Fixing BGP, One AS at a Time*

Jaideep Chandrashekar Zhi-Li Zhang Hal Peterson

Department of Computer Science & Engineering
University of Minnesota, Minneapolis
Minneapolis, MN 55455
{jaideepc, zhzhang, peterson}@cs.umn.edu

ABSTRACT

Debugging inter-domain routing problems on the Internet is notoriously hard. This is partly because BGP updates carry no information about the events that trigger them, and also because operation is highly distributed and complex, lacking a central point of control or authority. These factors have impeded the development of tools that can help in the diagnosis and troubleshooting of routing problems. Consequently, the dynamic behavior of BGP is not well understood, even though it forms a *critical* component of the Internet infrastructure.

In this paper, we argue that these problems can be effectively addressed by incorporating a *diagnostic* capability into the inter-domain routing system. Such a capability would enable operators to *effectively* pinpoint the cause (and location) of routing problems and take *timely* steps to correct them. We present arguments to show that this capability is best realized by *localized*, AS-specific entities called *BGP Auditors*. We then explore the high level design of a *distributed*, diagnostic framework and briefly discuss the relevant issues.

Categories and Subject Descriptors

C.2.2 [**Network Protocols**]: Routing Protocols; C.2.3 [**Computer-Communication Networks**]: Network Operations—*Network Management, Network Monitoring*

General Terms

Design, Measurement, Management

1. INTRODUCTION

Debugging routing problems on the Internet is exceptionally hard. BGP is an incremental protocol and updates are exchanged *only* when there is a change in the network, such as caused by link or router failure, policy change, misconfiguration, etc. It is extremely difficult to infer the original network event by observing the routing updates resulting from it. This has led to the situation where, despite being the most critical component of the Internet infrastructure, *few people understand the dynamics of inter-domain routing.* When routing problems do occur, it is very hard to pinpoint, and subsequently correct, the root cause. There are several reasons for this: (1) route selection in BGP is a complex, *distributed* computation guided by *locally* configured policies; (2) though the design of Internet routing explicitly separates *inter* and *intra* domain routing for scalability, in reality there is a certain degree of interaction (i.e., hot-potato routing) which introduces additional complexity into BGP, and (3) BGP updates contain very little information that can be used to reason about their root cause(s) [9].

All these factors have inhibited the development of tools to aid operators in troubleshooting routing problems. Currently, when a routing problem is detected, operators resort to extremely crude methods: *ping* or *traceroute* probes, mailing lists, and even phone calls to centrally located, "clueful" ISPs.

On the whole, the Internet performs surprisingly well, at least most of the time. At the same time it is also inherently unreliable; for example, if enough of the "right" people are away from their telephones, then an outage may go unchecked for hours or longer (see [14, 11]). This state of affairs has serious implications for the Internet and society at large that increasingly relies on it. Therefore, it is imperative that we take immediate measures to rectify it.

There are (broadly) two philosophical directions one might take to address this state of affairs: on one hand, we can pursue *preventive* measures by adopting routing configuration verifiers, improving policy configuration languages and so on, with the emphasis being on checking errors and detecting *potentially* anomalous behavior before it actually occurs. The other direction is *reactive* in nature.

In general, *preventive measures* do not address the problem effectively (and soon enough). Another serious drawback is the lack of *individual* incentive to deploy them. In other words, while these preventive measures may demonstrate a *collective*, theoretical benefit to the entire Internet community, there is little *personal* incentive for network operators to deploy them in their own networks. Most importantly, they offer little protection against malicious behavior, which is an increasingly important concern [15].

On the other hand, *reactive* measures emphasize building tools and mechanisms to aid in locating and correcting routing anomalies when they do occur. The underlying motivation is that in a system as large as the Internet, mistakes are inevitable and it is much harder to build safeguards that guarantee the opposite.

While it is important to prevent anomalous events, it is as impor-

*This work was supported in part by the National Science Foundation under the grants ANI-0073819, ITR-0085824, and CAREER Award NCR-9734428. Any opinions, findings, and conclusions or recommendations expressed in this paper are those of the authors and do not necessarily reflect the views of the National Science Foundation

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

tant (if not more) to build the *ability to reason about, and correct them, when they do occur* [4]. In this paper, we argue that a *reactive* capability is required in BGP and present the outline of a *supporting* framework enabled by so-called AS-specific *BGP Auditors*. Specifically, the *BGP Auditor* framework— combining *inference* and *forensic* components— will enable network operators to diagnose and troubleshoot the state of the routing system. In particular, the framework that we describe satisfies the following capabilities.

- Provide the ability to reason locally about BGP induced changes. By this we mean that when an event internal to an AS causes a routing change,[1] it should be possible to reason about the exact event and also identify its location.

- Provide support for the "root cause analysis" of routing events. In other words, when there is an event at a remote AS which results in BGP updates observed locally, the framework attempts to identify the remote AS, or perhaps some *small* set of *suspect* ASes which might be involved in the original event.

- Provide a mechanism to summarize the current BGP state at an AS. This can be used to present a birds-eye view of BGP routing (in the AS) to network operators.

It should be stressed that the tone of this paper is rhetorical. We sketch a very high level design but largely omit detail. Our chief intent is to stimulate discussion about the problems that we describe and understand the nature of solutions that are required. While we do present the outline of a framework that delivers the capabilities we argue for, it is intended to be a strawman construction and not a concrete design.

The rest of this paper is structured as follows: In Sec. 2, we describe in detail the problems that were alluded to, and then present arguments for a specific set of design parameters in the construction of the diagnostic capability in BGP. In Sec. 3, we describe a feasible framework that puts all the missing pieces in place. Subsequently, in Sec. 4, we address some important issues that are relevant to our design. Finally, in Sec. 5, we conclude with a summary of our arguments.

2. BACKGROUND AND MOTIVATION

BGP route updates are triggered by a variety of events such as link or session failures (or repairs), policy changes, software bugs, operator error, and so on. While some of these updates are to be expected and cannot be avoided (for example, the updates that occur in the convergence phase after a link failure), a large fraction of these can be attributed to *preventable causes* such as operator errors and misconfigurations [13]. Tracking the underlying cause(s) (or trigger event) of BGP route updates is extremely challenging. This is in part due to the "information hiding" nature of BGP. In addition, things are further complicated by the commercial environment that BGP operates in, where there is no centralized source of authority or control.

Consequently, performing "root cause analysis" of BGP updates— the initial step in troubleshooting routing problems— is extremely hard. Current efforts rely on passively monitoring BGP updates collected at a few public route servers[2] (or vantage points) and correlating the observed updates across different dimensions, e.g., destination, time, observation point, etc., to infer the (original) root cause(s) [3, 2, 12, 8]. However, these approaches have certain fundamental shortcomings: for example, an implicit assumption is that the location of the root cause is visible in the AS paths of the observed update(s), which is not true in general [16]. At a more fundamental level, the information contained in a BGP update *decreases* as it propagates through the network [9]. Thus, any attempt to make sense of BGP dynamics based only on updates collected at the public vantage points will have limited utility.

Based on these underlying facts, in the rest of this section, we present arguments that call for specific choices in the design of a *diagnostic* capability in BGP.

2.1 The case for a *reactive* component

It is well known that operator error is a significant cause of BGP instability. Effective ways to address this include designing better tools and (friendlier) languages to configure routing policy, thus reducing the scope for human error [13]. Given the complexity of router configuration,[3] it is all too easy for operators to make mistakes; even simple typographic errors can have disastrous consequences.

Alternatively, one could think of building tools to emulate the effects of configuration changes, so that operators can perform "what-if" analysis and sandbox testing on their configurations (or configuration changes) before they are applied on the live network [6]. Along the same lines, operators could also perform some sort of *program analysis* to verify the consistency of their router configurations [5].

However, such "preventive" measures cannot guarantee that routing errors will not occur. Not all routing instability can be blamed on the negligence of network operators. Many routing problems cannot be detected by verification techniques or captured by *emulation*. In fact, *locally correct* configuration can lead to route oscillations [10]. Given the sheer size and complexity of the Internet routing system, system design principles dictate that it is hard(er) to *guarantee* a complete absence of errors. In fact, assuming that such safeguards are in place, it may well be the case that the *few* errors that go unchecked will be more subtle, harder to detect and have greater impact. A more significant limitation is that these techniques offer no protection against malicious behavior.

Based on these arguments, we believe that a *reactive component* in BGP will effectively address these issues. With such a capability in place, operators may be able to track down the location and causes of routing anomalies sooner, then take the actions necessary to correct them. It should be clearly stated that we are advocating such a capability not in exclusion to the preventive measures described, but in a complementary role. In fact, both of these capabilities have distinctly *different* roles, and together might yield a greater benefit.

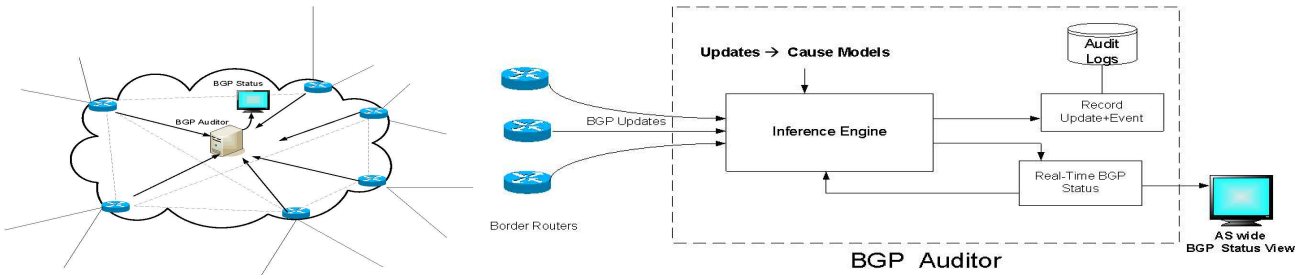
2.2 The case for *localization*

The observation made previously, regarding the lack of understanding of BGP dynamics, is certainly true of the general Internet. More surprisingly, the same problems persist, on a reduced scale, within individual Autonomous Systems. The complexity of BGP

[1] By event, we take to mean things that are *external* to the BGP protocol, for e.g., link failure, session reset, policy change, etc.

[2] Such as Route Views [18] and RIPE. Each of these maintains eBGP sessions with a number of ISP's and logs all routing updates received across these sessions.

[3] Router configuration files in backbone routers are (literally) thousands of lines long.



(a) BGP Auditor

(b) Detailed Operation of a BGP Auditor

Figure 1: The *BGP Auditor* framework within a single Autonomous System.

routing within the AS,[4] and the interaction with *intra*-domain routing protocols make it challenging to understand the impact of BGP routing changes even *within a single AS*.

Configuration and policy changes on a single BGP router, or even route updates learned from external neighbors, can have unexpected consequences in other parts of the network. Specifically, BGP route changes affect the mapping between ingress and egress routers and indirectly affect the loading on backbone links. While tools exist that can model the dynamics of IGP changes and the resulting impact on traffic loads in the network [7], the actual paths taken by traffic within the network are ultimately dictated by the particular BGP routes selected at the routers.

There are several interesting scenarios that one can think of where the debugging task would be greatly improved with the introduction of a local "eye in the sky". Consider the topology in Fig. 2. Here, AS 1 is a customer of AS 2, which is itself a customer of AS 3 and AS 4. Now suppose, for some reason, AS 1 loses reachability for a certain prefix (say p) which has vanished from its routing table, and this is brought to the attention of the operators in AS 2. The operators in AS 2 can query their routers to see if p is in their own tables and, if so, find out why it is not being advertised to AS 1. However, if p is not known to the relevant routers in AS 2, troubleshooting becomes more complicated: it could have been withdrawn by the upstream providers or caused by a buggy *export* filter in AS 2. Note that there is almost no information at the routers in AS 2 to rule out either one or the other possibility.

Now, consider another scenario: at some time, the router 2.2 is suddenly swamped with outbound upstream traffic (to AS 3) while at the same time, 2.3 becomes nearly idle. This in turn causes congestion and packet loss internally in AS 2. Suppose that the underlying cause for this was a router reconfiguration somewhere in AS 2 that changed the IGP metric for a link, which in turn adjusted iBGP nexthop calculations, which in turn switched upstream traffic from one provider to another. Currently, it is very hard to pin all these changes down to the (single) IGP weight change.

There are several reasons why a local framework, i.e., each AS employing a single entity that presents a "centralized" view of BGP routing in the AS, would be effective: first, this lends itself easily to providing operational reports of the current BGP "state" of the network, which will be useful to operators. For example, internal IGP changes often result in traffic loads being shifted between egress routers, and this could be easily detected by such a system; secondly, such a capability would be *effective* because it captures

[4]Such as introduced by Route Reflection hierarchies used in large networks.

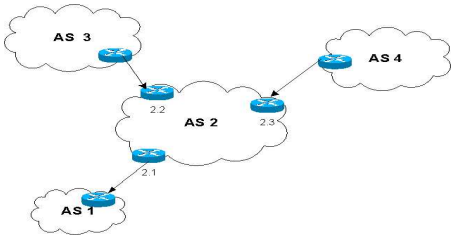


Figure 2: Directed arrows represent provider-customer edges. So for example, AS 1 is a customer of AS 2.

BGP updates at the *source*, where the *information content* of BGP updates is still considerable and "unattenuated". Lastly, addressing this important problem is challenging, even in this more restrictive setting, but has the advantage of separating technical issues from the more commercial and administrative difficulties that enter the picture when more than one AS is involved. However, by first implementing the functionality within individual ASes, it lowers the barrier to deploying a more comprehensive distributed framework. We will explore this last aspect further in the next section.

3. STRAWMAN DESIGN

In this section, we outline a *localized* framework which will introduce a *diagnostic, troubleshooting* capability to BGP operation. The key entities in this framework are so-called *BGP Auditors*. These AS-specific entities are responsible for tracking BGP routing changes and associating them with possible causes. *BGP Auditors* put into place a crucial element of the reactive capability that we advocate in this paper. By receiving BGP updates from all the routers, the *Auditor* can reconstruct a sequence of route changes for routers (within an AS) and build a hypothesis about the underlying trigger event.

In the rest of this section, we provide more detail about the design, capabilities and operation of the *BGP Auditors* and also discuss their utility in a larger, *distributed* setting.

3.1 *BGP Auditors*

In the framework we describe, each AS contains a *BGP Auditor* entity, the function of which is twofold: first, to construct and maintain an *audit trail* that can *explain* the "origins" of updates being sent by routers in the AS to neighbors; and second, to maintain an AS-wide *BGP Health Report*.

The inputs to the *BGP auditor* are the route updates generated by

all the routers in the AS (see Figure 1(a)). The collected updates are analyzed and the resulting analysis is used to: (1) update a real-time AS-wide status report, and (2) build an *audit trail* of BGP events. The *BGP status* report can be used by operators to see in real-time how BGP route updates are affecting the routers in the AS. More significantly, it can be coupled with other information— netflow summaries for instance— to provide a more *performance oriented* view of the network. On the other hand, the *audit trail* serves as a historical, forensic record that can be used to follow a chain of route changes leading to the root cause.

At a very abstract level, there are three distinct functions that are embedded into the *BGP Auditor*. The logical relationships between these different functions are shown in Figure 1(b). Note that the functional separation is somewhat arbitrary and is simply used to aid in the high level description.

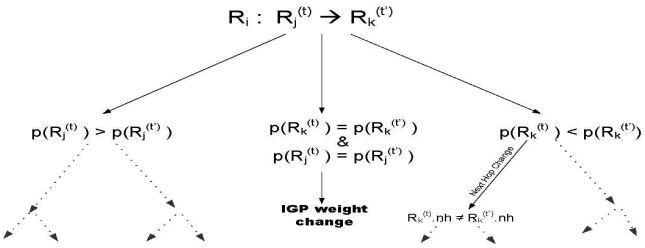


Figure 3: Decision Tree for a route change event (partial tree is shown). $R_j^{(t)}$ represents the route being used by router R_j at time t, and $p(R_j^{(t)})$ is the *relative preference* of this route. $R_j^{(t)}$.nh represents the NEXT_HOP attribute.

3.1.1 Inference Engine

The main task of the *Inference Engine* is to reason about the actual causes behind updates that are received (from various internal routers). In other words, given a set of routing updates generated by routers within the AS, it determines whether the *trigger* for the updates is internal to the AS, and if so, exactly what the root cause (and "best guess" location) is.

The inputs to the inference engine are the set of BGP updates from all the routers, as well as the routing policies specific to the AS. The inference engine attempts to *map* the set of updates to a particular root cause, i.e., link failure, policy change, session reset, etc.

To provide a very trivial example of how the inference would proceed, consider the following example: R_i, R_j, R_k are border routers of an AS with iBGP connections to each other. Suppose that at time t, router R_i had selected the route from R_j as its best route, but switches to the route from R_k at a future time (say t'). Figure 3 represents a portion of the "decision tree" that can be used to reason about this route change event. Note that this event involves only the routers R_i, R_j and R_k. There are *exactly* three possibilities that the route change event is predicated upon: (1) R_j announced a worse route (or withdrawal) to R_i, (2) R_j and R_k did not change their routes, but the relative preference of the routes changed at R_i, and (3) R_k announced a better (or new) route to R_i.

For the sake of argument, let us follow the middle branch in Fig. 3; i.e., R_i changed its best route without any route updates from R_j or R_k. This implies that the route update was caused by the *inversion* of the route preferences (for R_j and R_k). By following the route selection process carefully [1], it is clear that the route update was caused by a change in the IGP distance; either R_k has become "nearer" to R_i, or R_j "farther". This can be verified by correlating with other destinations for which R_j exhibits the same dynamic. In the absence of a *BGP Auditor*-like function, the same inference would be complicated: the particular BGP route update would have to be correlated (in time) with a corresponding IGP update that explicitly signals that IGP distance change [17].

In a similar vein, we can construct a chain of reasoning about the other two possibilities.

3.1.2 Maintaining *Audit Trails*

The updates and events that result from the analysis carried out by the Inference Engine are transformed into *audit records*— compact representations of route change— which may be queried at a later time. The audit records might be stored off-line, perhaps in a database for some period of time. With the availability of the *audit trail*, queries such as: "Was the update for 128.101.0.0/16, at 00:23:26 CDT, caused by an internal event?", or "What caused the updates from router 192.168.0.1 for destination 128.101.0.0/16 at 00:23:26 CDT?" might be easily answered.

This *forensic* functionality enables a wider inter-AS diagnostic framework, discussed at the end of this section. In this framework, BGP Auditors in neighboring ASes to query each other, making it possible for the Auditors to track down the cause of routing changes that originate from outside their own AS.

3.1.3 Real-time BGP Status

This provides the operators of the AS with an "eye in the sky" view of BGP status as it applies to their own AS. Such a capability is important for a variety of reasons; as discussed previously, BGP dynamics have a huge impact on the flow of traffic through a network. The abstracted BGP view allows operators to visualize the impact of BGP routing updates to the network. In cases when the change is unanticipated, the visual depiction may provide clues to the operator as to what might be wrong. It is also feasible to think of other tools and applications that might be integrated with this component. For example, by incorporating *netflow* data about the prefix volumes, the operator can easily visualize how BGP updates would affect the traffic loads on different peering links. While tools such as this might already be in operation in large ISP networks, we intend that the *all-in-one* nature of the BGP Auditor will make it easy for other *smaller* networks to deploy.

The most important benefit of this particular function that the BGP Auditor enables may well be that it provides individual ASes an important *incentive* to actually deploy the *BGP Auditors* in their networks. Thus, while the "real-time BGP status" capability is derived directly from the core function of the BGP Auditor, i.e., event-cause reasoning, it provides operators with the necessary incentive to deploy the *reactive component* into their networks.

3.2 Cooperation for *Root Cause Analysis*

When events internal to an AS cause routing updates, the event will be captured by the local *BGP Auditor*. However, events external to the AS cause route changes within, the *Auditor* in an AS does not have the *scope* required to pinpoint the location (and cause) of the original event many hops away. Consider the example in Fig. 2. Suppose that a link failure in AS 1 causes updates to travel through AS 2 and then to AS 3. At AS 3, the *Auditor* can only infer that the updates it saw originated from AS 2, which is only part of the answer. Thus, in order to enable effective and accurate root cause analysis, it is necessary to allow *BGP Auditors* in neighboring networks to query each other. Then, AS 3 would be able to query AS 2, who in turn would query AS 1 to arrive at the correct answer.

Note that the use of *audit trail records* enables the querying operations to be performed off-line on a much slower timescale. This

is especially important because the querying action might be triggered by an end network long after the route change event actually takes place in the local AS.

3.3 Improving BGP "Expressiveness"

The difficulties in understanding the dynamic behavior of BGP are directly tied to its "information abstracting" nature. In fact, these are essential to allow it to scale and operate in the commercial environment of the Internet. For example, the abstraction of route information is an essential requirement for a protocol that carries the entire Internet prefix space. The same principle is also implicit in the hop by hop route forwarding operation, which decreases the information contained in an update. These properties have important consequences for us: they limit what can be inferred by looking at BGP updates alone. The information contained in BGP route updates is insufficient to clearly distinguish different possible events.

To illustrate this, consider a sequence of BGP routers A,B,C,D,E and F, such that each pair of successive routers share a BGP session. Now consider two scenarios: in the first, the link between A and B fails, and in the second, the BGP session between D and E fails. In both scenarios, E will send the *same* withdrawal message to F about the destinations originated by A. This makes it impossible for F to distinguish between the two *different* failures. To state this in a more general way, BGP updates are "transformed" as they are propagated through the network, usually in a way that obscures the cause and location of the actual event [9].

This same situation also exists when the event is local to an AS. Suppose that router A in a certain AS maintains peering sessions with routers B and C in a neighboring AS. Now, when the peering session between A and B fails, A generates routing updates indicating that it is using C*— there is no direct information that reveals the session failure.*[5] Taking this one step further, suppose the particular peering session (between A and B) begins to flap: how can we know that the fault lies with this peering session and not some other peering session a few hops away?

To address this, we can use *additional* BGP route attributes, which will act as *tracers* (or markers) that remain invariant as the updates are transformed. For example, when a link fails, the router that first detects it will generate a withdrawal and explicitly include the identity of the failed edge in the message. Note that this would make the task of identifying the "root cause" trivial. Thus, an important consideration in the design of the *BGP Auditor* framework would be to understand exactly what additional attributes may enable *accurate* and *reliable* inference by the *BGP Auditors*.

However, note that it may not be feasible to export this additional information outside the AS.[6] In this case, the information is used only by the local *BGP Auditor*, and suppressed when the route is propagated to external neighbors. In fact, this is already implicit in BGP today, as iBGP advertisements are much richer and more expressive than eBGP advertisements.[7]

4. CHALLENGES

In this section, we briefly list some of the issues that are raised by our strawman design of a localized framework, then discuss ways in which they can be effectively addressed.

[5]Note that the same behavior would be observed if B sent withdrawals to A.

[6]For example, an ISP might not want to have its routing policies revealed to its neighbors.

[7]The so-called *non-transitive* attributes are only meaningful within an AS.

Scalability: There are two distinct concerns related to scalability: first the additional overhead imposed upon BGP routers in the AS; and second, the computation and memory requirements at the *BGP Auditor* itself.

To address the first issue, note that the least disruptive deployment scenario would entail each BGP router maintaining an iBGP session with the *BGP Auditor* over which it sends any route updates. This would involve minimal additional memory processing requirements at the BGP routers. Consequently, we believe adding the new component will not disrupt the normal BGP operation of an AS in any way.

To address the second concern, note that the *BGP Auditor* can summarize the state of each router by keeping track of two routes, the current best route and the previous selection, for each prefix. Considering that the largest ISP networks today contain about 600-700 routers, the overhead at the *Auditor* is to maintain the same number of TCP connections, which is well within the realm of possibility. Additionally, since it is not "directly" in the BGP control plane, it has to do a lot less work than a router. For example, it will not run the best path computation, generate updates, or maintain routes in memory (it can simply stage them out to a disk). Most importantly, it does not have to operate under real-time constraints like a router would have to. The inference algorithm and "BGP health status" update can be run at much slower timescales than the rate at which updates are received, say every 15 minutes. Thus, it is entirely feasible to run the *BGP Auditor* on a commodity PC with modest storage and processing capabilities.

In the light of these factors, we do not believe that there are significant scalability concerns associated with the framework we propose.

BGP Protocol Limitations: As discussed in Sec. 3.3, the effectiveness of the *BGP Auditor* inside an AS is limited by the expressiveness of the BGP updates it receives. The best "granularity" of inference may well be a mapping of the updates into *equivalent* event classes [2]. This somewhat restricted capability is far more effective than the current capabilities. At the same time, we believe that additional attributes can significantly increase the effectiveness of our proposed framework. An important issue is how the additional attributes will be incorporated into BGP and whether this would require changes to the protocol. However, note that BGP has support for extensible attributes, and these changes can be accomplished without significant disruption to an AS.

Moreover, it is a relatively easy administrative task to upgrade the software at the routers in a single AS (*all* the routers do not have to be upgraded at the same time). Thus, there is an easy, incremental path for the routers to be upgraded (to introduce the additional attributes).

Inter-AS Cooperation: Accurate root cause analysis requires ASes to cooperate. When an external change causes routing changes within an AS, i.e., some routers in the AS change their best route selections, then it is necessary to query the sequence of ASes that propagated the routing change. To enable this, *BGP Auditors* need to be deployed in a number of ASes and moreover, *they must be allowed to query each other*. This brings up several issues: first, ISP's might be averse to being queried by neighboring ASes; second, there is the issue of how much information should be exchanged when *BGP Auditors* query each other and what is the nature of the information that will be returned? For example, if AS 1 asks AS 2 about whether it was responsible for a particular update, and the update was external to AS 2, should AS 2 query its own neighbors? Lastly, if BGP Auditors are allowed to query each other, how can

we be sure that the answer can be trusted?

To address these concerns, note that ISPs already enter into service-level agreements (or SLA's) with their customers, which establishes a basis for cooperation. It might be a selling point for an ISP to enable this functionality as part of the SLA. If bilateral arrangements to query BGP Auditors are in place, it becomes possible to perform root cause analysis by constructing a chain of bilateral queries. However, this might not be feasible between ASes with mutual peering arrangements, as such arrangements traditionally do not include SLA's or other performance assurances.

At this time, we do not have definitive answers to address all of these issues effectively. Any specific mandate is bound to infringe on some ISP's (current) policies. In this paper, rather than provide answers to these specific questions, we wish to simply raise some interesting questions and hope that its leads to greater discussion in the broader community.

5. CONCLUSION

Building a *diagnostic* capability in BGP can help operators to effectively pinpoint the cause and location of routing faults, which is very hard to do in today's Internet. In this paper, we presented arguments for such a capability and also discussed how it is enabled by so-called, AS specific *BGP Auditors*. BGP Auditors maintain state about routers in an AS and attempt to "explain" routing changes. By querying similar entities in neighboring ASes, BGP Auditors can construct an accurate picture of how a route change is propagated, even when the root cause event is far away. Thus, the *BGP Auditors* enable the forensic capability that is sorely lacking in Internet routing.

In addition, *BGP Auditors*, by virtue of abstracting the routing state of the entire AS, enable several other interesting applications. For example, one could think of specifying high level AS routing policy at the *BGP Auditor*. Route updates generated by routers in the AS could be verified against this high level policy to ensure that they do not violate the *intended* router level configuration.

Clearly, there are many missing details in our high level description. Rather than present a complete specification, we intend the arguments presented in this paper to simulate discussion about specific ways to improve the (current) state of BGP.

6. REFERENCES

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

[2] CAESAR, M., SUBRAMANIAN, L., AND KATZ, R. H. Root Cause Analysis of BGP Dynamics. Technical Report., University of California, Berkeley, 2003.

[3] CHANG, D.-F., GOVINDAN, R., AND HEIDEMANN, J. The Temporal and Topological Characteristics of BGP Path Changes. In *ICNP* (November 2003).

[4] CLARK, D. D., PARTRIDGE, C., RAMMING, J. C., AND WROCLAWSKI, J. T. A Knowledge Plane for the Internet. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications* (2003), ACM Press, pp. 3–10.

[5] FEAMSTER, N. Practical Verification Techniques for Wide-Area Routing. In *ACM SIGCOMM Workshop on Hot Topics in Networks (HotNets-II)* (November 2003).

[6] FEAMSTER, N., WINICK, J., AND REXFORD, J. A Model of BGP Routing for Network Engineering. In *SIGMETRICS* (New York, NY, June 2004), ACM.

[7] FELDMANN, A., GREENBERG, A., REINGOLD, N., AND REXFORD, J. Netscope: Traffic Engineering for IP Networks. *IEEE Network Magazine* (March 2000), 11–19.

[8] FELDMANN, A., MAENNEL, O., MAO, Z., BERGER, A., AND MAGGS, B. Locating Internet Routing Instabilities. In *Proc. ACM SIGCOMM* (2004).

[9] GRIFFIN, T. What is the Sound of one Route Flapping. Network Modeling and Simulation Summer Workshop, Dartmouth, 2002.

[10] GRIFFIN, T. G., AND WILFONG, G. An Analysis of BGP Convergence Properties. In *Computer Communication Review*, vol. 29. ACM, October 1999.

[11] As 7007 incident. `http://www.merit.edu/mail.archives/nanog/1997-04/msg00340.html`, April 1997.

[12] LAD, M., MASSEY, D., AND ZHANG, L. Link-rank: A Graphical Tool for Capturing BGP Routing Dynamics. In *Proc. of IEEE/IPIF NOMS* (Apr. 2004).

[13] MAHAJAN, R., WETHERALL, D., AND ANDERSON, T. Understanding BGP Misconfiguration. In *Proc. ACM SIGCOMM* (October 2002), ACM Press.

[14] Router holes threaten net. `http://zdnet.com.com/2100-1105-990608.html`, February 2003.

[15] NORDSTRÖM, O., AND DOVROLIS, C. Beware of BGP Attacks. *SIGCOMM Comput. Commun. Rev. 34*, 2 (2004), 1–8.

[16] TEIXEIRA, R., AND REXFORD, J. A Measurement Framework for Pin-pointing Routing Changes. In *ACM SIGCOMM Network Troubleshooting Workshop* (August 2004).

[17] TEIXEIRA, R., SHAIKH, A., GRIFFIN, T., AND REXFORD, J. Dynamics of Hot Potato Routing in IP Networks. In *SIGMETRICS* (June 2004), ACM.

[18] VIEWS, R. University of Oregon Route Views Project. `http://antc.uoregon.edu/route-views/`, 2000.