small subset of rel. prefixes	both ASes	
EBGP attribute changes	attribute changes	small subset of rel. prefixes	both ASes	
IGP cost change	next hop changes	subset of rel. prefixes	AS	tie breaker in BGP decision
	AS path changes	subset of rel. prefixes	AS	alt. AS path of equal length
IP address changes	next hop changes	subset of rel. prefixes	AS	tie breaker in BGP decision
	AS path changes	subset of rel. prefixes	AS	alt. AS path of equal length
link availability				
internal no session change	next hop/AS path change	subset of rel. prefixes	none	via IGP cost changes
internal with session change	AS path changes	all relevant prefixes	AS	via reachability problems
external no session change	none	none	none	unlikely
external with session change	next hop/AS path changes	all relevant prefixes	AS	via EBGP multiple session
node availability				= multiple "link availability"
originator changes				
single homed	new updates/withdraws	single prefix	originator AS	
multiple homed	attribute changes	single prefix	originator AS	

Figure 5: Effects of BGP instability

Class of updates	subclass	discussion
Local pref changes	multi-homed customer	might cause next hop change
	route selection between	might cause next hop change
	peer/upstream/customer	should cause AS path change
AS path changes	withdraw	the only route is no longer available corresponds to "bad news".
	new "better" route	corresponds to "good news" and can mean a new route with
		- a shorter AS path is available (ignoring local pref)
		- same length AS path is available (ignoring local pref)
		with new MED smaller and same next hop AS / new IGP cost or ID smaller if next hop AS changes
		- longer AS path length if local pref or weight is used
	implicit withdraw	corresponds to "bad news" and can mean the route with
	old route "better"	- a shorter AS path no longer available (ignoring local pref)
		- same length AS path is available (ignoring local pref)
		with new MED larger and same next hop AS / new IGP cost or ID larger if next hop AS changes
		- shorter AS path length if local pref or weight is used
Origin changes	IGP/Incomplete to EBGP	implies changes to the AS path, *i.e.,* current AS is no longer the originator
	EBGP to Incomplete/IGP	implies changes to the AS path, *i.e.,* current AS is now the originator
	Incomplete to IGP	change of status, likely together with next hop change
	IGP to Incomplete	change of status, likely together with next hop change
MED changes		MEDs are comparable for paths from the same AS
		MED changes may reorder paths from the same AS which may cause next hop changes
		MED changes may change ties between path from different ASes which may cause AS path changes
Next hop changes	without AS path changes	new route uses different EBGP or IBGP session between same peers
Community changes		need to be propagated since they are transitive. Agreement on semantic of global community values missing

Figure 6: Effects of BGP updates

A and a node of AS B. For simplicity and to ensure that AS internal effects are captured, we assume that each AS has enough nodes so that no two EBGP peering sessions are terminated at the same node. Now consider some prefix p and its routing table entries at all routers. The graph that is induced by choosing the edges of those sessions over which the router received the update and directing them towards the router is a directed acyclic graph (DAG), as long as there are no temporary loops induced by BGP. Any changes to the BGP sessions may impose changes to the DAG by adding or deleting edges or changing their direction, and all updates for this prefix p have to traverse a subset of this DAG. Hereby one has to keep in mind that each update can only traverse each edge in one direction and that each router will only propagate information about prefix p if its best route has changed.

Next we consider what this implies for our above classification of updates. Pure next hop changes matter for the current AS and may have to be propagated to neighbor ASes. But unless the router ID is used as a tie breaker in these ASes, the best path for the prefix will not change. This implies that these kinds of updates are highly localized. The same is true for MED changes and local preference changes, as long as the AS path is not changed and the IGP metric is not propagated via the MED values, see Figure 3(b). Since experience has shown that communities are not necessarily filtered, these updates have to be propagated throughout the subgraph of the DAG that is reachable from the instability creator. In the worst case, AS path changes and withdrawals have to be propagated along the same subgraph. But in most cases, due to the high connectivity of the Internet, other alternative paths exist. In this case, the update has to reach only those nodes that benefit from the new alternative path or those nodes that have to now choose an alternative path.

In summary, while one expects BGP updates to several prefixes if a change to an EBGP session is the instability originator, some or all of the updates may be rather local, *e.g.,* in case they involve only next hop changes. But they can also impose major non localizable BGP updates, *e.g.,* if AS path changes are involved. This may depend on the specific policy of the AS, the ISP's topology, *etc.* Changes to individual prefixes may have only local impact, *e.g.,* if no AS path change is involved, or a global one.

3.3 Observation points

Different places on the Internet have a different view of the connectivity throughout the Internet. At least in theory, a tier-1 ISP (provider) should be able to construct its own full routing table. A full routing table corresponds to knowing a BGP route for all prefixes that are not part of the AS in the sense of reachable via IGP. This information is gathered via customer provider relationships, where the provider learns routes to the customers prefixes while the customer can use the provider as default or receive a full routing table from the provider, and peering relationships to other providers, where each provider informs the other providers about its customer prefixes. Routes to prefixes learned from another peer are not propagated. Consequently the routing table that tier-n ($n > 1$) provider receives from its customers and peers is not a full one. They need a provider in order to supplement their table to a full routing table. To ensure reliability they usually use at least two different providers. Another consequence is that tier-1 providers more or less have to peer with each other to build the core of the Internet, while ISPs that do not provide transit services, and "simple" customers, *e.g.,* multi-homed ASes, are at the periphery. Often the AS graph is directed with the core AS at the top and the periphery at the bottom.

The policies outlined above capture only a small subset of all possible relationships between two ASes, but they seem to apply to a substantial part of the relationships [24, 25]. This structure has certain implications regarding connectivity. At the top, the connectivity is excellent – many alternative paths of the same AS-path length are available. Closer to the bottom, this diversity is significantly restricted. Furthermore, since customer ASes often have a primary connection to one AS and a backup connection to another AS, the connectivity is further reduced, as the backup path may not be visible to most of the Internet unless a failure close to the customer occurs. Accordingly, a monitoring point at an AS towards the bottom of the AS topology may not see any of the updates caused for example by a session reset between two tier-1 ISPs. Such a session reset may not change any of the best routes at this AS. In contrast, a monitoring point at a tier-1 ISP may not see any updates caused by a peering link failure between two of its customers. The redundancy requirement inherent in peering should guarantee this.

In summary, our initial question of how to locate the origin of an instability leads to related structural questions: How far does each class of BGP update spread? What is the impact radius of an instability, and how is it related to the position of the instability creator in the AS topology?

4. ADOPTED METHODOLOGY

Beyond opening new questions for the general evaluation of BGP, Section 3 motivates and imposes certain adaptations of the proposed basic methodology as well as introduces additional heuristics. The final approach is outlined in Figures 7 and 8. The first reflects the necessary adaptations while the second corresponds in essence to the ideal methodology (see Figure 2).

4.1 Candidate sets

Since instabilities can originate within an AS or between ASes our basic units are edges either between two ASes or within an AS. The **candidate set** of an AS path consists of an edge for each AS and an edge for each pair of consecutive entries on the path. Note that typically the path received at the monitoring point does not contain the AS in which the monitoring point resides, for brevity called the monitoring AS. However, should the path indeed contain the monitoring AS, then we can exclude that AS if care is taken to exclude all updates associated with session resets on the monitoring session. For example, the candidate set for the path 4321 where AS4 is the monitoring AS is: candidates(4321) $= \{(1,1), (1,2), (2,2), (2,3), (3,3)\}$.

```
## preprocessing per observation point
foreach prefix p              ## condense updates per prefix
    foreach observation point o
        U = updates(o) − flapping(o)
        burst_set_p = update_burst(U, timeout)
        foreach b in burst_set
            r_p = as_path(old_stable_route(b))
            r_n = as_path(new_stable_route(b))
            r_b = best_path_set(r_p, r_n)
            candidate_set c_ob = candidates(r_b)
## identify event set E
foreach time-unit t and foreach prefix p
    E_p = E_p ∪ new_event(t);
foreach event e ∈ E_p
    event_burst_set_e = associate_event_bursts(burst_set_p, e)
## condense bursts to identify instability origins
foreach event e and foreach prefix p
    foreach observation point o
        foreach (burst b, o) in event_burst_set_e
            candidate_set c_o = ∪c_ob
    foreach observation point o
        if candidate_set c_o == {}
            candidate_set s_o = stable_route(o, e)
    instability candidates = ∩c_o − ∪s_o
```

Figure 7: Per prefix – adapted meth. for locating instabilities.

```
## identify correlated events CE across prefixes
foreach time-unit t
    CE = CE ∪ new_correlated_event(t);
foreach correlated_event ce ∈ CE
    event_set_c e = associate_ce_events(ce)
## Greedy heuristic for clustering instabilities
foreach correlated event ce and event e ∈ event_set_c e
    P = ∪ prefix(e)
while (P != {})
    reset counts to 0
    foreach p ∈ P
        increase count(instability_candidates(event(p))
    i = instability with count(i) == max(counts)
    P = P−{p with i ∈ instability_candidates(event(p))}
    print instability i with prefixes Q
```

Figure 8: Across prefix – adapted methodology.

Best path: In general (see Section 3), it is hard to determine which route is the better one. For updates with path changes, this corresponds to the problem of deciding which of the two AS paths is the better path. Accordingly, the **standard** heuristic uses the conservative approach of including the union of the edges from both AS paths. This yields, ignoring induced updates, a lower bound with respect to pinpointing instability origins. For example, if monitor AS4 sees updates 4321 and 421, this results in best_path_set(421, 4321) $= \{421, 4321\}$. Should one of the two stable routes be a withdrawal then the best_path_set is the AS path of the other. If both stable routes are withdrawn then we do not gain any information and the best_path_set consequently contains all possible paths. If the AS path does not change then the best_path_set corresponds to one of the two paths.

If one ignores the impact of local preference and if the AS paths are of different length then the better path is the shorter one. Accordingly, the **best path** heuristic only considers the shorter one, *e.g.,* best_path_set(421, 4321) $= \{421\}$. Since we are lacking the information used by the BGP decision process when the AS paths have equal length, we use the standard heuristic.

Shared path segments: The example in Figure 3(b) shows that if the new and the old stable AS path have the same initial segments, it is in general not permissible to exclude it. Yet for an instability in the joined initial segment to cause the later AS path change, the instability has to be propagated through multiple ASes without path changes. This is possible via IGP/MED interactions or by specific filter combinations. Since experience shows that such combinations across multiple ASes are unlikely, one can expect that a heuristic, called **initial path**, which excludes the initial segments up to but not including the edges to the divergence point, does not do too badly. For example, with the initial path heuristic we have at AS7: best_path_set(754321, 764321) $= \{7543, 7643\} - 3$. This notation means that we can exclude the edge $(3,3)$ from the candidate set. This heuristic is especially helpful when only non-transitive attribute changes occur, *e.g.,* next hop, local preference, and originator changes. With regards to changes to transitive attributes, *e.g.,* communities, more care may be necessary.

The example in Figure 3 shows that, if the new and the old stable AS paths have the same final segments, it is in general not possible to exclude it. Therefore, applying a heuristic that excludes the final path segment, called **final path**, can be dangerous in the sense of excluding the cause of the instability. Nevertheless exploring it is of interest. For example with the final path heuristic we have at AS7: best_path_set(765321, 765421) $= \{65421, 65421\} - 6$.

Summary: Note that the standard heuristic tries to never exclude the cause of an instability or the induced instability. The heuristics: best, initial, and final paths can exclude some ASes and AS pairs that might have caused an instability. On the other hand they provide the benefit of narrowing the candidate sets. Accordingly, we are interested in evaluating the benefits and dangers of using the heuristics, especially since Chang et al. [4] by default assume that the initial and the final path heuristic are applicable.

4.2 Update bursts and stable path

The assumption that BGP converges within some limited time and that it is possible to identify the new best route is addressed next. While, *e.g.,* Griffin [2], Labovitz et al. [15], and Maennel and Feldmann [10], have shown that changes to the AS path can lead to BGP path exploration involving many BGP updates spread across a significant time period, it is possible to identify certain "stable" routes [10, 9]. Note that not all possible updates within the BGP path exploration process are indeed observable at all points [2].

Update bursts and stable route: To identify stable routes we use the notions introduced by Maennel and Feldmann [10]. We group updates observed at a given observation point and for a given prefix into a burst of updates, just as one would group packets into flows, using a timeout (referred to as the update_burst heuristic). We choose timeouts that are larger than typical min-route advertisement interval timer values, as well as typical propagation and processing delays. This allows us to group updates related to one event. Yet the timer is chosen smaller than typical delays due to route flap damping since we consider these as new instability events (Figure 4). The last update of each prefix burst approximates a stable route. After all, this AS uses this route as its best route for at least the length of the timeout. Given a burst, the **new_stable_route** is the last update in the burst. The valid route before the beginning of an update burst is the **old_stable_route** (or **previous_stable_route**). It is the new_stable_route of the previous burst at this observation point. The **stable_route** (without the modifier of "new" or "old" or "previous") is the route for the given prefix that was valid before the beginning of an update burst that is observed at some other observation point but for which no updates were received at the given observation point.

Experience [8, 10, 26] has shown that some prefixes will never have a stable route at certain observation points. This happens if a prefix is subject to continuous instabilities. Since this behavior is likely to violate the assumption that each prefix is only subjected to one instability at a time, we use a **flapping** heuristic to identify updates associated with such instabilities. This heuristic identifies all updates that are part of flapping bursts, an update burst determined using a timeout larger than the approximate maximum delay due to route flap damping and that persist for more than one day.

4.3 Events

So far we have adapted the ideal methodology of Figure 2 to include specifics about how to determine (old, new) stable routes, route and path changes as well as best paths calculations. But we are missing a way to identify instability events.

Note that all non-local instabilities of a prefix, *i.e.,* those that cause AS path changes, have to be propagated along a subgraph of the DAG of this prefix. This implies that an instability may be visible at multiple observation points within the DAG subgraph at about the same time. After excluding most effects due to path exploration, propagation delays, and BGP timers by computing per prefix update bursts, we now identify the start and end times of events and associate each burst with an event.

When identifying events we still have to deal with timing problems: within BGP, as well as with the propagation to the observation points, as well as with the "accurate" time synchronization of the monitors. To identify for each prefix the event start and end times together with the appropriate update bursts, those that started within this time window, the two heuristics **new_event** and **associate_event_burst** are used. One way of grouping bursts into events is similar to grouping updates into bursts or packets into flows using a relative timeout. The first event is initialized with the start time and the finish time of the earliest update burst across all observation points. If the time difference between the event finish time and the start time of the next update burst is less than the timeout value, the event end time is set to the end time of this next burst. This implies that the quiet period between the events has to be greater than the timeout value. This approach is referred to as **relative timeout**. A major drawback is that events can span a long time period and may therefore contain multiple actual events. An alternative is to use a **static timeout**. But now the sensitivity of the timeout becomes a problem. An appropriate value for one prefix may not be a good one for another. Furthermore, a static timeout may separate two related update burst into different events.

Accordingly, we also pursue the third option of an **adaptive timeout**. It consists of two steps: During an initial period which starts with the beginning of the event and ends at start + timeout, a relative timeout of timeout$/2$ value is used. Then a relative timeout of 0 is used. The first part desensitizes the specific choice of timeout values and takes advantage of the nice properties of relative timeouts. The second part together with excluding continuously flapping updates ensures that events are not too long and that parallel bursts are associated with the same events.

4.4 Correlated events

Section 3 clearly outlines that BGP updates to multiple prefixes are often correlated. Accordingly, the next step is to correlate the located instability origins across multiple prefixes and identify clusters. The **Greedy heuristic** outlined in Figure 8 provides a simple approach, but it requires us to identify which prefixes experience correlated events. We solve this problem by grouping events per edge to correlated events in the same manner using the same heuristics as for identifying events. But to prevent long events from attracting all other events we put a limit on how much each event can extend the per edge clusters. The number of events in each edge clusters is used in the ranking for the Greedy heuristic.

4.5 Related work

We use the same three dimensions for inferring the origin of routing instabilities as Caesar *et al.* [3] as well as Chang *et al.* [4]: *time, views,* and *prefixes*. Yet Caesar *et al.* first distinguish quiescent and turbulent periods while Heidemann *et al.* perform the per view and prefix steps in one clustering step. Lad *et al.* [5] use an idealized model assuming shortest path routing. Given this assumption, Lad *et al.* reasonably consider prefixes before views. However, if one relaxes this assumption and considers that non-shortest path routes do occur in BGP, then considering prefixes before views can lead to errors. We propose to always consider time first, then views, and finally prefixes. This way we can take the most advantage of our knowledge of BGP.

5. DATA SETS

Our work relies on external BGP routing tables dumps and update traces obtained from RIPE [18], Routeviews [19], a local ISP, and Akamai Technology. Throughout this paper we only present results in an exemplary fashion for the following raw data sets.

BGP update traces: from 12/04/03, 00:00 GMT to 12/16/03, 00:00 GMT, consisting of more than $343,600,000$ updates from more than $1,100$ different peering sessions to more than 650 ASes, including Tier 1 ISPs, major European ISPs, Asian ISPs, and stub ASes. Some ASes provide full feeds while others are partial feeds, with multiple sessions to about 43.3% of the monitored ASes.

Basic statistics: Overall the number of observed prefixes is $276,556$ of which $28,110$ are either from the private address space or contain only duplicate announcements. The latter are excluded from further consideration. Of the remaining, included prefixes, 42.7% were, at some point in time on at least one observation point, subject to AS path prepending, which is a popular policy used for traffic engineering purposes. In terms of inconsistencies we found that $4,038$ or 1.46% of the prefixes had multiple originating ASes. Also, $11,507$ of the pairs of (observation points, prefixes) were continuously receiving updates in the sense that they have at no inter-update time larger than 2 hours for more than 1 day.

Inferred AS topology: We took one day of BGP table and update data on December 10, 2003 for the purpose of analyzing AS relationships and inferring AS paths for simulation purposes as described below. $3,428,464$ distinct AS paths after ignoring AS prepending are used as input to the relationship inference algorithm. The graph consists of $16,757$ nodes with $45,376$ edges. Based on the relationship inference, we have $30,653$ customer-provider relationships, and $1,532$ pairs of ASes are found to have peering relationships. 97.9% of the input AS paths are found to be valid policy paths, *i.e.,* conforming to the inferred relationships.

6. WHAT IF – SIMULATIONS

To understand the accuracy of our algorithm for inferring the location and the cause of routing instability, we validate via extensive simulations on the inferred AS topology. We make use of RouteScope [27] in inferring all valid policy paths between two ASes. RouteScope uses a simple algorithm based on shortest AS hop count for inferring AS paths between two end systems, without access to either host, by using information from BGP tables collected from multiple vantage points. Given the collection of AS paths from BGP tables, the AS relationship inference algorithm by Battista *et al.* [28] is used to identify all valid policy paths. Valid AS paths are assumed to go through paths in the form of *CustomerProvider* PeerPeer? ProviderCustomer** (denoted as *AS path rule*), where "*" represents zero or more occurrences of an AS edge and "?" represents zero or a single occurrence of an AS edge.

Based on the inferred AS relationships, edges in the AS graph are grouped into the following four categories: (i) custom-provider link (UP link), (ii) provider-custom link (DOWN link), (iii) peering links (FLAT link), and (iv) unknown AS relationship. For the last type, we replace the edge with one UP link and one DOWN link, effectively removing any restriction on the inclusion of edges with unknown AS relationships. We repeated our analysis with all such edges excluded from the AS graph. The results are very similar to what we report here. The accuracy of RouteScope in predicting AS paths from several selected ASes to the entire Internet is around 85%. Inaccuracy stems from the following reasons: (1) Inaccuracy in AS relationship inference. (2) AS prepending effect is ignored. (3) Special routing policies for particular prefixes. We emphasize that such inaccuracy does not affect our evaluation methodology, as we aim to have a reasonably accurate AS topology to study whether our algorithm can precisely identify the location of simulated failures, given the AS paths selected before and after the failure.

6.1 Controlled experiments

We perform the following set of controlled experiments. RouteScope can infer a set of most preferred valid policy paths between any two AS pairs in the AS graph. Oftentimes, multiple AS paths appear to have the same preference, *i.e.,* with the same AS path length and of the same type (customer, peer, or provider routes). To understand the effect of an arbitrary link failure, we randomly select a set of observation points and destination points by picking from tier-1 ISPs, tier-2 ISPs, ISPs with other ranks (based on ranking algorithm in [29]), and stub ASes based on a fixed proportion. We also attempt to include the observation points from which we have the BGP feeds as part of the source ASes.

Given the selection of source and destination AS nodes, we study the effect of a failure by computing the set of best AS paths before and after the failure. We remove the inter-AS link affected by the failure. In practice, there may be multiple peering links between two ASes, especially two large providers. We simplify this by assuming a single link and thus simulate the worst case scenario. We select 100 failures strategically by considering a variety of combinations of ASes in different parts of the Internet hierarchy, *e.g.,* between two tier-1 ASes, a tier-1 AS and a stub AS, two tier-2 ASes, *etc.* Given the set of equal cost paths between two ASes, we impose a selection among all equal cost paths to make sure the routing decision is consistent. For instance, if AS X selects a route advertised by AS Z, and AS Y chooses a route from AS X, Y's route must also go through Z.

To our surprise, just randomly selecting 100 destinations and observation points can make it hard to observe any changes in the best paths. Contrary to previous claims, in our simulations, we found that BGP failures are fairly well-isolated due to redundant paths, see Section 3.2. We plan to explore this further using the

RouteScope by understanding the properties of topologies where a particular failure between two ISPs of given ranks can affect.

6.2 Results

Given the results of the failure, we apply our heuristics (see Section 4) to the data sets to compute possible instability candidate sets. We note that none of the heuristics has ever excluded the failed edge from the resulting instability set indicating that the approach is sound. Indeed it is not surprising that none of the examples from Section 2.2 materialize since the simulation scenario mainly follows the assumptions except for best path. Preferring customer and peering relationships over upstream providers may cause the best path heuristic to fail.

To explore whether the failure location has any impact on our ability to locate it, we divide the failed links into three classes: "top tier" (between tier-1's and tier-2's or between a tier-1 and a tier-3), "middle tier" (between tier-2's, tier-3's or tier-4's), "bottom tier" (all others). Unfortunately the observation that random selection of points makes it hard to observe any changes also implies that the number of observation points that observe any best path changes is limited. This is extreme for the bottom class where of the $9,112$ events, about 15% are observed at multiple observation points. For the middle tier, this value is close to 40% and for the top tier 69%. While this might limit the potential gains of our methodology, Figure 9 shows that it is possible to derive instability set sizes with the conservative standard heuristic of $5-7$ for more than 68% of the failures with only two observation points. With 10 observation points this increases to almost 88% and almost 100% with further observation points. This implies that it is easier to track those failures that are percolating through larger parts of the Internet.

Narrowing the instability set size to three is the best one could hope for since the candidate is at a peering link which causes the instability set to already contain three edges. This set can only be further reduced by using other heuristics. Figure 10 shows this benefit for the bottom class. The best path heuristic helps decrease the instability set size from two inter-AS links (= 5 AS-AS edges) to one (= 3 AS-AS edges) for more than 20% of the events. The stable, the initial, and final heuristics exclude more edges so that even the precise instability origin can be determined. Overall the combination of all heuristics reduces the instability set to less than 5 AS-AS edges including intra-AS edges for more than 88% of the cases. Figure 11 shows that using our heuristics with five observation points, one can pinpoint the origin for more than 80% of the events down to less than 5 edges for bottom tier, 6 edges for middle tier and 7 edges for the top tier. Overall this indicates that the huge redundancy within the Internet core adds additional complexity to pinpointing the origins. On the positive side these edges are used by many different prefixes so that the Greedy heuristic will capture the appropriate edges, which it indeed does for these failures.

7. WHAT IS – DATA ANALYSIS

Having shown that our methodology is sound in the "what if" world, we can now apply it to the "real world". For this we need to determine how sensitive the results are to the heuristics as well as their parameters. Furthermore each step provides us with useful information about how far BGP updates spread and their impact radius. For the purpose of the evaluation, we partition the approach into the following four stages: (1) update burst calculation, (2) grouping of bursts to events (3) instability candidate calculation for each event, and (4) correlation of events.

7.1 Update bursts

The update burst calculation is used to identify prior and post stable routes and one of the more interesting questions is how do these stable routes differ. We classify the changes into the following groups: *path* summarizes all updates that have changes in their AS path, *reachability* counts those prefix bursts where a route either became available or was withdrawn, *community* sums up those without path and reachability changes but with a change in their community attribute, *nexthop/med* includes those with either nexthop or med but no AS path or community change, *other* captures all other attribute changes not within any of the previous classes, while *none* pools those where all attribute values of the two stable routes remain equal. Note that if, *e.g.,* the AS path and the MED changes then the burst is counted as a path change.

Figure 12 shows a histogram of the number of bursts per group for different timeout choices. Notice that while more than 23% of the 2 minute bursts include a path change, more than 24% do not result in a change, even though within the burst there was a change. Furthermore, a significant fraction only propagates attribute changes that should mainly have local impact. On the other hand, about 5% of the bursts with pure community changes have to be propagated through the full reachability graph unless some providers filter such community values.

We decided to study timeouts ranging from 2 to 16 minutes. The smallest value, 2 minutes, can be sufficient to group updates caused by path exploration together into one burst. For example, Mao et al. [6] have shown that most beacon announcements converge within a two-minute window. Yet since there are various ways in which BGP updates can be delayed, including MRAI timer, route reflectors, *etc.*, not all path exploration can be captured with timeout values of 2 minutes. Accordingly, we study larger timeouts of 4, 8 and, 16 minutes as well. The problem with large timeouts is that they can group the results of several instabilities together, *e.g.,* an instability together with the instability repair, which may correspond to combining bursts with smaller timeouts of group path or group reachability to one of group none. The decrease in the absolute number of bursts as well as the above average decreases in the path and reachability groups is apparent in Figure 12. The larger timeout values are especially problematic since they are larger than the delays imposed by route flap damping. Furthermore, they limit our ability to pinpoint the exact time of the instability which is needed for the next steps for determining the origin of the instability.

In terms of the duration of the instabilities, Figure 13 shows a histogram of the number of bursts per group for a timeout choice of 2 minutes, by duration of the burst. That a large fraction of the bursts lasts less than 64 seconds, independent of the timeout value, is an indication that most timeout values are reasonable. On the other hand we note that the longer a burst lasts the more likely it is to recover its old stable route. This is (not shown) even more dominant for larger timeout values. The bursts in the community, nexthop/med, and other groups tend to be significantly shorter than others. This may be an indication that these bursts are not the result of a path exploration. As the duration of a burst is somewhat correlated with the number of updates in a burst, it can be expected that most bursts contain a fairly small number of updates. Figure 14 shows the result of first grouping bursts according to the number of updates that they contain and then computing the relative distribution across the groups. Most of these involve a change in either reachability or in the AS path. The fact that most bursts in the

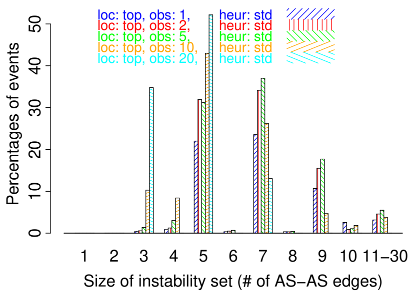


Figure 9: Sim: instability set size hist. for # of obs. (heur.: standard, loc.: top).

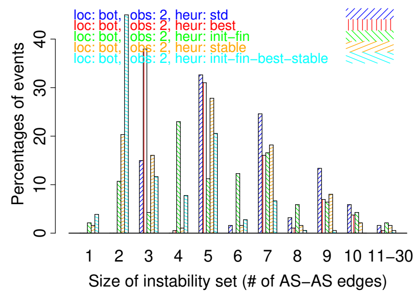


Figure 10: Sim: instability set size hist. for various heuristics (obs: 2).

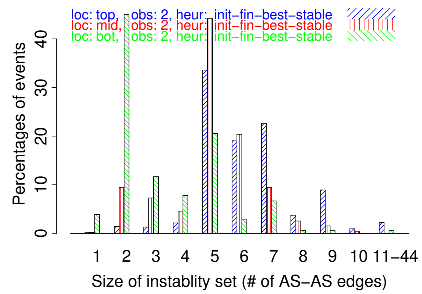


Figure 11: Sim: instability set size hist. for failure locations (heur.: all, obs.: 2).

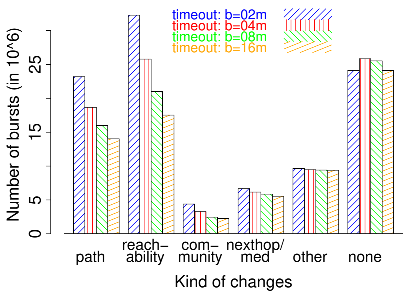


Figure 12: Stable route differences for various timeouts.

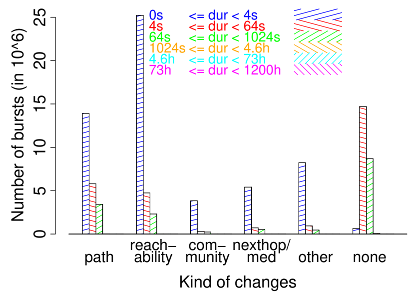


Figure 13: Stable route differences for burst length with 2 minute timeouts.

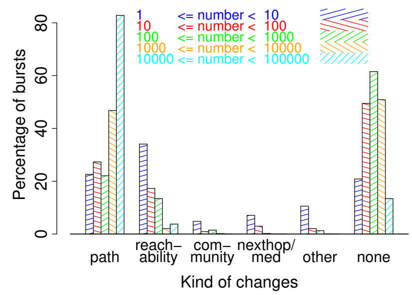


Figure 14: Stable route differences by # of updates in burst for 2 minute timeouts.

community, nexthop/med, and other group are dominated by small bursts is another indication that no extensive path exploration takes place. Yet, longer timeout values cause the numbers of updates within bursts to increase and these are likely to be in group none. Since we want to pinpoint instabilities in time and since bursts of group none have lost most of their information about the location of the instability, we proceed with update bursts of 2 and 4 minutes.

7.2 Events

This stage associates bursts from various observation points with events using various timeout heuristics: relative, static, and adaptive timeout. With regard to choosing parameters, the event associated timeout should be compatible with the update burst timeout. Choosing an event timeout less than the burst timeout indicates that one is more stringent for grouping events than for updates and can create situations where a burst should be part of two events. To avoid this, we choose to be more lenient and use a timeout greater than or equal to the burst timeout. Still the timeouts should not be too large to avoid grouping those bursts together that were separated by the timeout of the burst calculation. Accordingly we select the following parameter sets for relative: (bursts=2m,events=4m), (b=4m,e=8m); static: (b=2m,e=8m), (b=4m,e=16m); adaptive: (b=2m, e:(max=16m,rel=4m), (b=4m, e:(max=16m,rel=4m)).

Overall we notice that the specific choice of parameters and heuristic does not appear to cause major differences. For example, Figure 15 shows a grouping of the events into similar categories. An event belongs to group "path" if at least one of its bursts belongs to this group. An event belongs to group "reachability" if no burst belongs to group "path" and at least one burst belongs to group "reachability," *etc.* Some observations about bursts carry over to events, *e.g.,* events are again dominated by path and reachability changes. But while more than 24% (28%) of the 2 (4) minute bursts are in group none, less than 14% (18%) of the events are. In comparison, note that the fractions of events with community, next-hop, or other attribute change have increased significantly.

One of the reasons for proposing to use the static and adaptive timeouts is that we want to limit the duration of each event in order to pinpoint the origin of instability, either in the next step or the final step of event correlation. This is indeed the case for these heuristics. Figures 16 and 17 show histograms of the event durations for a subset of the parameter choices. Note that most events, just as most bursts, are short, yet a few last for a long time. While the events identified by the relative heuristic can be long (more than 4 hours) the ones generated by the static heuristic are indeed less than 16 minutes. The adaptive timeout events are somewhere in between since they separate active from quiet periods and are patient enough for active periods to finish.

The next question motivated by our experiences with the simulated failures is how many observation points observe each change. More than 66.6% of instabilities are only observable at a single observation point and 16.1% at two. Overall 95.3% are observable at less than 10 observation points. This again confirms that BGP indeed provides significant isolation against routing updates.

7.3 Instability candidates

Once the bursts are grouped to events, we can apply path heuristics to compute the instability candidates for each event to compare with simulation results.

7.3.1 *Beacons*

We first apply our heuristics to BGP beacon prefixes [6]. A BGP beacon is an unused prefix which has a well-defined schedule for announcement and withdrawal; thus, we know precisely the origin of instability.

Our experiences with the simulation results have shown that we

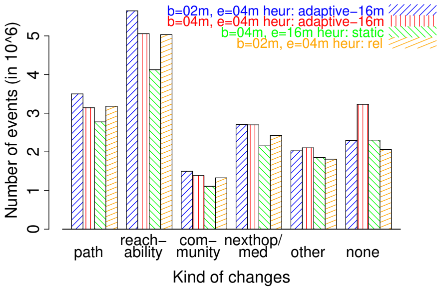


Figure 15: Event characterization for various timeout heuristics.

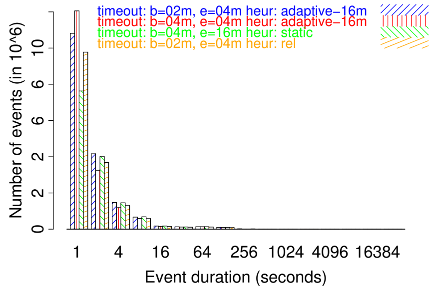


Figure 16: Event duration for various timeout heuristics .

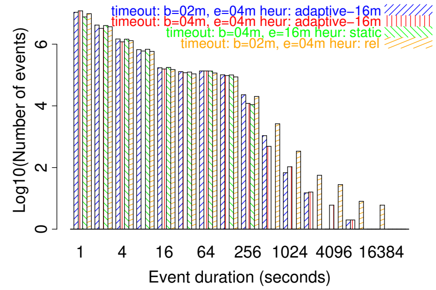


Figure 17: Event duration (log y-scale) for various timeout heuristics .

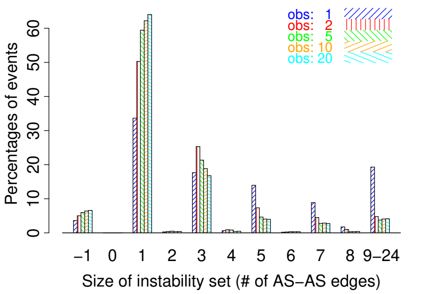


Figure 18: Beacons: instability set size hist. for # of obs. (adaptive (b=2m,e:(max=16m, rel=4m)); heur.: standard).

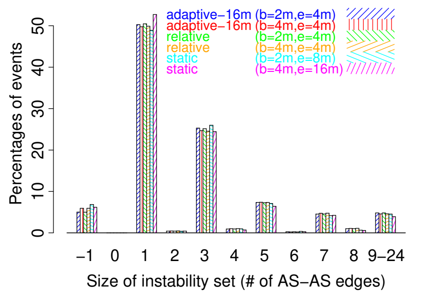


Figure 19: Beacons: instability set sizes hist. for timeout heuristics (obs.: 2; heur.: standard).

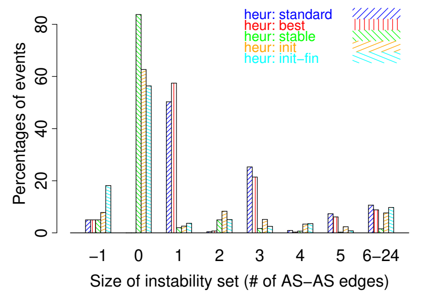


Figure 20: Beacons: instability set size hist. for heuristics (adaptive: (b=2m,e:(max=16m,rel=4m)); obs: 2).

can usually narrow the candidate sets to an instability candidate set of about $3-7$ edges for more than 70% of the examples. For the beacons we can do even better. With the adaptive event timeout (b=4m,e:(max=16m,rel=4m)) heuristic and at least two observation points we can narrow the instability set to three or fewer AS edges for more than 76% of the events (see Figure 18). We verified that each instability set contains the edge for the origin AS. While in theory each beacon should be observable at all observation points, this is not the case. For example, some events consists of only withdrawals and the previous event is also a withdrawal. These kinds of events are captured in the category labeled "-1" and sum to about 5%. Furthermore, with the adaptive event timeout, more than 31% of the events are only observable at a single location. This can be explained by BGP update delays, *e.g.,* due to route flap damping, filtering at intermediate peers, time synchronization problems of the collectors, or other instabilities that affect beacon prefixes. Using the static (b=4m, e=16m) heuristic reduces this to only 18% since it ensures that appropriate events are clustered while other unrelated events are separated from the beacon events. Thus the percentage of events with an instability set of three or fewer AS edges increases to 77.6%. Note that identifying three AS edges usually includes the edges of two ASes and the edge between the two ASes. For more than 50% of the beacon events both heuristics let us identify the origin AS correctly as the instability creator. Increasing the number of observation points to at least two increases this to more than 64%, a rather nice success rate. Indeed if one considers a set of four ASes good enough for pinpointing the instability, we succeed for more than 90% of the cases with only two observation points. Figure 18 highlights again the benefit of having information at multiple observation points. Figure 19 accentuates that while the timing heuristics differ, *e.g.,* in terms of the number of events that are only observed at a single observation point, they still generate results of a similar accuracy level in terms of the instability sets they identify.

Figure 20 shows the histogram of the sizes of the instability sets for the different path heuristics. The conservative approach of the "standard" heuristic is rather successful, and the "best path" heuristics improves the accuracy by more than 7%. But the "stable", "initial", and "final path" heuristics prove to be disastrous. The intersection size is empty in more than 55% and for stable up to 83% of the cases. This indicates that the examples shown in Section 2 are not just possible but that BGP features that create similar results are in use in the Internet. Note that the results are similar for different choices of timeout heuristics.

7.3.2 All prefixes

Next we apply our path heuristics to all prefixes and consider the same set of plots as for the BGP beacons (Figures 21, 22, 23). Clearly the results are not quite as good as for the BGP beacons. On the other hand being able to pinpoint the origin of an instability, which is observed at two observation points, to three AS edges for more than 42% of the cases, and to five AS edges for about 70% for all timeout heuristics, and more than 76% for some, is quite impressive and shows that we have made significant progress towards understanding the origin of BGP instabilities. We checked that almost all of the time, 99.9%, instability sets with three AS edges correspond to those that surround a BGP peering location, *e.g.,* AS1 and AS2 are peers and the instability set contains (AS1-AS1, AS1-AS2, AS2-AS2). Furthermore, most of the time the AS edges in the instability set are continuous on some AS path.

Furthermore, with increasing number of observation points, the ability to pinpoint increases (see Figure 21). But most important,

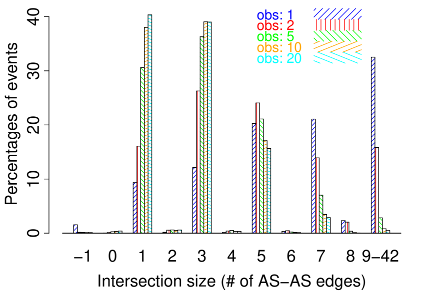


Figure 21: Instability set size hist. for various # of obs. (adaptive: (b=2m, e:(max=16m,rel=4m)); heur.: standard).

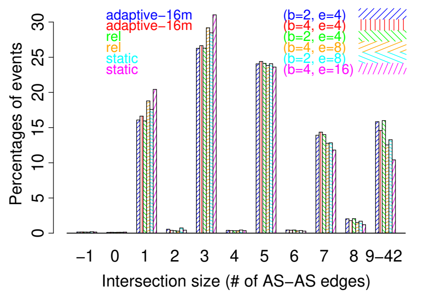


Figure 22: Instability set size hist. for timeout heuristics (obs.: 2; heur.: standard).

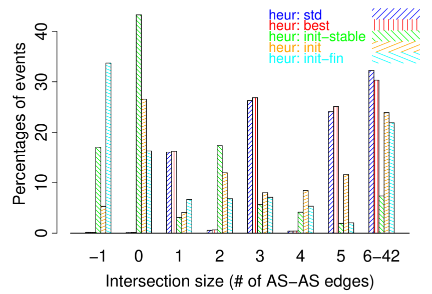


Figure 23: Instability set size hist. for various heuristics (adaptive: (b=2m, e:(max=16m,rel=4m)); obs.: 2).

the instability set is hardly ever reduced to size 0, which confirms that using the standard methodology is a safe approach for reducing the candidate set size. Indeed with more than 5 observation points it is possible to reduce the size to five edges for almost 90% of the events for all timeout heuristics without increasing the fraction with zero size instability sets.

Comparing the various timeout heuristics, Figure 22 shows that the impact of the specific methods increases but is not that dramatic with the exception of the static heuristic based on 4 minute bursts. This indicates that timeout values of 2 to 4 minutes used in the other heuristics may not yet be optimally chosen.

The impact of the path heuristics (Figure 23) is on the one hand positive, *e.g.,* for "best" path, but on the other hand again disappointing for "initial," "final," and "stable." The results for "stable," however, are not quite as bad as for the BGP beacons. In a significant number of cases, the "stable" heuristic helps to exclude quite a number of suitable AS edges, *e.g.,* for those events that with "stable" have an instability set of two AS edges. But it appears that the heuristic needs to be fine tuned to require multiple observation points to classify a given edge as stable before declaring it as such. Overall the results are rather promising.

Regarding the origin of instabilities, we further inspect the instability sets with up to and including three edges for the adaptive heuristic (b=2m,e:(max=16m,rel=4m) including all observation points. For these, 30.4% of the time the origin AS is the only AS in the instability set. For 66.3%, the origin AS is one of the edges. This leaves us with 3% that are unconnected to the origin AS. Of these, .4% included only a single AS, while 1.3% include multiple ASes. The others 1.3% include some inter AS link.

7.4 Event correlation across prefixes

So far we have seen that it is possible for most events that involve more than a single observation point to identify a reasonable-sized instability candidate set. To locate a plausible origin for the remaining events we can take advantage of our knowledge of BGP instabilities, see Figure 5. Most of the plausible events cause updates to multiple prefixes at about the same time. Given that the instability set computation has already narrowed the origin of the instabilities, majority decisions help us now to further pinpoint the instability origins using the Greedy heuristic, see Section 4.4.

Using rather aggressive timeouts of 4 minutes to determine which edges are considered correlated for each event, and artificially shortening the duration of each event to a maximum of 16 minutes help us to ensure that we mainly catch correlated events. In each time period, selecting the edge that is involved in the most instabilities is easy as the number of events differ by a rather significant factor, usually larger than 1.5. We have even observed factors of up to 3. Indeed the distribution of correlated events by edge appears to be consistent with a Zipf distribution, which justifies using the Greedy heuristic. This also explains why the Greedy heuristic is rather successful in identifying instabilities. Once the most likely candidates have been identified the Greedy heuristic may have to break ties. We choose to not break ties but rather include all edges as possible instability origins.

After applying the Greedy heuristic, with the adaptive timeout heuristic, we are able to associate 93.4% of the prefixes to a single AS as a possible origin for the instability. A single AS corresponds to the intra-AS edge. In more than 97.2% of the cases Greedy narrows the possible instability origin to at most three AS edges. Even if the instability origin includes three AS edges, the instability origin points to a single peering connection in more than 47.5%. As the edges correspond to the two intra-AS edges and the link between them. In the other 52.5%, the instability origin seems to lie inside any of the three involved ASes. If we require that each correlated event has at least 100 prefixes, we are able to associate 96.3% of the prefixes with a single instability origin.

7.5 Validation

Given that Greedy appears to be able to further pinpoint the instability origin, the obvious question is if the caveats and dangers highlighted by the examples in Section 2.2 have been sufficiently addressed or if the heuristic misguided us. We address this issue by further validating our results in two ways. First we use syslog data from a large tier-1 ISP to see if we can correlate times when Greedy identifies the AS edge *A-A* (A is the AS number of the tier-1 ISP) as instability origin with session resets within the tier-1 ISP. Second we again take advantage of our simulator by comparing the inferred origin of instability with simulated results. For this purpose we select some instabilities identified by Greedy to simulate and then compare the outcome.

For the first validation step we selected 35 events for which Greedy identified *A-A* as instability origin and the syslog data is available. We checked them against the appropriate router syslog data from the ISP. (Note that the syslog data is not available for the entire week.) We say that we find a session reset that corresponds to the event if at most the 5 minutes of the instability event overlaps with the session reset time window – starting 1 minute before the time session reset occurs shown in the syslog and ending 1 minute after that. Adding 1 minute to the time window addresses potential clock synchronization issue and the delay in observing the change at the BGP monitoring points due to BGP rate limiting timers and propagation delays. The results are rather promising as we can find related session resets for 26 or 74% of them. One should not expect to find all events since some may not have been caused by session resets. Furthermore, note that not all session resets will cause updates as discussed in Section 3.

For the second validation step we selected 20 events for which Greedy identified a single AS-AS edge as the instability origin. The corresponding simulation first excludes those prefix origins that do not use this AS-AS edge on one of their best path under the simplified simulator BGP model. Note that otherwise this will introduce some errors due to inaccurate policy inference. It then fails the appropriate edge, or approximates the failure inside an AS by deleting all edges containing the corresponding AS. This effectively assumes that all paths going through the AS are affected. We find that the heuristics when applied to the outcome of these simulations identify the same AS-AS edge in 90% of the cases improving our confidence in the appropriateness of the heuristics.

8. SUMMARY

Trying to identify the origin of global Internet routing instabilities poses challenges that stem from the complexity of the BGP decision process, the challenging problem of achieving a globally optimal routing via local routing configuration by various administrators as well as the global traffic dynamics. In this paper we propose a methodology for identifying the origin of routing instabilities by examining and correlating BGP updates along three dimensions: *time, views,* and *prefixes* and show how it can be adapted to account for the complexities of BGP. By applying our heuristics first to an ideal world where we control the failure and the observation points and then to a huge amount of actual BGP updates, we show that the methodology is sound and accounts for cases that have been previously ignored. Indeed with only two observation points, we are able to pinpoint the origin of instabilities due to beacons to no more than three AS edges for more than 76% of the cases. This increases to only five AS edges when considering all prefixes. Relying on Zipf like characteristics of correlated events across prefixes, the Greedy heuristic is capable to further pinpoint the origin to a single AS for more than 93% of the prefixes. Accordingly we conclude that despite the intricacy of ISP routing policies, and the issues regarding propagation, or lack thereof, of BGP update messages, and complexity of the Internet topology, we have demonstrated significant ability at narrowing down the location of BGP instabilities.

Clearly, the work reported herein has not exhausted the problem area, and there is much more that could be done. In future work we plan to explore the accuracy of our methodology by simulating failures inferred from the data analysis and checking the resulting updates for consistency. Furthermore, we plan to explore how well BGP indeed isolates routing instabilities and how this is correlated with the location in the topology of the instability as well as the observation points. In addition, we plan to take advantage of our new capability to explore the causes of BGP instabilities.

Acknowledgments

We thank support from Akamai staff, especially Sean Butler and Andy Champagne for helping us access and understand the data used in this work. We are grateful to Ramesh Govindan, our shepherd, as well as the anonymous reviewers for valuable suggestions regarding the material itself as well as on how to improve its presentation.

9. REFERENCES

[1] J. W. Stewart, *BGP4: Inter-Domain Routing in the Internet*. 1999.

[2] T. Griffin, “What is the Sound of One Route Flapping?,” 2002. IPAM.

[3] M. Caesar, L. Subramanian, and R. H. Katz, “Route cause analysis of Internet routing dynamics,” tech. rep., UCB/CSD-04-1302, 2003.

[4] D.-F. Chang, R. Govindan, and J. Heidemann, “The temporal and toplogical characteristics of BGP path changes,” in *Proc. ICNP*, 2003.

[5] M. Lad, A. Nanavati, D. Massey, and L. Zhang, “An algorithmic approach to identifying link failures,” in *Proc. PRDC*, 2004.

[6] Z. M. Mao, R. Bush, T. G. Griffin, and M. Roughan, “BGP Beacons,” in *Proc. ACM IMC*, 2003.

[7] N. Feamster, D. G. Andersen, H. Balakrishnan, and M. F. Kaashoek, “Measuring the effects of internet path faults on reactive routing,” in *Proc. ACM SIGMETRICS*, 2003.

[8] J. Rexford, J. Wang, Z. Xiao, and Y. Zhang, “BGP routing stability of popular destinations,” in *Proc. ACM IMW*, 2002.

[9] Z. M. Mao, R. Govindan, G. Varghese, and R. Katz, “Route flap damping exacerbates Internet routing convergence,” in *Proc. ACM SIGCOMM*, 2002.

[10] O. Maennel and A. Feldmann, “Realistic BGP traffic for test labs,” in *Proc. ACM SIGCOMM*, 2002.

[11] R. Teixeira and J. Rexford, “A measurement framework for pin-pointing routing changes,” in *Proc. ACM SIGCOMM Network Troubleshooting Workshop*, 2004.

[12] T. G. Griffin and B. J. Premore, “An experimental analysis of BGP convergence time,” in *Proc. ICNP*, 2001.

[13] C. Labovitz, R. Wattenhofer, S. Venkatachary, and A. Ahuja, “The impact of Internet policy and topology on delayed routing convergence,” in *Proc. IEEE INFOCOM*, 2001.

[14] H. Tangmunarunkit, R. Govindan, S. Shenker, and D. Estrin, “The impact of Internet policy on Internet paths,” in *Proc. IEEE INFOCOM*, 2001.

[15] C. Labovitz, A. Ahuja, A. Abose, and F. Jahanian, “An experimental study of delayed Internet routing convergence,” in *Proc. ACM SIGCOMM*, 2000.

[16] C. Labovitz, R. Malan, and F. Jahanian, “Origins of Internet routing instability,” in *Proc. IEEE INFOCOM*, 1999.

[17] R. Govindan and A. Reddy, “An analysis of Internet inter-domain topology and route stability,” in *Proc. IEEE INFOCOM*, 1997.

[18] RIPE’s Routing Information Service. http://data.ris.ripe.net/.

[19] University of Oregon RouteViews project. http://www.routeviews.org/.

[20] L. Subramanian, V. N. Padmanabhan, and R. H. Katz, “Geographic properties of Internet routing,” in *Proc. Usenix*, 2002.

[21] A. Feldmann, A. Greenberg, C. Lund, N. Reingold, and J. Rexford, “NetScope: Traffic engineering for IP networks,” *IEEE Network Magazine*, 2000.

[22] N. Feamster, J. Borkenhagen, and J. Rexford, “Guidelines for interdomain traffic engineering,” *ACM CCR*, 2003.

[23] D. Wetherall, R. Mahajan, and T. Anderson, “Understanding BGP misconfigurations,” in *Proc. ACM SIGCOMM*, 2002.

[24] L. Gao, “On inferring autonomous system relationships in the Internet,” in *Proc. IEEE Global Internet*, 2000.

[25] B. Norton, “The art of peering: The peering playbook,” 2002.

[26] K. Varadhan, R. Govindan, and D. Estrin, “Persistent route oscillations in inter-domain routing,” tech. rep., USC/ISI-96-631, 1996.

[27] Z. Mao, L. Qiu, J. Wang, and Y. Zhang, “Inferring AS-level paths with RouteScope,” Tech. Rep. TD-5T3RRP, AT&T Labs – Research, 2003.

[28] G. Battista, M. Patrignani, and M. Pizzonia, “Computing the Types of the Relationships Between Autonomous Systems,” in *Proc. IEEE INFOCOM*, March 2003.

[29] L. Subramanian, S. Agarwal, J. Rexford, and R. H. Katz, “Characterizing the Internet hierarchy from multiple vantage points,” in *Proc. IEEE INFOCOM*, 2002.