Solving the Interdomain Routing Puzzle –
Understanding Interdomain Routing Dynamics

by

Zhuoqing Mao

B.S. (University of California at Berkeley) 1998
M.S. (University of California at Berkeley) 2000

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Computer Science

in the

GRADUATE DIVISION
of the
UNIVERSITY of CALIFORNIA at BERKELEY

Committee in charge:

Professor Randy H. Katz, Chair
Professor Scott Shenker
Professor John Chuang

Fall 2003

The dissertation of Zhuoqing Mao is approved:

Chair Date

Date

Date

University of California at Berkeley

Fall 2003

Solving the Interdomain Routing Puzzle –

Understanding Interdomain Routing Dynamics

by

Zhuoqing Mao

Abstract

Solving the Interdomain Routing Puzzle –

Understanding Interdomain Routing Dynamics

by

Zhuoqing Mao

Doctor of Philosophy in Computer Science

University of California at Berkeley

Professor Randy H. Katz, Chair

BGP, the Border Gateway Protocol, is the de facto standard protocol for performing interdomain routing on the Internet today. Its main function is to distribute reachability information across the Internet, serving as the "glue" that holds the Internet together. BGP allows flexible configuration of routing policies from each local network and scalable operation, both at the cost of global visibility leading to complex and hard to predict dynamic behavior.

BGP's dynamic behavior has so far received relatively little attention in the research community, due to ill-understood operational practices as well as insufficient resources to perform experiments. This thesis combines controlled active measurement in a testbed environment as well as on the actual Internet, correlating routing traffic with the forwarding plane and analyzing the protocol in detail using simulations to expose problems of interdomain routing dynamics. This class of problems may be difficult to reason statically due to the interaction of protocol components and can be observed more easily during run time. They include reachability, forwarding behavior,

convergence dynamics.

The first part of this work exposes an unexpected interaction between two protocol features in BGP – route flap damping and path vector-based convergence, significantly delaying convergence. It illustrates the need to carefully analyze BGP's run time behavior and shows the inherent complexity in the BGP dynamics. It provides an excellent example of how new features added to BGP for the purpose of solving an immediate problem can create additional hidden problems due to protocol feature interactions. Better visibility into the dynamic protocol behavior through measurement and analysis greatly helps expose such problems. A mechanism such as route flap damping, although specified in detail in the RFC, is in fact difficult to use operationally due to the many knobs it provides. This leads to the second part of our work.

We designed and implemented an active routing measurement infrastructure – *BGP Beacons*. It differs from previous efforts in its public and long-term nature and several experimentation features to facilitate measurement interpretation. We demonstrate several uses of this infrastructure to better understand BGP dynamics, one of which being the validation of our conjecture of the interaction of route flap damping and convergence as described in the first part of the dissertation.

Contrary to common wisdom, the routing information may not always reflect the underlying packet forwarding behavior or how data packets flow on the Internet. The discrepancies can be caused by various routing anomalies and need to be carefully studied. To understand the impact of routing dynamics on the data plane dynamics, in the last part of the thesis, we describe the development of a tool and associated optimization algorithms to study the interaction between the control plane and the data plane.

The main contribution of this dissertation is an experimental framework for improved understanding of the dynamics of interdomain routing. Using this framework, we validated our conjecture that route flap damping can delay convergence. We focus on routing problems such as reachability and forwarding behavior that are difficult to reason statically due to lack of topology and policy information and illustrate through measurement they can be much better understood. As a byproduct of this work, we contribute to the research and operator community an active measurement infrastructure for controlled BGP route injection – BGP Beacons, and a tool to correlate the routing behavior with the packet forwarding behavior – AS-traceroute tool.

Professor Randy H. Katz
Dissertation Committee Chair

To my loving parents, who are always there for me

and my dear friends, who made the journey fun.

Contents

List of Tables

List of Figures

Acknowledgements

A professor in our department once said that *PhD* actually stands for “poor”, “hungry”, and “driven”. Naturally, students don’t have much money, but PhD students are hungry for knowledge. And when they work on research, they are very driven and dedicated in search of breakthroughs. I couldn’t agree more with his comment. However, I have to add to his definition that h should also stand for “happy.” Getting a PhD was an extremely fun and rewarding experience, but also challenging. My journey of getting PhD was uneventful. Now that I am close to the end, I am amazed at how much support I have received over the past five and a half years that made the experience enjoyable.

First, I would like to thank my advisor, Professor Randy Katz, for his guidance, support, and his trust in me throughout the years. Randy is my role model in many ways. I am very grateful for his high standard for research, care for students, kindness and sincerity. Despite his busy schedule, he meets with us regularly, sometimes even on weekends. Even when I was away doing internships, he still took time to talk to me on the phone on a weekly basis. He provided amazingly detailed feedback on this dissertation with surprisingly fast turn-around time. He will continue to be my role model in the future.

I would like to thank my dissertation and qualifying exam committee members, Professors David Culler, Scott Shenker, and John Chuang for their comments and valuable feedback.

I was extremely fortunate to have had many mentors during my PhD career. Professor Ramesh Govindan gave me the opportunity to work on the route flap damping analysis, which is the starting point of my PhD thesis. He along with Professor George Varghese had been an inspiration for me with their conscientiousness and insightful comments.

The latter two parts of my dissertation were completed while I was at AT&T Labs-

Research. I thank Dr. Fred Douglis for giving me the opportunity to start the collaboration with the lab in the Summer of 2001. I had the good fortune to work with many excellent researchers during the many summers I spent in New Jersey: Chuck Cranor, Tim Griffin, David Johnson, Jennifer Rexford, Matthew Roughan, Oliver Spatscheck, Kobus Van Der Merwe, and Jia Wang. Chuck Cranor was always helpful with BSD related questions. I would like to say special thanks to Tim Griffin, whose tutorial on BGP created interest in me to work on BGP. It was such an honor to work with David Johnson, who provided many feedback on the various algorithms in my work. Jennifer Rexford is the best mentor and role model a graduate student could ask for. She always gave me amazingly insightful comments. I learned so much from her in both ways to do research as well as writing and presentation skills. Matt Roughan taught me many things in statistics and it was a lot of fun to work with him. Both Oliver and Kobus had been great mentors to me, giving me many professional advices. Jia Wang had been a very good friend and also a great researcher to work with. I also thank Joel Gottlieb, Fred True, and Carsten Lund for their support in data analysis and computational resources.

The Beacons work would not have been possible without the support of the Beacon hosts: Randy Bush, Dave Meyer, Andrew Partan, and Geoff Huston. I especially thank Randy Bush for his patience and critical feedback. Many thanks are due to the RouteViews support stuff John Heasley, and Joel Jaeggli. I also thank the RIPE support staff for making their data public and for setting up the RIPE Ris Beacons.

New Jersey would have been a very lonely place without the many friends I had there. Aman Shaikh was my buddy and lab-mate. He really made my stay there a lot of fun. I also had the pleasure of the company of many other interns: Amit Sehgal, Madanlal Musuvathi, Ruomei Gao, Hyunseok Chang, Rob Sherwood, Ting Yu, Dan Pei, and Renata Teixeira.

During the first several years of my graduate study, I was part of the Ninja and ICEBERG projects. I sincerely thank the group members for their feedback on my work. Many thanks are due to all my Berkeley friends: H. Wilson So, Alec Woo, Xia Hong, Adam Costello, Drew Roselli, David Oppenheimer, Angela Schuett, Tina Wong, and Andy Begel. I especially thank Wilson for his support, encouragement, and faith in me.

Finally, I thank my parents for always believing in me.

Chapter 1

Introduction and Motivation

The Internet today is owned by no single administrative entity, but instead consists of thousands of networks. Each has its own routing polices. A network belonging to a single administrative entity is considered a unit of routing policy, also known as an *Autonomous System*. One administrative entity, however, can have more than one Autonomous System. As of October 2003, there are about 17,000 Autonomous Systems visible in the routing tables [55]. The Internet is expected to grow more complex with increasing number of users, as more people get online, more places get wired, and more stub networks are connected or multihomed to multiple upstream providers for increased redundancy. Thus, a better understanding of how to make the Internet robust and fault-tolerant is increasingly important. To achieve these goals, it becomes critically important to have better insight into the run-time behavior of BGP: to answer why routes take so long to converge, some destinations are unreachable, and packets flow along an unexpected path deviating from the routing information. Answers to these questions today are not easy to obtain due to lack of visibility. We describe how our work enables debugging routing problems and gives insight into the dynamic behavior of BGP.

Routing is the control plane protocol responsible for finding ways to reach any destination on the Internet. There are two types of routing protocols – intradomain and interdomain routing. Together they achieve global reachability. Within each AS, *intradomain routing protocols* also know as *IGP* (Interior Gateway Protocol) such as OSPF [77] [78], IS-IS [81], and RIP [71] are responsible for maintaining reachability from any location within a given AS to any other location within the AS by calculating and advertising routes within an AS. The *interdomain routing protocol*, also known as *EGP* (Exterior Gateway Protocol), *i.e.,* the Border Gateway Protocol (BGP) [54, 92, 100] guarantees global reachability across the entire Internet by computing routes in other ASes while conforming to the commercial relationships between ASes. There are three major AS relationships between two ASes creating an Internet-scale relationship hierarchy: *customer-provider*, *peering*, and *siblings*. Customer networks pay their providers for transit or carrying their traffic to the rest of the Internet. Peers exchange traffic to their respective customers for free. Siblings or mutual transit agreements allow a pair of ASes to provide connectivity to the rest of the Internet for each other. Besides conforming to these relationships, the typical operational routing practice does "hot-potato routing" given the choices of multiple exit points between two networks: A network usually tries to hand off the traffic destined to other networks as soon as possible by selecting the earliest exit points. This reduces the amount of traffic carried in one's own network and is the main reason for asymmetric routing where the forward and reverse path differs due to each AS in the AS path attempting early exit, affecting later AS's routing decisions.

Both interdomain and intradomain routing protocols dynamically adapt to failures and attempt to route around them. Due to the commercial nature of the Internet, the dynamic behavior of BGP which is essentially policy-based routing has been rather difficult to understand due to lack of policy and topology information. In Section 1.1, we describe the state of the art of BGP's operations and our limited understanding of how well Internet routing works today and the causes for known routing problems we observe. We briefly summarize previous measurement studies in this area to show our understanding of BGP's performance today. In Section 1.2 we summarize the difficulties in understanding interdomain routing to provide strong motivations and illustrate the importance of our work. We precisely define the problems in BGP dynamics we study in this work and our key evaluation metrics in Section 1.3. At the end of this chapter in Section 1.4, we describe our framework for understanding and improving BGP dynamics and how an active measurement infrastructure can disambiguate the interpretation of routing dynamics and provide insight into routing problems. Finally, we provide an overview of this dissertation work.

1.1 How Well Does Internet Routing Work Today?

We would first like to understand how well Internet routing works today. We first turn to understanding *how* to evaluate Internet routing. In fact, today we do not have good tools to answer this question and this is a main motivation for our work – build better tools to answer questions about routing dynamics.

1.1.1 Difficulties in Evaluating Internet Routing

When a customer complains about routing problems either in terms of reachability or poor performance, it typically is in the context of some applications. It is not easy to understand the root cause of such problems, especially when they are caused by suboptimal routing decisions. There are many reasons why performance degrades. Network operators, for instance, can install filters in their routers to determine which to accept in calculating the best forwarding path. Packet filters at the routers are much more flexible in the sense that they determine which packets are accepted for forwarding based on attributes of the packets, *e.g.,* port numbers and protocol types. Given a route in one's routing table received by one's upstream provider, there is no guarantee that all application traffic can reach the destination due to the presence of packet filters. Some networks, for instance, perform port-based filtering to protect against known worm traffic. When debugging routing problems, one needs the application's view to understand which type of application traffic is being correctly forwarded.

Thus, it is difficult to judge the quality of Internet routing because of the inability to determine if application degradation is due to routing problems. Network operators have very limited tools to debug routing problems. Only primitive tools like traceroute and ping are used to determine existing routing behavior. However, such probes may not reproduce problems experienced by applications. They require support from routers, which is not universally available. Moreover, there is little visibility into the routing behavior of other ASes from a given AS's perspective, making it even more difficult to identify the source of any routing anomalies. Thus, it is difficult to predict the impact of routing policy changes on global routing behavior. Nevertheless, an important class of routing problems are relatively easy to detect. They involve the lack of reachability for destination prefixes caused by the unavailability of certain routes in the routing tables. Another class of more easily identifiable routing problems involve forwarding loops either within or between ASes. Such routing problems can be detected based on routing table information and routing announcements can reveal the source of the problems.

One metric, recently proposed by Randy Bush to measure the health of Internet routing system is to determine how "happy" packets are [26] in terms of amount of loss, jitter, delay, and reordering they experience. He argues that the quality of the control plane should be judged by measuring the data plane. This certainly is a meaningful metric, as ultimately the goal of routing is to provide good quality data paths for applications. There has been a large amount of work on metrics evaluating the data plane [58]; In theory it should be easy to apply them to evaluating routing protocols. Yet there is no measurement infrastructure in place to collect such data on a wide scale to systematically evaluate Internet routing over time. Instead, researchers in this area with access to unused address spaces inject artificial routing changes, resort to measuring convergence times of such routing changes as a way to evaluate routing performance. We discuss this next. Previous work in evaluating BGP performance provides motivation for our experimental framework, as they reveal there are many unexplained long convergence delays.

1.1.2 BGP Convergence Times

We now describe the current experience of researchers measuring BGP convergence times based on artificially injected routing changes on the Internet. It shows that we do not have an explanation for some important, but unexpected routing behavior and there is a need for an ongoing experimental infrastructure for BGP monitoring. Labovitz *et al.* has conducted numerous studies between 1997 and 2001 on Internet routing performance [63, 65–68]. In his studies, *convergence time* is defined to be the delay in observing an injected routing change from the BGP measurement data collected from various locations. The main observation in terms of routing performance is the following:

- Most injected routing changes converge within 3 minutes.
- There exist pathological cases where convergence is delayed up to more than 15 minutes.
- Convergence delay is directly proportional to the length of the longest backup path between the source and destination AS for the route.
- Errant or "vagabond" paths are frequently explored during delayed convergence, possibly due to software bugs and misconfigurations.

Thus, the general consensus based on empirical data is that routing changes usually take on the order of minutes to propagate. Based on operational experiences, minor routing problems occur occasionally and then eventually disappear [18]. Operators may not understand why these problems occur. As most of these routing problems do not persist, there has not been any exhaustive studies to explain them. It is also known that router misconfigurations occur frequently [70] and may result in routing anomalies [6]. This motivates our work to provide an ongoing and continuous monitoring of BGP dynamics to expose problems when they occur. As important routing problems are typically associated with degradation of application performance, such an infrastructure requires correlation of the data plane with the routing plane. We will describe such a framework in Section 1.4.

1.2 Difficulties in Understanding Internet Routing

We have described how researchers currently measure the health of interdomain routing. We now summarize the difficulties in understanding Internet Routing which is a major motivation for work in this dissertation.

- Unknown information: local policies and internal topologies of ASes are considered private information and not revealed globally. Various Internet mapping and policy inference efforts exist [69, 99]; however, they are hard validate. The routing behavior heavily depends on both the policy and topology information; therefore, it is rather difficult to do root cause analysis given a feed of BGP updates today [49].
- Ambiguous specifications: the protocol specification for BGP as defined in RFC 1771 intentionally left out some details by giving the freedom to the vendors on defining things such as whether MinRouteAdvertisement Timer is applied per prefix or per peer. Thus the behavior

the protocol depends on the router implementation variants.

- Operational realities differ from specifications (RFCs): similarly to the above point, vendors may deviate from the router specifications, suiting to their own router architecture design. Suggested default timer values, for example, are often not followed.

- Large distributed systems: Internet is a large distributed systems, consisting of thousands of autonomous systems, each can have hundreds of thousands of routers. The dynamics of such a system can be very difficult to reason.

- Local changes may or may not propagate globally depending on the the policies: causal analysis of BGP updates is extremely hard, as there is often insufficient information on whether a local routing change can affect other ASes' best route selections.

We now describe a simple example, where even if given the precise topology and policy information, it is still extremely difficult to infer the root cause of a routing update. In Section 1.4 we will refer back to this example to show how our proposed framework could ease the diagnosis of any routing problems associated with this example. We examine a simple topology in Figure 1.1 used by many previous works [49]. Here we have a 5-node topology, each node denotes an AS for simplification. At each node, we list the alternate routes to destination in AS1 in the order of decreasing preference. Now, assume in steady state all nodes use its most preferred route to destination AS 1. And now we receive an update from AS5 to our route monitor of the AS path $[5\ 4\ 3\ 1]$. The question is what routing changes has occurred to cause AS5 to send out this routing change. There can be several possibilities. We describe a subset of them below.

1. AS3 sends out a withdrawal as the link between AS1 and AS3 is down. This causes AS4 to subsequently send to AS5 the route of $[4\ 3\ 1]$ and triggering AS5 to send out $[5\ 4\ 3\ 1]$.

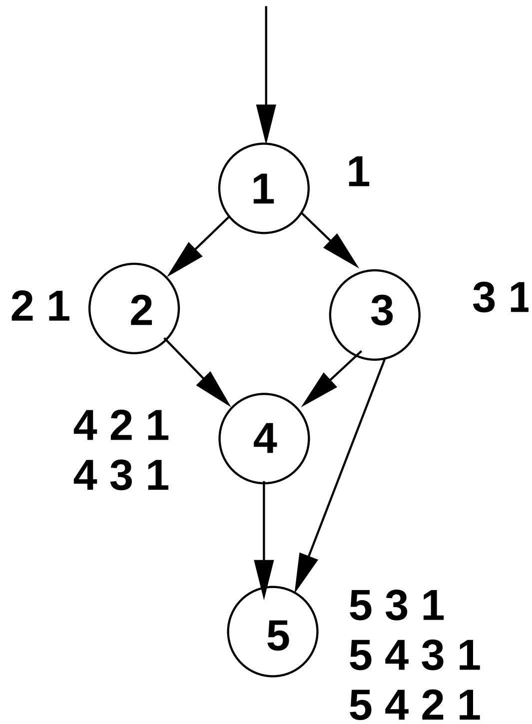


Figure 1.1: A simple topology to illustrate the complexity of BGP dynamics.

2. The link between AS5 and AS3 is down.

3. AS5 is doing traffic engineering, its routing preference has changed preferring the longer route over the previously announced shorter route.

4. AS2 withdraws its route to AS1, triggering AS4 to announce [4 3 1], causing AS5 to prefer [5 4 3 1] to [5 3 1].

This example shows that even knowing the initial routing preference and actual routing topologies, it is difficult to identify the root cause for any routing change. This strongly motivates an experimental infrastructure where the routing change can be precisely *controlled* to understand its impact on the Internet. In such an infrastructure, it then is possible to measure basic metrics such as convergence delays. It also allows us to understand policies and configuration settings used in remote networks.

Given such difficulties of understanding Internet routing, we would like to emphasize the importance of research in this area. First of all, if the control plane experiences bad performance, then the data plane will certainly be affected. Today's trend of multi-homing or connecting to multiple upstream providers from stub networks is rather expensive and does not even provide any guarantees of good fail-over behavior when link fails. And both upstream providers can be affected by a single failure due to limited path diversity. Secondly, there is an increasing concern of protecting the Internet infrastructure against attacks after the September 11 incident and the more recent power outage on the east coast, as we heavily depend on the performance of the Internet at large. In the latter incident, several networks were known to have experienced reachability problems [35]. Security researchers have also recently observed a growing number of infrastructure attacks. It is therefore critical to understand the routing dynamics, both in the case of malicious attacks as well as unintentional misconfigurations.

1.3 Problem Definition and Evaluation Metrics

In this section, we precisely define the problems we address in the area of BGP dynamics and describe the evaluation metrics for our measurement methodology. This dissertation focuses on providing more *visibility* to understand BGP routing dynamics that affect *reachability* (availability of a prefix in the routing table), packet-level *forwarding behavior* (the actual AS-level routes used in forwarding packets), and transient *routing convergence* and fail-over behavior (the sequence of routes explored when a routing change occurs). We achieve this by building an active measurement infrastructure (*BGP Beacons*) performing controlled routing experiments, detailed protocol analysis through modeling and simulation with direct input from a router testbed, and finally a tool called *AS-traceroute* that enables the correlation between the routing control plane and data packet forwarding plane. We choose these three areas in routing dynamics for our work, as they have a big impact on the application performance and determine the performance and stability of the Internet routing system.

We use the following evaluation metrics for the purpose of our study: convergence delay, convergence update count, fail-over delay, and the conformance of forwarding paths to routing paths. We define each in turn. *Convergence delay* is the elapsed time between an injected routing change and the finally converged route from the point of observation. *convergence updates count* is the number of updates observed from an observation point given a routing change at a known location and time. *Fail-over delay* is the amount of time it takes for the route to be reachable again from an observation point, given a routing failure. *Conformance between forwarding paths and routing paths* refers to the agreement between the AS-level paths traversed by data packets and the best BGP AS paths in the routing tables. Due to the hop by hop nature of Internet routing, these two paths may not always match.

1.4 Framework for Understanding and Improving BGP Dynamics

We propose a framework as shown in Figure 1.2 to improve our current understanding of BGP routing dynamics by combining measurement, analysis, and correlation with the data plane. We briefly describe the tight coupling of the three major components of our work shown in the Figure. We start with analyzing aspects of the BGP protocol of interest by understanding specifications in detail. Due to lack of completeness of some specifications, we may resort to the actual vendor implementations in a testbed setting to clarify potential ambiguities. Initial insight from the analysis is then confirmed through measurement on the actual Internet using BGP Beacons, an active routing measurement infrastructure. To understand the impact of the particular protocol feature on the data plane. We make use of the AS-traceroute combined Beacons to study the packet forwarding behavior of the Beacon prefixes. The process is iterative, as insight from measurement studies can provide input into analysis for detailed simulation scenarios. Similarly, proposed changes to the protocol may need trace-driven simulation combining measurement and analysis.

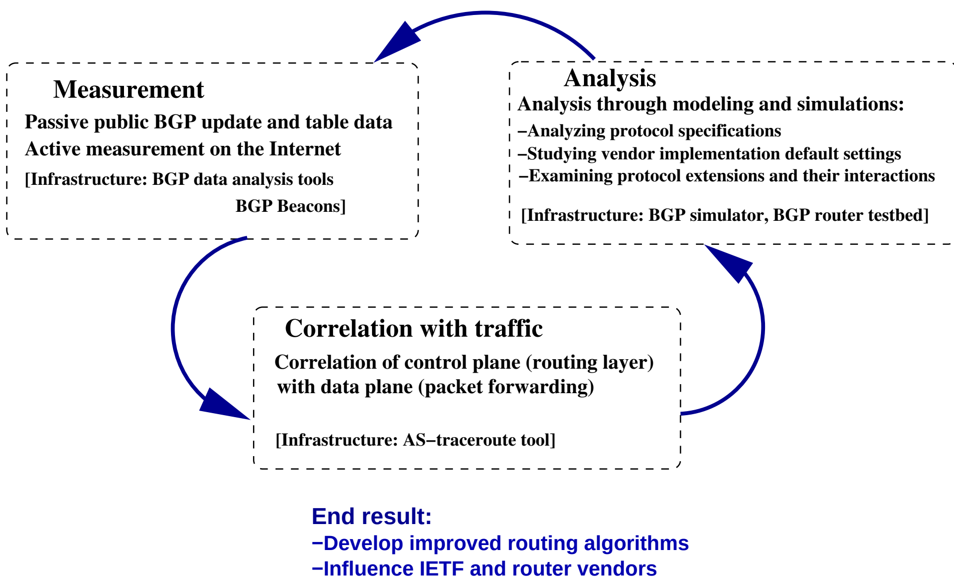


Figure 1.2: Framework for understanding and improving BGP Dynamics

1.4.1 Analysis, Measurement, and Correlation with the Data Plane

- Analysis: The BGP protocol evolves quickly. Many new features are being added over time. For instance, a recent proposal has been to use BGP to provide auto-discovery of Layer-3 VPNs [82] and virtual private LAN services [37]. Thus, an important component of this framework is to analyze new protocol features to understand their interactions. One such example of such analysis is described in Chapter 4. Besides studying the protocol specifications, we also perform analysis of selected scenarios using representative topologies and policies.

- Measurement: Since many factors (*e.g.,* topologies, policies, protocol implementation details) are unknown, we must perform empirical studies to understand how quickly the Internet converges today and the details of implementation variants of router vendors and their influence on convergence. We perform controlled active measurement by injecting routing changes of unused prefixes to make more accurate observations how BGP responds to particular routing changes. We also use a router testbed consisting of real routers to evaluate the current implementation of protocols. Furthermore, we ask feedback from network operators to better understand operational practices and constraints.

- Data plane correlation: If the control plane converges fast, the data plane is not guaranteed to have good performance. It may be counter-intuitive that the forwarding paths of data packets may not exactly follow the routes advertised by BGP due to routing anomalies. Thus, it is also important to correlate the performance between the control and data plane to identify the root causes of data plane performance problems.

1.4.2 An Example Case Study of Identifying Routing Problems

We now illustrate the use of this framework to improve our interpretation of BGP dynamics and identify the source of potential routing problems. We clearly show the benefit of our infrastructure of combining active measurement, analysis and traffic correlation. We reuse the previous simple topology shown in Figure 1.1 in our example. Suppose a user connected to AS5 experiences difficulties reaching a destination in AS1 in the form of a high packet loss rate. The user subsequently calls up its network provider of AS5 to complain. With the state of art of debugging capability, the network provider of AS5 can perform "traceroute" to attempt to reach the affected destination. However, traceroute provides only IP-level paths and cannot be directly correlated with the routing information in the local BGP tables. One could also look into the looking glass or route servers of the ASes 1, 2, 3, and 4 to check the presence of the destination of interest. However, none of these can definitively identify the source of the routing problem if the destination is not black-holed. The debugging process must also test the reachability of the customer prefix in AS5 from AS1, as the reverse path may be the actual problem.

Using our proposed framework, the AS-traceroute tool can provide AS-level forwarding path information for both the forward and reverse path from AS1 to AS5. Such AS-level routing information can be directly correlated with the routing information obtained from BGP route monitors. If there is a discrepancy between the two, a routing anomaly such as deflection and the associated AS responsible for the anomaly can be identified. Furthermore, given the presence of a Beacon in AS1, one can use the active measurement infrastructure to inject a routing update associated with Beacon prefix from AS1 and carefully analyze the observed update sequence. This can reveal the set of alternate paths between AS5 and AS1 and identify why none of them is available. We can perform the same analysis for a Beacon prefix originating from AS5 and observe from AS1's perspective. Our framework provides significantly more information to help diagnose problems associated with routing dynamics by combining active measurement as well as correlation between the forwarding plane and the data plane. In summary, this framework provides insight into the diverse policies used by various networks and helps explain the routing changes observed and its impact on the data plane.

1.4.3 Dissertation Overview

Now we lay out the roadmap of this dissertation. In Chapter 2, we first give background of BGP and then summarize the related work in this area to point out the difference from our work. The related work can be classified into the following categories based on their methodologies and the scale at which the studies are conducted: (i) BGP convergence analysis through measurement and simulations, (ii) Router benchmarking to understand for example how well the router can sustain large routing table input, (iii) formal analysis of protocols to understand policy divergence and configuration of the protocol to prevent routing anomalies, and finally (iv) how to use BGP to improve overlay route selection and detect failures.

Chapter 3 presents our research methodology as summarized above: combining analysis, measurement, and correlation with the data plane. We provide some more details about the tools we developed for the purposes of protocol analysis and measurement. Specifically, we describe a long-lived active measurement infrastructure supporting ongoing BGP research where one can measure the convergence delay of announcement and withdrawal updates, the speed of fail-over for multi-homed customers, and how "happy" the data packets are. Another tool we developed is AS-level traceroute, which is essential for correlating the data plane with the control routing plane by translating the infrastructure IP address to AS numbers. We also describe our extensions to an existing BGP simulator for the purpose of analyzing specific BGP features and how we used a router testbed to confirm some of our analysis results.

Following Chapter 3, we present the three main technical chapters of the thesis. Chapter 4 describes our finding that Route Flap Damping, a mechanism in BGP aimed at achieving routing stability can unexpectedly severely delay convergence for stable routes. We analyze this behavior in detail, and validate our model of this important network phenomenon through simulations and router testbed experiments. We conclude with a proposal of a new route flap damping algorithm that alleviates the interaction between route flap damping and BGP convergence. This chapter illustrates the importance of analysis when new BGP features are added, primarily through the use of simulations and testbed experiments, to provide insights of the protocol dynamic behavior.

Chapter 5 describes the active measurement infrastructure we developed for the ongoing study of BGP dynamics. We explain in the detail how to use the infrastructure in terms of how to preprocess the data to identify the injected routing events. We illustrate the use of the measurement infrastructure by analyzing route flap damping, inter-arrival times, and convergence delays. This chapter illustrates the active measurement component of our research methodology. In particular, we demonstrated its utility by showing how it provides strong evidence that route flap damping suppresses stable routes on the current Internet, a phenomenon we discussed in Chapter 4.

In Chapter 6, we describe the techniques and algorithms behind our AS-level traceroute tool useful for identifying forwarding path anomalies to be correlated with the control plane. Due to operational practices, it is difficult to obtain the exact IP to AS mapping for infrastructure addresses as needed by the tool. To maximize the accuracy of the inference we iteratively apply a dynamic programming algorithm to continuously improve. We demonstrate the usefulness of such a tool by describing the class of routing anomalies the tool can detect.

Chapter 7 concludes with the summary of our contributions and directions for future work. Our main contributions lie in a novel experimental framework for studying BGP routing dynamics. The framework consists of the active Internet-scale measurement infrastructure, simulation and test-bed based experimentation, and correlation between routing and the data plane. We illustrate the use of the framework by discussing in depth a new problem in BGP we discovered: delayed convergence caused by rout flap damping, and subsequently propose a solution to eliminate this undesired interaction. There has been many work in the literature in the area of BGP. In the next chapter, we first provide the background information of BGP and then proceed with an overview of related work while pointing out their difference from our work.

Chapter 2

Background and Related Research

In this chapter, we first provide the background information for BGP by describing its basic operations and emphasize on how routing policies contribute to the complexity of BGP's run-time behavior. We then provide an overview of related work in the area of BGP. The focus of the dissertation is on understanding BGP dynamics, so there are many related works in this area. We do not attempt to cover all work related to BGP, but instead only focus on those most closely relevant to this dissertation. These form the foundation on which the thesis is built on and sets our new work in the appropriate context. The previous studies can be divided into the following main categories based on their methodologies and the scale at which BGP is studied (network-wide vs. router-level scale): (1) Network-wide protocol characterization, (2) Router-level protocol analysis, (3) Internet characterization, (4) BGP-based network diagnosis, (5) Policy analysis and configuration, and (6) Protocol improvement. Network-wide characterization focuses on analyzing data from multiple vantage points to understand the dynamic behavior of routing from a network-wide perspective. In contrast, router-level analysis examines only at a router level the behavior of the router given external input and the detailed implementation decisions and default parameter settings made by router vendors. Internet characterization focuses mostly on topology, policy, and AS relationship inference using snapshots of routing data. BGP-based network diagnosis makes use of BGP information to improve application performance in the context of application overlay routing. Protocol analysis studies how to change the BGP path vector's path exploration aspect to reduce convergence delays. We now turn to the background information of BGP.

2.1 BGP 101: Introduction to BGP

The Border Gateway Protocol, abbreviated as BGP, is a *path-vector* protocol. Path-vector protocol by definition is same as distance vector protocol with additional path information. Routing information in BGP contains the length of the path to the destination, as well as the *actual full path* used to reach the destination. This eliminates the "counting to infinity problem", a well-known problem associated with distance vector protocols due to lack of loop detection when links fail. Loop detection in BGP is possible because the AS path information carried in advertisements stores the list of ASes the route traverses. The specification of BGP in RFC 1771 defining the vanilla base protocol without extensions is surprisingly simple given the important role of BGP and its known difficulties to debug due to lack of global visibility of topology and routing policies. The simplicity in specification is due to the enormous amount of freedom the protocol leaves to the operators in configuring specific policies for choosing the best paths in the BGP decision process. Router vendors in turn provide many knobs for operators to specify routing policies. Below we first describe the basic operation of BGP and subsequently the aspects of BGP dynamics that affects application performance: stability, convergence, and predictability.

2.1.1 Basic Operations of BGP

Internet routing is destination-based and is done on a hop-by-hop basis. The destinations are expressed as network prefixes supporting Classless Interdomain Routing [41]. BGP has two distinct modes of operation: EBGP (External BGP) exchanges reachability information between Autonomous Systems, while Internal BGP (IBGP) exchanges external reachability information within an Autonomous System. Note, IBGP allows internal sources to reach external destinations and vice versa, but it does not maintain reachability within an AS. This is done by IGPs such as OSPF. There is, however, interaction between IBGP (BGP's routing of external routes) and IGP in the form of IBGP's route selection of the ones with the lowest IGP distance given routes exiting through the same egress point or with the same nexthop [54, 78]. Although problems with IBGP also exist as shown by Griffin *et al.* [53], we focus on EBGP, as IBGP in general is simpler and easier to debug given the availability of internal router configuration information within an AS. The dynamic complexity due to inter-AS operation inherently is more complex and interesting to study. In the following discussions, BGP refers to the EBGP mode.

BGP messages are sent using the TCP protocol to ensure reliable delivery. Since TCP itself can operate without exchanging any messages for a long time, BGP uses periodic keep-alive messages between two peers in a BGP session to ensure their liveness. In the case of congestion, dropped keep-alive messages can result in session time-out. The in-order delivery of TCP can exacerbate this [97]. An AS uses BGP to advertise routes to its neighboring domains indicating its routing decisions and its willingness to carry traffic on behalf of others. The essential BGP protocol is based on two types of BGP messages: *announcements* and *withdrawals*.

An announcement message sent from a router R_a in AS A to a router R_b in AS B to the destination prefix P with routing information R indicates that R_a is willing to carry traffic from R_b

destined to P following the route R. The routing information advertised also indicates the routing decisions of the advertiser. In this case, R_a uses the route R to reach destination P. Given all the available routes received from one's neighbors stored in the RIB (Routing Information Base), the router uses its routing policies to select the best route. In this example, R_b may have received multiple routes for destination P, one from each of its providers. It will choose the best one based on its own routing policies and may in turn announce the route to its customers. Routing policies also determine the available routes between a pair of Internet hosts. R_a may decide not to advertise the route to R_b due to its specific policies. Because of the way in which BGP policies filter and propagate routes, physical connectivity on the Internet does not imply logical reachability. Thus, BGP does not ensure global reachability, which is important for assure the ability for any to any communication.

Withdrawals, as expected, indicate that the sender no longer has a route to reach the given destination. They invalidate the last announcement sent regarding the route. This can be caused by things such as transient failures caused by a flapping link, congestion which can cause session time out, and software upgrade on routers which may require reboot. Announcements can also serve as *implicit withdrawals*, modifying the attributes of the route that was last sent.

Local BGP changes in routing decisions are propagated globally and may result in each AS changing its best routes and subsequently propagating updates to its neighbors. This means that local instability or churn can generate global churn. Consider the following example. Assume AS A *originates* the routes to destination P, meaning that A owns the network P and no other ASes can originate P. Assume also that the route to P is not aggregated by any ASes and thus any changes for P will be visible in all default-free routing tables on the Internet. Then in this case, if A flaps the routes to P by continuously sending alternate announcements and withdrawals, these messages will propagate to the entire Internet. All routers with default-free routing tables will respond also with announcement and withdrawal messages. Chapter 4 is devoted to discussing the impact of how local churn affects global routing. The BGP routing dynamics we focus in this work has to do with the reachability of networks, fail-over behavior in case of routing changes, and the forwarding behavior at the AS level.

We now discuss an important property of BGP: BGP is *stateful*, meaning that there are no periodic refreshes of state information and only changes, *i.e.,* deltas of routing information, are exchanged. This requires BGP speakers to retain the entire routing information without expecting any refreshes of routing state from its neighbors. BGP's stateful design is to make it scale to the Internet. To understand the importance of scalability, we cite two numbers. In today's core routers, the default-free (*i.e.,* without any default routes) routing table typically have about 120,000 routes; and there are about 170,000 Autonomous Systems. Consequently, globally propagated route refreshes cannot be allowed. This property of BGP has significant implication on its dynamics: Session resets result in exchange of the entire routing table between the two peers. This effect can propagate to the rest of the Internet.

2.1.2 BGP Routing Policies

Now that we have illustrated the basic operations of BGP, we describe in more detail an important aspect of BGP operations – policy-based routing. The flexibility in defining routing policies and their proprietary nature are two main reasons why BGP is hard to understand. BGP Version 4 has evolved through the demand of operators to achieve certain desired configurations of the networks often based on customers or neighbor networks' demands. An example of such a demand is to enable traffic to be routed using "cold potato" (*i.e.,* carrying the traffic as far as possible before handing off to the neighboring network) leading to the introduction of MED (Multiple Exit Descriminator) attribute. An aspect of BGP that continuously grows in complexity is its flexibility in expressing policies. As mentioned before, ASes have commercial relationships between them that determine routing. There can be dual transit/peer relationships [80] between ASes or relationships on a per destination basis. Such complex relationships require flexible knobs to determine routing decisions. BGP gives each AS enormous freedom in defining semantically rich routing policies specifying which routes are preferred in carrying traffic. Independently defined local policies at an AS level can interact in unexpected ways on the global scale leading to routing anomalies [51].

The simplified BGP route selection process consists of the following six steps.

1. Prefer routes with the highest local preference: The goal is to enforce economic relationships between ASes. Often customer routes are preferred over any other routes; peer routes are preferred over provider routes, because peer routes are "free" while provider routes cost money.. The relationships between ASes are typically customer-provider, peer-peer [24, 42, 101]. These relationships determine whether an AS exports its best routes to its neighboring ASes. For instance, providers provide connectivity to the Internet as a service to their customers. Peers, on the other hand, only provide connectivity to their respective customers. In practice, however, very complex relationships exist, and precisely inferring routing policies is impossible. For instance, relationships can be based on destination prefixes and are not so clearly defined between two ASes, and there can be dual transit/peer relations. Figure 2.1 illustrates one such example where the relationship between AS3 and AS4 is of a dual transit/peer nature. Therefore AS paths such as $[6\ 3\ 4\ 2]$, are also considered legal, *i.e.,* do not violate commercial AS relationships. The typical legal paths consist of an uphill portion (from customers to providers), an optional flat link (between peers), followed by a downhill portion (from providers to customers). For example, one such path is $[6\ 3\ 1\ 2\ 5\ 7]$, where $[6\ 3\ 1]$ is the uphill part, and $[5\ 7]$ is the downhill part.

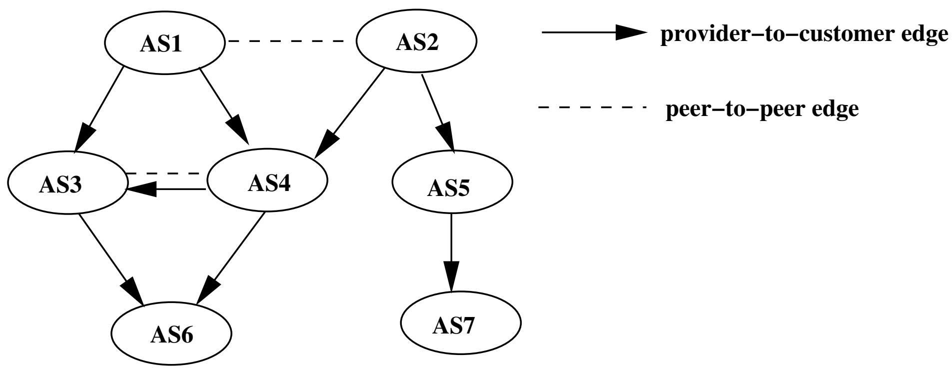


Figure 2.1: Annotated AS relationship graph illustrating complex AS relationships.

2. Prefer routes with the shortest AS path: This is a way for choosing routes with better quality, as BGP does not perform load-sensitive routing globally. However, routes with shortest AS path do not necessarily yield the best performance. Note, this selection criterion comes after the the policy-based selection, as BGP allows policy-based metrics to override distance-based metrics.

3. Prefer routes with the lowest MED (Multiple Exit Descriminator) attribute: This is how BGP implements cold-potato routing with neighboring domains. The MED attribute overrides the next two route selection decisions (for hot-potato routing) in picking routes and can force the domain receiving the MED values to carry more traffic in its own domain before handing it off to the neighboring domain.

4. Prefer EBGP routs to IBGP routes: The motivation behind this rule is to hand off traffic to other networks as early as possible, *i.e.,* perform hot-potato routing.

5. Prefer routes via the nearest IGP neighbor: As mentioned above, this rule together with the rule of preferring EBGP over IBGP routes implements “hot-potato” routing. These are typically useful in transit domains, *e.g.,* tier-1 ISPs like AT&T and Sprint. Furthermore, this is

also how IBGP interacts with IGP. Because of this step, routing to external destinations can be affected by intradomain instability. For instance, if a link flaps causing an egress point to be preferred for a large number of routes, this can result in many EBGP routing advertisements to neighboring domains.

6. Tie-breaking rules (*e.g.,* using router ID): if there exist equivalent routes on the basis of previous criteria, BGP uses this to break ties. Some implementations pick the routes with the lowest routerid [93] while others prefer the oldest routes [32]. Note, if the latter is used, BGP route selection is then non-deterministic and may be difficult to debug. However, oldest routes may be more stable than others.

Given the above process of choosing the best routes, unfortunately, today's mechanisms in defining routing policies are rather primitive and closely resemble assembly-language based programming. The current interface provided by major router vendors such as Cisco and Juniper is rather cumbersome and filled with low-level details. There is no support for network-wide specification of routing policies, as desperately needed by operators configuring networks with hundreds of routers. The mechanisms present in BGP to tweak policies are in the form of import, export policies, route filters, and route maps. The state-of-art tools for configuring networks are error-prone and difficult to reason with at a network-wide level.

A more serious problem of configuring routing policies independently is that the policies defined locally can lead to BGP protocol oscillations that never reach a stable routing. Such protocol divergence has been illustrated by Griffin *et al.* [51] and shown to be difficult to detect statically (with NP-hard complexity). In some cases, guarantees of protocol convergence is shown to be computationally feasible. For instance, to ensure safe backup routing without causing protocol divergence while conforming to AS relationships, Gao *et al.* has proposed a model for backup routing [43]. More recently, Griffin *et al.* has proposed several design principles of policy languages for path vector protocols such as BGP [46] to simplify reasoning of complex policies and enforcement of routing properties.

2.2 Network-wide Protocol Characterization

We have given the background information of BGP. We now describe related work in the area of BGP routing dynamics. There has been many studies to characterize BGP's behavior at the scale of the entire Internet by analyzing BGP routing updates collected either passively or actively. The latter type of updates are generated by injecting controlled routing changes associated with unused prefixes into the Internet. The former type of data are available from projects such as Oregon RouteViews [15] and RIPE NCC [11]. Many networks peer with the routers of these projects to provide default-free routing data for the purpose of easier debugging. Such data contain all changes to the best routes of the routing tables belonging to these networks. Our studies analyze both passively collected data and active measurements. The latter are needed due to the inherent difficulty in interpreting passive measurements.

2.2.1 Passive Data Analysis

In such cases, two types of data are of interest: BGP table dumps and BGP updates. The former provides a snapshot of the routing table in terms of all the best routes. The latter contains a time series of the changes to the best routes. We illustrate an example of how the content of BGP table dump, a BGP announcement update, and withdrawal update looks like below. Each field is separated by "|" in this particular representation. The format is explained directly below each type of entry: table dump, announcement, and withdrawal. A table entry is taken from a snapshot of the best routes in the routing table. Announcements and withdrawals are BGP updates indicating changes to the best routes.

```
Table entry:  TABLE_DUMP|1067715079|B|128.32.1.200|25|3.0.0.0/8|
              2152 3356 7018 80|IGP|137.164.24.105|82|30|
              2152:65489 2152:65497 2152:65499 3356:3 3356:86
              3356:575 3356:666 3356:2011|NAG

Format:       TABLE_DUMP|time stamp|B|peer IP|peer AS|prefix|
              AS path|Origin|Nexthop IP|local preference|MED|
              community|aggregated|aggregator IP|aggregator AS

Announcement: BGP4MP|1067673681|A|128.32.1.200|25|195.80.227.0/24|
              2152 16631 6762 12654|IGP|137.164.24.105|82|30|
              2152:65473 2152:65497 2152:65499 16631:1000|NAG|
              10.0.112.128|64726

Format:       BGP4MP|time stamp|A|peer IP|peer AS|prefix|
              AS path|Origin|Nexthop IP|local preference|MED|
              community|aggregated|aggregator IP|aggregator AS

Withdrawal:   BGP4MP|1067673733|W|128.32.1.200|25|203.10.63.0/24

Format:       BGP4MP|time stamp|A|peer IP|peer AS|prefix
```

We now briefly describe the fields in the three data types. All three entries contain timestamp, prefix, neighbor IP, and AS information. Neighbor IP and AS indicate where the update comes from or the owner of routing table. *Origin* indicates where the route originated from: internally (IGP) or externally (EGP). *Nexthop IP* indicates the IP address of the nexthop router for the route. *Local preference* is the locally defined preference for the route based on local policies; it is the first selection criteria for choosing routes as described in Section 2.1.2. *MED* or Multiple Exit Descriminator, as defined earlier, is used for doing cold-potato routing. *Community* attribute encodes specially defined information for the purpose of grouping prefixes with similar properties. The three fields associated with aggregation: *aggregated*, *aggregator IP*, and *aggregator AS* indicate whether the prefix belongs to a large aggregate or supernet.

A table dump entry is a special announcement with the same attributes. Withdrawal update is much simpler containing only the prefix and neighbor information. Some studies [55] make use of the data to compute general statistics to provide valuable insight in the behavior of BGP over time. Commonly calculated statistics include the growth of routing table size, the number of ASes, and average prefix length. Researchers have also analyzed the data for the purposes of identifying routing anomalies or unexpected routing behavior, *e.g.,* address black-holes [62], routing misconfigurations [70], and Multiple Origin AS (MOAS) conflicts [110]. It is possible to get some basic understanding of BGP dynamics based on BGP updates as shown by numerous studies [20, 30, 36, 94, 108, 109]. Typically, in such studies, analysis focuses either on particular destination prefixes to study the trend of BGP update growth over time, or on a particular time period, *e.g.,* during worm outbreak, to understand how BGP behaved while Internet is under stress. It is in general difficult to infer the *root cause* of an observed BGP update given opaque routing policies and limited visibility into the internal topology [49]. A direct consequence is that passive measurements are insufficient even to measure a very basic metric such as *convergence time* defined to be the amount of time to propagate a routing change. For each update message observed, one does not know the actual input that triggers the update. So, we must resort to active measurement to study BGP convergence behavior.

2.2.2 Active Controlled Measurement

Active measurement has been used in many areas including operating system and architecture, where the experimenters perturb the system under study by injecting an input typically of the form of a probe or a fault. The advantage of such form of measurement is that the experiment is well-controlled, but at the cost of the overhead created. The first set of active BGP measurement studies were performed by Labovitz *et al.* [63, 65–68]. They injected routing changes at exchange points as well as stub networks that multi-home to multiple providers to study fail-over behavior as shown in Figure 2.2, as well as convergence times of routing changes. The routing changes or BGP faults injected are associated with unused prefixes to which they have access at a stub AS to upstream ISP1 and ISP2 shown in the Figure. They subsequently studied the routing changes observe from RouteViews data collection probe consisting of feeds from many ISPs such as ISP3, ISP4, ISP5, and ISP6 in the figure. The collection probe peers with these ISPs to passively obtain changes to the best routes for the routing tables of these ISPs. In general, it is difficult to obtain such an address spaces for experimental purposes, and such experiments often require upstream providers to modify their router configurations to accept and propagate updates associated with these. The overhead associated with extra routing updates due to the injected routing change is typically very minimal, as the Internet has the order of 100,000 prefixes.

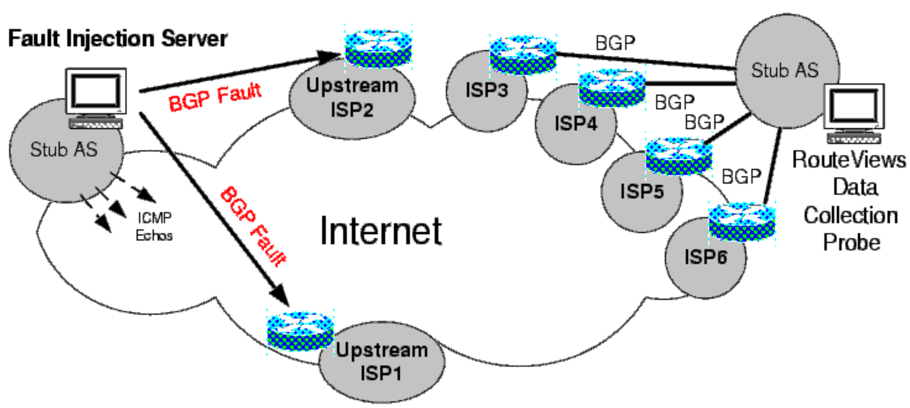


Figure 2.2: Labovitz's Active BGP Measurement Infrastructure – BGP Fault Injectors.

Labovitz *et al.* also performed some "happy" packet measurements associated with routing changes to study the reachability, packet loss rate, and delay to a set of web hosts from the stub networks whose address space experiences routing changes. Pei *et al.* attempted such studies using simulations [89]. One important empirical and analytical observation of Labovitz is that during the convergence process there is a *path exploration* process where alternate routes may be explored before finalizing the best route during the announce and withdrawal processes. Such effects are inevitable, as there is no synchronization between routers across ASes; and thus, less preferred routes may be explored before the ultimately preferred one. As mentioned in Chapter 1, routes with shorter AS paths tend to be preferred. However, such routes may not necessarily propagate faster. Thus, alternate less preferred, and potentially longer routes may still be explored first. Related to the happy packet measurements, there has been much previous work in understanding end-to-end routing behavior [84, 86]; however, they did not attempt to directly correlate with BGP measurement data.

2.2.3 Summary

We now summarize related work in this area and point out the distinction from our work. Passive measurement data are easy to obtain today; however, they are only of limited use and cannot be used to measure the critical metrics of interest to us, like convergence times and the number of updates associated with an input routing change. This is because it is difficult to do root cause analysis or even associate updates with the actual input change. In our work, we also do extensive analysis using passive BGP data to understand BGP dynamics. However, we build an extensive active measurement infrastructure – BGP Beacons – to support active injection of routing changes and accurate association of routing changes with input change. This infrastructure is intended to be public and is supported with long-term commitment. We describe our measurement infrastructure in detail in Chapter 5.

2.3 Router-level Protocol Analysis

Parallel to analyzing protocols at the network level, several studies such as [29, 72, 97] have examined the protocol at the router level to understand the details of the implementation variants of different vendors. Studies at the router level explain the behavior of protocols at the network scale. Examples of such studies include the default setting of timer values, response of routers when given a large input routing table, implementation of timer on a per peer or per destination basis, and application of rate limiting to withdrawals. In the IETF there is working group on Benchmarking Methodology [1] focused on recommending the measurement of the performance characteristics of various internetworking technologies. A draft related to measuring BGP convergence is already available [25]. This establishes the terminology to standardize how to measure EBGP convergence in the control plane of a single BGP router. Commercial products such as Agilent's RouterTester [103] help evaluate routing protocol implementations. This is quite different from network-wide protocol analysis as described above. Our work also falls into this category, as we make use of the Beacons infrastructure to differentiate the BGP signaling path with different types of protocol implementations as the last-hop router. Analysis of vendor implementations are critical to better understanding of protocol dynamics, as the network-wide protocol behavior is heavily influenced by router-level protocol behavior.

2.4 Internet Characterization

Due to the large scale of the Internet, there have been many measurement studies aimed at better understanding characteristics of the Internet. Relevant to Internet routing, researchers so far have focused on discovering Internet topologies, ISPs' policies, and AS relationships. All these are relevant to understanding the BGP dynamics. AS an example of a major Internet mapping effort [13, 45], the Mercator project uses traceroute to discover network topologies. The more recent Rocketfuel project [99] using similar tools but aimed at more efficient discovery with minimal probing. Comparison [96, 102] has also been made between the shortest possible paths based on measurements between intermediate hosts and the actual path taken by BGP. As expected, the end-to-end path taken using the BGP protocol is not necessarily the shortest path.

In general an ISP's routing policies are difficult to infer, and there are few studies that are aimed at understanding or inferring how ISPs configure their networks [69] with no access to the configuration information. Even without knowing the detailed routing policies from ISPs, we can have a rough idea of which routes are preferred if the relationship between ASes are known. The general rule of thumb is that customer routes are preferred over peer routes, which are preferred over provider routes. Work in this area to characterize the Internet hierarchy [24, 42, 101] make the simplified assumption that there is *no* complex relationships such as dual transit/peer between ASes. It is known that such assumptions may not hold in practice. Thus, the relationship inferred even if assumed to be accurate cannot always accurately predict the routing policies. It is inherently difficult to predict BGP route selection accurately in a simulation scenarios based on inferred relationships, and thus measurement is still needed.

2.5 BGP-based Network Diagnosis

In this category of related work, BGP data is used to identify routing anomalies in real-time to proactively perform application-layer mechanisms for improved performance, *e.g.,* reactive routing. Reactive routing employs overlay routing by performing flexible routing at the application layer. Using empirical data, Feamster *et al.* [39] studied how BGP instability correlates with application-level performance and whether BGP messages can be used to predict routing failures to help improve reactive overlay routing. This is the first step towards making use of BGP data to predict failures for overlay routing. However, the study does not evaluate the mechanism in terms of the false positives and negatives of how often BGP is accurate at predicting failures between two overlay nodes. In general, there has not been much work in the area of using BGP update data to predict and improve application performance. This is because it is not easy to interpret observed BGP routing changes and to understand how these affect traffic. Our work provides an active measurement infrastructure that helps calibrate BGP updates and facilitate the interpretation of routing data.

2.6 Policy Analysis and Configuration

Pioneering work by Griffin *et al.* [51] has shown that policies configured independently by ASes can cause oscillations at the network-wide level and never converge to a stable routing. Unfortunately, even if the policies were fully known, the complexity for statically checking policy convergence is NP-hard. Consequently, Griffin speculated that the solution for detection will have to be dynamic. Subsequent work by Griffin *et al.* [47, 48, 53] uses formal analysis to model BGP path selection based on *Simple Path Vector Protocol (SPVP)* that abstracts away all non-essential details. SPVP is used to design a safe BGP that is guaranteed to converge. More recently, Griffin has shown [52] that the MED attribute of BGP can also cause oscillations. Due to complexity of routing policies and the possibility for routing anomalies such as policy divergence, recently researchers are proposing the use configuration automation [28], logic-based verification and policy language-based reasoning [38, 40, 46] to improve the ability to reason BGP dynamics. The work discussed in this dissertation also benefits from such formal reasoning, and the empirical measurement and AS-traceroute tool can aid in identifying routing anomalies.

2.7 Protocol Improvement

In our final category of related work, we discuss work to improve BGP protocols. Researchers have proposed modifications [87, 88] to BGP to improve its convergence times by reducing the effect of path exploration. Such modifications are in general hard to get adopted without a compelling reason. In our work, we also proposed changes to BGP's route flap damping mechanism. However, our proposed change does not require changes to the protocol implementation and can be implemented either by agreeing on a BGP community attribute or by selecting more liberal default flap damping parameter settings.

There are several working groups at IETF proposing modifications to BGP and evaluating changes proposed by researchers. Two most most relevant ones are Inter-Domain Routing (idr) Charter [5] and Global Routing Operations (grow) Charter [3].

2.8 Summary

To summarize, the work in this dissertation encompasses the characterization of BGP dynamics both at the network-wide level, as well at the router-level through simulation, active and passive measurement. And the proposed changes we make to BGP are incrementally deployable. Previous studies have demonstrated the importance of understanding BGP dynamics particularly in the area of routing policy conflicts, where independently selected policies can result in unstable routing globally. Below we list several important contributions from related work and point out the lack of a public, ongoing active measurement infrastructure to allow continuous monitoring and controlled measurement of injected routing changes, as well as correlation between the data plane and the routing plane.

- Network-wide protocol characterization using passive and active data: Previous work showed unexpected long convergence delays and proposed several changes in the protocol to improve convergence. However, there are still a few cases of unexpected long convergence with no known explanations. Our work provides one possible explanation and employs the active measurement infrastructure to validate our conjecture.

- Router-level protocol analysis: Both router vendors and researchers study and test the behavior of protocol implementations by doing testing in a small testbed setting. Previous work has shown the deficiency in routers when given large routing tables. Such scenarios are not uncommon due to misconfiguration of networks leaking a large number of deaggregated prefixes. Our work also includes the component of testbed-based analysis, where we analyze the implementation details and default parameter settings to explain the protocol behavior at the network-wide scale.

- Internet characterization: Related work in this area infers Internet characteristics such as network topologies, ISP routing policies, and AS relationships. Such information is assumed to be fairly stable. It helps explain the dynamic behavior of routing protocols and constitutes an important input to our work as well.

- BGP-based network diagnosis: Previous work in the context of overlay routing makes use of BGP updates to forecast performance degradation at the application layer due to routing failures. Our work examines the correlation between routing and packet forwarding behavior at a fundamental level. Applications depending on the user behavior may not always be affected by all routing changes.

- Policy analysis and configuration: Such analysis can be done statically, especially by individual ISPs who have complete information of policies and network configurations. We are interested in understanding the interaction of policies and impact of configurations at the global Internet scale. Due to lack of information, active measurement is needed to measure the routing behavior.

- Protocol improvement in BGP convergence: In the past, a series of proposals have been made to improve BGP convergence, mostly addressing the path exploration effect of the path vector protocol. Our work shows another reason – route flap damping – that causes delayed convergence.

Past work in the area of BGP clearly illustrates a need for long-term controlled measurement of the protocol behavior at the Internet scale. Our work fills this important gap. Here we have given an overview in this chapter the related work in BGP. In ensuing chapters, we describe in more detail previous work as background information to our work. Next, we turn to a discussion of our research methodology that is unique in our work.

Chapter 3

Methodology

In this chapter, we give the high-level overview of our research methodology as shown in Figure 3.1 used to conduct our studies. It contains an iterative process of analysis, measurement, and traffic correlation. We start with analyzing the protocol by modeling the protocol behavior through

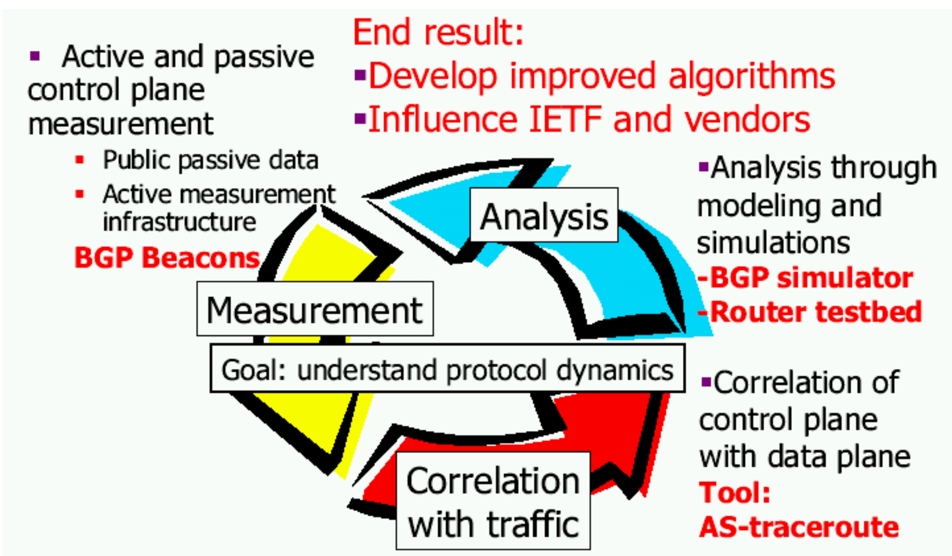


Figure 3.1: An overview of our research methodology

simulations with direct input from a router testbed needed for clarifying ambiguities of specifications. We then use both passively and actively collected measurement data to understand the routing behavior at the Internet scale to confirm conjectures from the analysis. BGP Beacons, our active measurement infrastructure, allow us to inject routing changes of interest. Our AS-traceroute tool then enables us to correlate the observed routing changes with the impact on the traffic. The three components are tightly coupled and provide feedback to each other. For example, measurement reveals more realistic simulation models, and measurement is essential for traffic correlation. Results from the latter can also provide input to analysis for developing more accurate protocol models or reveal potential problems with certain protocol features. In the ensuing chapters, we will describe each component in more detail.

Overall, we combine protocol analysis, simulations, a router testbed, as well as the means for passive and active measurement collection. We describe the tools and software we built for the purpose of studying BGP dynamics. They include an extension to the BGP simulator, software to inject routing changes, AS-traceroute tool for discovering AS-level forwarding paths, and various software for BGP data analysis. Section 3.1 describes the sources of BGP data we use to analyze protocol behavior and the important preprocessing step we perform to assure accurate interpretation of data by minimizing data artifacts. Since passive measurement is insufficient for our study, we also built an active measurement infrastructure which we describe briefly in Section 3.2. As part of the analysis component of our work, we use both a BGP simulator and a router testbed to model and study the protocol dynamics at a relatively small scale and in controlled topologies. These are described in Sections 3.3, 3.4. Finally, as we would like to understand how routing impacts the data plane and ultimately the application perceived performance, we describe a tool we built to achieve such correlation in Section 3.5.

3.1 Routing Data Analysis

Two main public sources of BGP table and update data are Oregon Route Views [15] and RIPE NCC [11]. Both archive daily table dumps as well as NTP-synchronized updates in MRT format – a well-known binary format for archiving BGP data. Such data are often collected by the zebra software router [4], a public GNU based software with several routing functions including OSPF, RIP, and BGP. Zebra conforms to the most recent BGP RFC1771 and supports several BGP extensions. It has been tested to work with several different router vendors with different protocol features and uses the MRT format to store the data.

The Route Views Project peers with around 30 networks mostly within the U.S., but some in Europe. Among them there are many tier-1 ISPs such as Level3, Cable&Wireless, AT&T, Sprint, and Verio. RIPE peers mostly come from Europe. A majority of these feeds are configured to be default-free; thus, one can observe the entire routing table from those view points. The motivation behind the tier-1 ISPs to provide such data feeds is to provide easy data access for debugging routing problems such as reachability. These peering sessions are passive and typically one-directional. The router from RIPE and Route Views (*monitor router*) only receive the updates from the router in different networks (*operator router*), but do not inject any routes in the routing session. However, to be defensive against routing misconfigurations, the monitor should configure the export policy to be a brickwall, *i.e.,* no routes are allowed to be sent out. Similarly, the operator router should configure the brickwall import policy for the peering session associated with the monitor router, *i.e.,* no routes are accepted. The peers simply provide a view of their routing table and do not provide any connectivity; thus, Route Views or RIPE routers cannot use them to forward traffic.

These feeds contain the best routes of the particular routers who peer with RIPE and Route Views. It is a view from a *single* router. ASes can consist of hundreds of routers; thus, these

BGP feeds do not capture the intradomain routing dynamics. Furthermore, other routers within the same AS from which we have a BGP feed may choose a different set of routes as their best paths. This does not necessarily violate the consistency of routing policies within an AS; as such paths can have the same length and consequently the same routing preference value. We need to be careful interpreting the BGP data, as we do not have available all the data from all routers.

Oregon Route Views also provides route flap dampening data, as well as Cisco and Juniper CLI (Command Line Interface) access. We used the route flap dampening data to have an initial estimate of how prevalent route suppression is on today's Internet. CLI access is an easier and more flexible way to obtain the snapshot of the routing table, as the syntax allows one to focus on specific prefixes. However, due to the overhead of displaying an entire default-free routing table, rate-limiting is typically imposed on CLI to frequently access the full table information. CLI also allows us to quickly find information such as the list of peers, and to perform traceroute and ping measurements.

3.1.1 Data Cleaning and Preprocessing

One would assume that the BGP monitoring session's routing updates reflect the actual changes of the best routes of the peer's routing table. However, this may not always be the case due to data artifacts. It is thus important to preprocess the BGP data to remove any artifacts before doing data analysis or drawing any conclusions, as the data obtained from BGP monitoring session may not always indicate *actual* routing changes. For instance, one has to filter out spurious session resets caused monitoring session time out. The monitoring BGP session between the monitor router and the operator router typically are over multiple router hops, as the operator routers are locally far away. Such sessions are aptly named multi-hop BGP sessions. In practice, EBGP sessions are configured over a single hop, *i.e.,* the two peers are directly connected. A multi-hop BGP session is much more prone to congestion effects, leading to loss of keep-alive messages and subsequently BGP session resets. When a BGP session is reset, all the routes previously exchanged are implicitly withdrawn. Thus, session resets are expensive and can create considerable BGP churn which may propagate globally. The session reset of the monitoring session does not affect the global routing system. This does however affect the data collected. It is therefore very important to ignore BGP updates associated with such artificial BGP session resets, as they do not reflect true Internet routing stability. BGP has a type message called STATE message that can be used to identify session resets. Another way of session reset identification is to look for a large number of prefixes being withdrawn and then subsequently re-announced within a short period of time. There may be other artifacts associated with monitoring BGP data. A good way to discover these is to log the status of the monitoring *i.e.,* zebra software itself to identify such artifacts.

It is highly valuable to combine data from multiple sources to correlate BGP routing changes with the source of the routing problems. This depends on the different sources of data to have relatively good clock synchronization. Thus, another aspect of data cleaning involves the identification of clock drifts, which is harder. When we correlate data from multiple monitors, *e.g.,* RIPE data with Route Views data, we depend on the time stamps of the BGP data being fairly accurate. One simple sanity check we perform is look for any incidents of data sequence where the clock seems to go backwards. This would be a clear indication that the clock was reset. There is no guarantee that the clock is well-synchronized even if we do not detect such incidents. Analyzing data from a single source, for example Route Views data from all its peers, can give precise information on the relative convergence delay without depending on clocks being synchronized. We discuss these issues in Chapter 5.

3.2 Measurement Infrastructure

Passive measurement data are prevalent and they provide useful input to construct AS topologies for simulation purposes. They also provide a first-order analysis of the health of Internet routing, as problems such as blackholes, address deaggregation, and obvious policy violations can be easily identified. We always combine data from *multiple vantage points* as they provide more rich and complete views than data from a single location. However, to test out hypothesis of our work and to analyze in detail routing convergence delays, we need an infrastructure to perform controlled active measurement, so that we can clearly identify updates associated with a single input routing change. For this purpose, we constructed an active BGP measurement infrastructure [72] deployed at several locations on the Internet. Figure 3.2 shows one of the Beacons announced from a stub network and observed from all places where BGP data is publicly available.

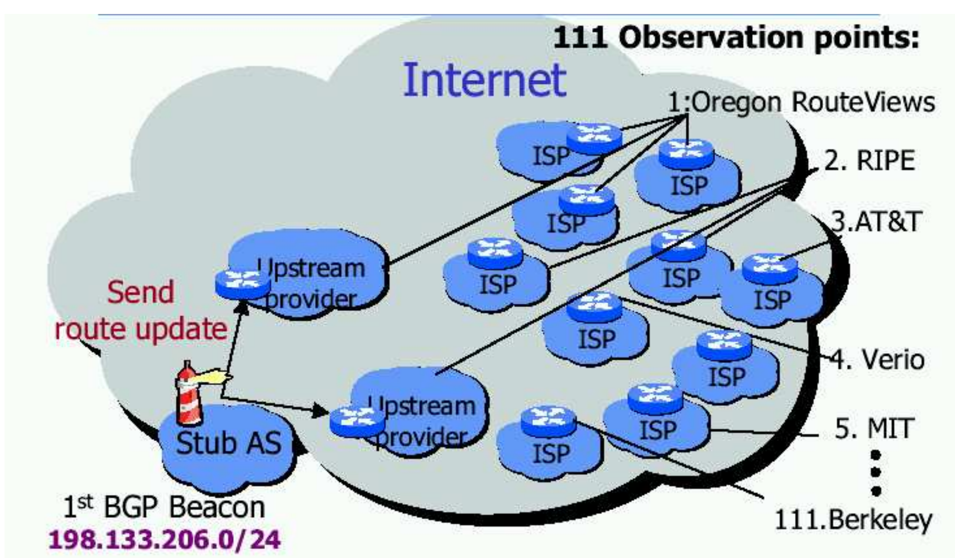


Figure 3.2: BGP Beacons: active measurement infrastructure

We instrumented a software router that peers with the stub AS router to allow us to inject specially constructed routing announcements encoding timestamp and sequence number information for ease of data correlation. Knowing the exact time at which the routing change is injected, we can obtain metrics such as convergence delays, convergence update count, and update inter-arrival times that characterize the BGP dynamics. It also gives us insight into routing policies used by various networks from the transient updates observed, which usually indicate backup routes. Details of the measurement infrastructure and methods to analyze the data are described in Chapter 5. The software custom written to facilitate the injection of routing updates are publicly available at `http://www.psg.com/~zmao/Software`. It has been tested to work in various intra-AS topologies, *e.g.,* AS confederation and route reflector set up, and shown to be compatible with several different router vendors.

3.3 BGP Simulator

Measurements on the Internet allow us to study the current routing behavior. Simulation is very helpful to analyze the impact of particular parameter settings, implementation variants of protocols, or the effect of topologies. We incorporate realistic topology models from measurement studies and understanding of vendor implementation details from the router testbed study in Section 3.4. For example, we simulate BGP protocol in topologies that exacerbate BGP's path exploration effect to study the BGP convergence behavior. Such topologies typically have multiple alternate paths of different AS path lengths, and the different alternate paths are mostly independent or do not share many common ASes. One example of such topologies is clique. We make use of a publicly available BGP simulator, SSFnet [14]. It is a Java-based simulation package capable of modeling large, complex networks. It has a BGP simulator conforming to the RFC 1771. We implemented route flap damping in SSFnet conforming to RFC 2439 to study the effect of flap damping on stable routes. We provide several parameter settings used as default settings by major router vendors. The extension has been integrated with the latest release of SSFnet v1.5 on May 8, 2003.

3.4 Router Testbed

Although the BGP simulator conforms to RFC 1771, it only implements one possible interpretation, as many details are left unspecified or are recommendations only. Furthermore, understanding details of vendor implementations can help us build more realistic simulation models. In our work, to test the impact of vendor implementations, we set up a small router testbed of both Cisco and Juniper routers with the help from Sprint Research Lab and AT&T Labs-Research to verify the assumptions in the simulator. This was an extremely useful exercise, as the BGP simulator may deviate from actual router implementations widely deployed on the Internet. For the purpose of our study, we validated that the route flap damping implementation on commercial routers keeps route penalty on a per prefix and per neighbor basis, and as long as any update attribute differs, the routes are considered different contributing to penalty increase. Furthermore, we validated the default flap damping settings as published by vendor specifications.

3.5 AS-traceroute Tool

To correlate data traffic with routing traffic, we need a tool similar to traceroute but one that provides AS-level rather than IP-level information, as BGP data are expressed in ASes. We developed such a tool available now at `http://www.research.att.com/~jiawang/as_traceroute/`. We heavily make use of BGP update and table data and IP-level forwarding paths obtained using traceroute from multiple vantage points. The assumption we make is that most of the discrepancies between AS paths from BGP tables and AS paths from traceroute stem from inaccuracy of the IP-to-AS mapping. This tool can be used along with the active measurement infrastructure to study packet forwarding paths during injected routing changes.

This chapter has provided an overview of our research methodology of combining measurement, analysis, and correlation with traffic to improve our understanding of BGP dynamics. In the next chapter, we provide a detailed example of where analysis of the BGP protocol combined with simulations and a router testbed discovers an important phenomenon in BGP dynamics: Route flap damping, a mechanisms designed to achieve stability can significantly delay routing convergence. Later in Chapter 5, we demonstrate the validity of our discovery by showing instances of such occurrences on the Internet with the help of the measurement infrastructure.

Chapter 4

Route Flap Damping Exacerbates Delayed Internet Convergence

In this chapter we describe the interaction between route flap damping and BGP's path-vector convergence process. We are the first to fully study this unexpected interaction: Flap damping can severely delay routing convergence, because temporary instability due to routing convergence causes more updates than expected and the default damping parameter settings by major router vendors today are excessively aggressive. The problem is detected using both simulations and a router testbed, and later confirmed through controlled active measurement on the Internet. We propose a solution to eliminate this interaction by modifying the protocol to encode route preference. Our solution is evaluated in detail using simulations and analysis. In Chapter 5, we use BGP Beacons to calibrate the updates to understand their root causes and validate that route flap damping can indeed delay convergence of stable routes on today's Internet. The phenomenon described in this chapter serves as a motivating example of complex BGP dynamics where various BGP features interact in unexpected ways. In this case, the path-vector nature of BGP inherently explores a set of alternate paths during convergence, and depending on the topology, the number of paths explored can be large enough to trigger route suppression. This work also illustrates the success of our methodology of combining analysis and measurement to detect, analyze and correct a problem in BGP dynamics. It also shows the importance of understanding real-world BGP implementations to understand its dynamics.

This chapter is organized as follows. First, we give an overview of the problem associated with the tradeoff between routing stability and fast convergence in Section 4.1. This tradeoff is inherent in any dynamic system. Section 4.2 then introduces route flap damping–a widely deployed BGP mechanism, explains its algorithm based on exponential decay of penalty values. It highlights the difficulty of debugging reachability problems due to the lack of feedback in flap damping and its different parameter settings. We examine in Section 4.3 several examples of how damping is invoked by a *single* announcement or withdrawal using small artificial topologies. The general property of such topologies is that there exists multiple alternate paths explored typically of different AS path lengths. Such property is not uncommon in Internet topologies due ot prevalence of backup paths. In Sections 4.4 and 4.5 we use a simple analytical model to explain the phenomenon and use extensive simulations to understand the impact of various BGP features have on this interaction. We found that none of the features can completely eliminate this undesired interaction. To have an initial estimate of how likely this interaction occurs on the Internet, we did some trace analysis of BGP update sequences in Section 4.6. We observe thousands of update sequences each day covering a wide range of prefixes from many BGP peers that indicate this interaction. Finally in Section 4.7 we present our proposal to eliminate this problem in BGP dynamics through an improved flap damping algorithm evaluated both analytically and using simulations. Section 4.8 summarizes the key results of this chapter: (1) The unexpected interaction between route flap damping and path-vector convergence discovered though analysis and measurement is a clear example of the complex

BGP dynamics. (2) We propose an improved flap damping algorithm to eliminate this interaction and validate it through simulations and analysis.

4.1 Introduction

Routing mechanisms that trade-off route convergence or optimality for increased stability are often described in the routing literature. One such instance is the experience with load-based routing in the old ARPAnet, where routing system stability was achieved only by significantly *damping* link metrics [61]. Similarly, Cisco and Juniper deliberately delay route calculations in IS-IS [81] implementations to increase stability [27]. A second instance is the default setting of keepalive or Hello timers in intra-domain routing protocols. Existing implementations use fairly conservative values for these timers, resulting in slower detection of link state changes and consequently less routing update traffic [27]. In this chapter, we analyze a third instance, BGP *route flap damping*.

Route flap damping is a mechanism designed to selectively limit the propagation of unstable routing information [106]. works as follows. Each BGP-speaking router maintains a route *penalty* associated with every prefix announced by each BGP neighbor. This route penalty increments by some fixed value whenever the state of the route changes and exponentially decays with time. In effect, the penalty measures the instability of a route. The router uses locally configured thresholds to decide when to *suppress* the route (*i.e.,* not use the route because it is unstable) and when to subsequently *reuse* the route.

Originally proposed in the early days of the commercial Internet, route flap damping is generally assumed by network operators to be widely deployed in today's infrastructure [2, 56]. Furthermore, it is widely held to be one of the main contributors to the overall stability of the Internet inter-domain routing system [56]. However, there have been no rigorous studies to quantify the extent of deployment of route flap damping, nor any studies to quantify the impact route flap damping has on the stability of the Internet.

While the original target of route flap damping was route flaps caused either by router mis- or re-configuration, or by chronically unstable links, the mechanism can prevent the widespread propagation of other kinds of routing pathologies. These include persistent route oscillations caused by mutually incompatible policies [51], route changes resulting from the repeated BGP connection tear-down and re-establishment that has been known to occur as a result of incompatible implementations.

However, as we show in this work, route flap damping can actually *exacerbate the convergence of relatively stable routing information, sometimes by up to an hour*. The intuition for this comes from the work of Labovitz *et al.* [63], who showed that a single route withdrawal can result in other routers exploring a sequence of alternate paths before deciding that the destinations is unreachable. We show that this kind of exploration causes what we call *secondary flaps* that can trigger the suppression threshold of the route flap damping algorithm. This prevents the widespread propagation of a subsequent route announcement, resulting in the delayed convergence of the route. We describe this phenomenon – *withdrawal triggered suppression* – in greater detail in Section 4.3.

We conjecture that withdrawal triggered suppression explains the tail of the convergence distribution from the experiments of Labovitz *et al.* [63, 68]. Even though their experiments injected route changes roughly once every two hours (and therefore should not have triggered route flap damping), they found that routes took nearly fifteen minutes to converge (a time constant that is consistent with at least one set of route flap damping parameter values [83]). Autonomous Systems are multi-homed today [56], one can expect greater levels of path exploration, resulting in greater likelihood of route suppression.

In addition to describing the withdrawal triggered suppression, we gain insight into the phenomenon both through analysis (Section 4.4) and simulation (Section 4.5). Analysis characterizes the progress of secondary flaps and their impact on convergence in simple topologies. Simulation in SSFNet [14] studies how, if at all, various proposed BGP features (such as sender-side loop detection and withdrawal rate-limiting [63]) impact withdrawal triggered suppression. To our surprise, topologies with more alternate paths do not necessarily have a greater likelihood of exhibiting withdrawal triggered suppression. We also find that in some topologies, sender-side loop detection is effective in eliminating this phenomenon. In Section 4.6 we analyze real traces to show that such flaps that can cause long convergence delays occur frequently.

Finally, we evaluate the effectiveness of a simple modification to route flap damping called *selective flap damping*. It eliminates withdrawal triggered suppression in all the topologies we studied (Section 4.7). The key new idea is to ignore monotonic route changes (as is typical in path explorations after failure) as flap damping triggers. Section 4.8 concludes the chapter, which is mainly an example of how our research methodology is used to detect, analyze, and remedy a problem in BGP dynamics.

4.2 Background

This chapter investigates the interaction between route flap damping and BGP convergence. It provides insights into problematic BGP dynamics and suggests the need for better visibility into BGP update propagation through the use of an active measurement infrastructure with controlled routing update injection.

Route flap damping has received very little examination in the research literature, as BGP dynamics is difficult to study in practice due to lack of measurement infrastructure and heterogeneous implementations. In the standards world, there are two documents most often referenced in connection with route flap damping. The route flap damping standard [106] describes the rationale for route flap damping and outlines a possible implementation strategy for the mechanism. While that document discusses some interactions between flap damping and topology, it does not discuss announcement or withdrawal triggered suppression. An associated document, the RIPE recommendations [83] tantalizingly hints that one or both of these phenomena may have been observed in practice. To quote

> ...The only explanation would be that the multiple interconnections between Ebone/AS1755 and ICM/AS1800 did multiply the flaps (advertisements/withdrawals arrived time-shifted at ICM routers through the multiple circuits). ....This would then potentially hold true for any meshed topology because of the propagation delays of advertisements/withdrawals.

However, it then proposes a solution that we do not believe addresses the problem, nor does it analyze the phenomenon in any level of detail.

Also related to route flap damping is a technique for damping link state changes. Rodeheffer *et al.* [95] proposed a filter, called a skeptic, that penalizes unstable link state information for a time that increases logarithmically with the number of flaps of the link state. The details of the algorithm are different from route flap damping, and it would be interesting to compare how the two perform on various kinds of flaps.

In the academic community, there have been two threads of prior research into the following properties of BGP: stability and convergence delays.

Stability: The first thread started with the observation that there existed certain policy configurations which could cause persistent route oscillations in BGP [105]. Later, Griffin and Wilfong [51] showed the intractability of determining a safe policy configuration for BGP. Finally, Rexford and Gao [44] proved that if BGP's policy expressiveness is confined to a simple set of policies, persistent route oscillations cannot occur. Independently, Labovitz *et al.* [66] showed that instability could occur even without policy conflicts because of implementation artifacts. Thus this first thread confirmed the value of the route flap damping standard and probably influenced the RIPE recommendations.

Convergence Delays: The second thread of BGP research is a careful analysis of the dynamics of BGP's route convergence properties [63, 68] and resulted in the interesting finding that BGP's route withdrawal process could result in a combinatorially large number of path explorations.

Thus our work can be considered to be a convergence between these two threads of research because it shows that the RFD mechanism used to improve stability can exacerbate convergence delays.

Other more recent prior work has explored and attempted to solve delayed Internet routing convergence. Griffin *et al.* [50] explored how convergence is affected by the MRAI timer setting and addressed its impact on various topologies. In their future work, they pointed out the potential for route flap damping to be invoked by oscillations inherent in the BGP protocol. In this work, we confirm their suspicion by thoroughly studying its interaction with convergence.

More directly related is the work of Pei *et al.* [87] who attempted to avoid path exploration during route withdrawal by using consistency assertions. They showed that their approach can invalidate all paths within one MRAI round in some cases. This is an intriguing approach that might work, although it needs extensive experimentation to be widely deployed and does not work in all cases (e.g., when policy is used for say traffic engineering). It should be clear from this work that a fix for withdrawal path exploration in BGP will reduce the occurrence of the phenomenon we see in this work. Despite this, the value of our work is that it provides the first analysis of the interaction between RFD and convergence and suggests an alternate solution for this interaction that is useful, if a general solution to eliminate withdrawal path exploration turns out to be hard to design and deploy.

Route flap damping, which we abbreviate as *RFD*, was designed and deployed on the

Internet in the mid 1990s, primarily in response to frequent route flapping. This phenomenon, usually thought to be caused by router re-configuration or by links with intermittent connectivity, manifests itself as frequent BGP route changes. Each such route change causes route recomputation and increases the computation load on the route processor. At the time when RFD was deployed, route processors were less powerful than they are today, and its deployment led to a significantly more stable routing system.

RFD has been shown to be effective in ameliorating the effects of routing instabilities other than those for which it was originally designed. One kind of routing instability is that resulting from the repeated tear-down and re-establishment of a BGP peering session (a peering session flap) that was a hallmark of some early BGP implementations. Peering session flap occurs when these BGP implementations receive BGP routing tables that exceeded the router's memory or receive an incorrectly formulated BGP update. Such flaps can result in frequent route changes for a large collection of routes. RFD can suppress these until the peering flap is resolved by operator intervention. Implementations have now been largely fixed to avoid peering session flaps, but route flap damping remains an important safeguard against future implementation errors that lead to large-scale repeated propagation of routing information. A second kind of routing instability that RFD can[1] suppress are persistent route oscillations caused by mutually conflicting routing policies [51]. These oscillations manifest themselves at a router as repeated route changes. RFD can significantly reduce the frequency of these oscillations.

Today, route flap damping is widely regarded as an important contributor to the overall stability of the Internet routing system by the operator community. To quote Geoff Huston [56]:

> . . . coupled with widespread adoption of BGP route flap damping, has been very effective in reducing the short-term instability in the routing space.

[1] In theory at least. The authors are unaware of actual observations of this kind of routing instability.

In what follows, we describe the route flap damping mechanism in some detail. To do this, it helps to have a simple model of the way a BGP router processes routing information. We describe a simplification of the route processing model in the BGP RFC [92]. Each BGP router has several *peers* (neighbors) from each it receives *routes* to IP address prefixes over a transport connection. Conceptually, routes received from each peer are stored in a peer-specific database called the *Adj-RIB-In*. For a given prefix, the router's BGP decision process computes the most preferred route to the prefix from all the Adj-RIB-Ins and stores it in the *Loc-RIB*. The decision process then determines what subset of the *Loc-RIB* should be advertised to each peer. This subset is stored in a per-peer database called *Adj-RIB-Out* and advertised to the peer.

An important feature of BGP implementations is a hold-down timer on routes advertised to peers. This timer, called the `MinRouteAdvertisement` timer (or MRAI timer as defined in [50]) has a default value of 30 seconds. After a route to a prefix has just been advertised to a peer, subsequent changes to the route are held down until the MRAI timer expires (some vendors implement MRAI on a per-peer, rather than a per-route basis [63]). In doing so, the MRAI timer reduces routing instability during route convergence. As Labovitz *et al.* have shown, it also qualitatively affects the convergence process by limiting the exploration of alternate routes after route withdrawal.

While the MRAI timer was designed to reduce route changes during convergence, it clearly cannot suppress route instabilities caused by extraneous factors (such as unstable links) that cause flaps on larger time scales. Route flap damping was designed for this and works as follows. For each prefix P and for each peer or neighbor N, a BGP router maintains a penalty $p_{[P,N]}$. The penalty changes according to two simple rules:

- Whenever a peer N's route to prefix P changes (either the route transitions from being available to being unavailable, vice versa, or from one route to a better route, or vice versa), the router increments $p_{[P,N]}$. This increment is fixed, dependent on the type of the change.

- $p_{[P,N]}$ decays exponentially with time according to the equation

$$p_{[P,N]}(t') = p_{[P,N]}(t)e^{-\lambda(t'-t)} \tag{4.1}$$

where λ is a configurable parameter.

Intuitively, the penalty maintains an exponentially decaying instability history of a particular route from a particular peer.

When a router receives a route from N to prefix P, it first updates the penalty $p_{[P,N]}$ according to the rules described above. It then determines whether $p_{[P,N]}$ has crossed a configurable threshold, called the *suppression threshold*. If so, it marks the route as suppressed and inserts it into N's Adj-RIB-In. Suppressed routes are not used to compute the Loc-RIB. When it marks a route as suppressed, it also sets a timer for the time at which the current penalty would decay to below a *reuse threshold*. If the route's state changes before the reuse timer expires, the router cancels the reuse timer, recomputes the penalty, and starts a new reuse timer. When the reuse timer expires, the BGP decision process is invoked to compute the new best route to the prefix. Based on the default Cisco parameter setting (Table 4.1), Figure 4.1 pictorially depicts a route's penalty as a function of time and the times at which the route is suppressed and reused for a route that flaps three times with a 2 minute interval. In this case, the route flaps is suppressed for more than 28 minutes.

A typical implementation of route flap damping supports several parameters, all of which are in principle configurable:

- A value of λ, usually expressed using a *half-life* parameter H – the time for the penalty to

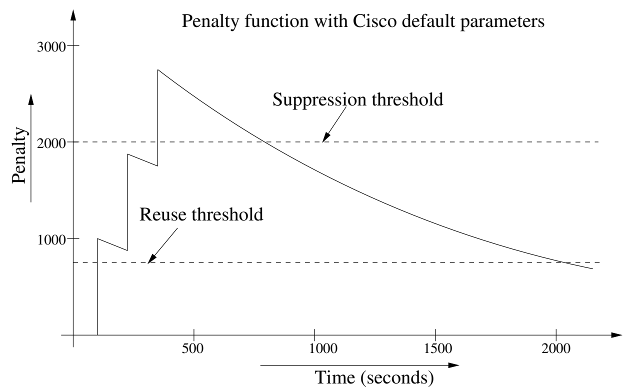


Figure 4.1: RFD penalty function with Cisco default parameters

decay to half its value.[2]

- A suppression threshold, which is the value of the penalty above which the route is suppressed.

- A reuse threshold, which is the value below which the route is considered reusable.

In addition to the above, implementations also have a parameter that limits the duration a route is suppressed. This is achieved either using a configurable maximum penalty or a configurable maximum suppress time. Some implementations also support different penalty increments for route withdrawals, route readvertisements, and route attribute[3] changes.

Despite the richness of the parameter set, deployment experience has shown that connectivity problems can be hard to debug if different routers use different sets of RFD parameters [106]. This is a typical problem in BGP, as implementation variants can differ in both default settings as well as in detailed algorithms. Consider the case where a customer's upstream provider is multihomed and the provider's backup path applies less aggressive damping than the primary path. In

[2]Using Equation 4.1, we can obtain λ from H using the equation $e^{-(\lambda H)} = 0.5$

[3]Recall that BGP routes carry several attributes, the AS path being one of them.

Table 4.1: Default route flap damping parameter settings

RFD parameter	Cisco	Juniper
Withdrawal penalty	1000	1000
Readvertisement penalty	**0**	**1000**
Attributes change penalty	500	500
Cutoff threshold	**2000**	**3000**
Half-life (min)	15	15
Reuse threshold	750	750
Max suppress time (min)	60	60

this case, when the customer's route flaps, traffic to the customer might flow in through the upstream provider's backup path which does not suppress the customer's route, even when the primary path is available.

For this reason, the operator community has recommended a standard set of flap damping parameters [83]. Three salient features of this recommendation are worth pointing out. First, the recommendation calls for different parameter sets for different prefix lengths, a recommendation called "progressive" flap damping. The intuition behind this is simply that smaller prefix lengths should be less aggressively suppressed because they represent a larger address space. Second, to prevent route suppression of relatively stable routes, it specifies that route should not be dampened until at least the fourth flap. Third, the recommended parameters are fairly aggressive. Even the least aggressive parameter set, governing prefixes of length 20 and lower, has a minimum outage time of 10 minutes and a maximum of 30 minutes. Longer prefixes can be suppressed for up to an hour if they flap at least four times.

It is not clear to what extent the recommendations for the flap damping parameters are followed by operators. We note that different vendors have different default parameters (Table 4.1), and we suspect that most ISPs simply use these parameters.

Stage	Time	Routing Tables	Messages Processed	Messages Queued in System
0	N/A	steady state		steady state
		2(*1, 31, 41, 51) 3(21, *1, 41, 51) 4(21, 31, *1, 51) 5(21, 31, 41, *1)		
1	N/A	1 withdraws the route		
		2(-, *31, 41, 51) 3(*21, -, 41, 51) 4(*21, 31, -, 51) 5(*21, 31, 41, -)	1→{2,3,4,5}W	2→{1,3,4,5} [231], 3→{1,2,4,5} [321], 4→{1,2,3,5} [421], **5→{1,2,3,4,X} [521]**
2	N/A	announcement from 2		
		2(-, *31, 41, 51) 3(-, -, *41, 51) 4(231, *31, -, 51) 5(231, *31, 41, -)	2→{1,3,4,5} [231]	3→{1,2,4,5} [321], 4→{1,2,3,5} [421], 5→{1,2,3,4,X} [521]
3	N/A	announcement from 3		
		2(-, -, *41, 51) 3(-, -, *41, 51) 4(231, 321, -, *51) 5(231, 321, *41, -)	3→{1,2,4,5} [321]	4→{1,2,3,5} [421], 5→{1,2,3,4,X} [521]
4	N/A	announcement from 4		
		2(-, -, -, *51) 3(-, -, 421, *51) 4(231, 321, -, *51) 5(*231, 321, 421, -)	4→{1,2,3,5} [421]	5→{1,2,3,4,X} [521]
MRAI timer expires				
5	30	announcement from 5		
		2(-, -, -, -) 3(-, -, *421, 521) 4(*231, 321, -, 521) 5(*231, 321, 421, -)	5→{1,2,3,4,X} [521]	2→{1,3,4,5} W, 3→{1,2,4,5} [3421], 4→{1,2,3,5} [4231],**5→{1,2,3,4,X} [5231]**
6	N/A	withdrawal from 2		
		2(-, -, -, -) 3(-, -, *421, 521) 4(-, *321, -, 521) 5(-, *321, 421, -)	2→{1,3,4,5} W	3→{1,2,4,5} [3421], 4→{1,2,3,5} [4231], 5→{1,2,3,4,X} [5231]
7	N/A	announcement from 3		
		2(-, -, -, -) 3(-, -, *421, 521) 4(-, -, -, *521) 5(-, 3421, *421, -)	3→{1,2,4,5} [3421]	4→{1,2,3,5} [4231], 5→{1,2,3,4,X} [5231]
8	N/A	announcement from 4		
		2(-, -, -, -) 3(-, -, -, *521) 4(-, -, -, *521) 5(-, *3421, 4231, -)	4→{1,2,3,5} [4231]	5→{1,2,3,4,X} [5231]
MRAI timer expires				
9	60	announcement from 5		
		2(-, -, -, -) 3(-, -, -, -) 4(-, -, -, *5231) 5(-, *3421, 4231, -)	5→{1,2,3,4,X} [5231]	3→{1,2,4,5} W, 4→{1,2,3,5} [45231], **5→{1,2,3,4,X} [53421]**
10	N/A	withdrawal from 3		
		2(-, -, -, -) 3(-, -, -, -) 4(-, -, -, *5231) 5(-, -, *4231, -)	3→{1,2,4,5} W	4→{1,2,3,5} [45231], 5→{1,2,3,4,X} [53421]
11	N/A	announcement from 4		
		2(-, -, -, -) 3(-, -, -, -) 4(-, -, -, *5231) 5(-, -, -, -)	4→{1,2,3,5} [45231]	5→{1,2,3,4,X} [53421], **5→{1,2,3,4,X} W**
12	N/A	announcement from 5		
		2(-, -, -, -) 3(-, -, -, -) 4(-, -, -, -) 5(-, -, -, -)	5→{1,2,3,4,X} [53421]	5→{1,2,3,4,X} W, 4→{1,2,3,5} W

Table 4.2: Example of withdrawal triggered suppression in a 5-node clique

4.3 Withdrawal and Announcement Triggered Suppression

Route flap damping was designed to limit the propagation of unstable routing information. In this section, we show by working through two simple topologies that route flap damping can actually suppress relatively stable information. In particular, a single announcement of a route or a single withdrawal of a route followed by an announcement can cause route penalties to accumulate beyond the suppression threshold, causing the route to be suppressed. We call the former *announcement triggered suppression*, and the latter *withdrawal triggered suppression*.

For simplicity of exposition, we assume the following BGP model: (a) Route selection is based shortest AS paths. In case of ties, the route starting with the lower router ID is chosen. (b) The MRAI timer is 30 seconds and only applies to route announcements not withdrawals as recommended by the BGP RFC [92]. (c) No sender-side loop detection (SSLD) is used.[4] (d) Message propagation and processing delay are both bounded and negligible relative to the MRAI value. (e) We show how route suppression can occur for both the Cisco and Juniper parameters in Table 4.1 even when following the RIPE recommendation [83] of not suppressing a route until at least four flaps are received. In Section 4.5 we explore how variations on this model impact withdrawal triggered suppression.

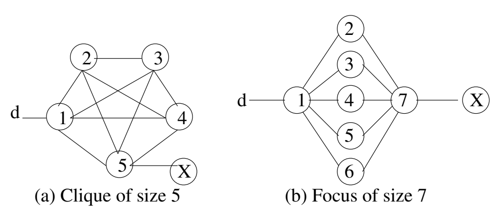


Figure 4.2: 5-node clique and 7-node focus: node 1 announces route to d, route changes are observed at node X.

[4] At the time of this writing, at least one major router vendor does not yet implement SSLD. In Section 4.5, we also show that even with SSLD enabled, withdrawal triggered suppression can happen.

4.3.1 Withdrawal Triggered Suppression

To illustrate withdrawal triggered suppression, we use a clique of size 5, shown in Figure 4.2(a). The clique topology is a canonical topology that has been used to explain pathological route convergence in BGP [63]. Note, we have verified the occurrence of withdrawal triggered suppression in a 4-node clique in a testbed for both Cisco and Juniper routers with default parameter settings [74]. In Section 4.5, we show that withdrawal triggered suppression is not unique to the clique, but the extent to which it occurs does depend on the topology.

Our example from Table 4.2 starts at the point after node 1 has announced a route to destination d, and all nodes have reached steady state. We now show if node 1 flaps *just twice*, by first withdrawing and then re-announcing the route to d, node X will suppress the route. Table 4.2 illustrates the convergence process corresponding to a single route withdrawal by node 1, following the notation in [63]. Each stage denotes the processing of a single set of messages from a node to all its peers. The “Routing Table” column shows the state of routing tables of nodes 2, 3, 4, and 5. The active route is denoted with an asterisk, and an invalid path with a dash. Thus, $4(231, *31, -, 51)$ means that node 4 currently uses route $[3\ 1]$ and has a backup route going through nodes 2 and 5. As an example, in stage 1 node 2 sends the route $[2\ 3\ 1]$ to its neighbors. When this message is processed in stage 2, node 3 realizes that this route goes through itself and so records the route from node 2 as invalid, and switches to the route from node 4.[5]

The “Message Processed” column shows the message processed at a given step, and the messages waiting to be processed are indicated in the last column. Messages from each peer are processed in the order they are received; messages from different peers can be processed in any

[5]The reader may wonder why this problem cannot be entirely avoided by simply invalidating all routes that contain a node i when node i sends a withdrawal. For instance, in stage 1, when node 2 receives a withdrawal from node 1, it seems intuitive to invalidate the routes $[3\ 1]$ and $[4\ 1]$ as well. Sadly, this is not possible in general because policies may require invalidating direct routes without invalidating indirect routes. This is the basis of a recent proposal [87], but it does not eliminate such path explorations due to withdrawals caused by policy changes.

order. We use $i \rightarrow \{j_1 \ldots j_n\}[path]$ to describe that node i sends to nodes $j_1 \ldots j_n$ a route of the ASpath, $path$. Withdrawal is indicated by W.

Consider the messages sent by node 5 to node X (indicated in Table 4.2 in bold font). Four messages are received by X (three announcements and one withdrawal), which account for four flaps. At X, the penalty value associated with the route to d is slightly less than 2500, depending on the precise message propagation delays. Using Cisco's setting, the penalty already exceeds the suppression threshold–2000, causing route suppression. For Juniper's setting, the subsequent announcement by node 1 accounts for another flap, causing the penalty to be close to 3500, also exceeding the suppression threshold–3000. And since X can only reach d through 5, its connectivity is affected because of route flap damping! In our example, it takes *at least 15 minutes* for the route to be restored.[6]

Note that the batching effect of the MRAI timer improves the convergence time in this example by preventing extra updates. For example, when node 3 gets the announcement from node 2 in stage 2, node 3 switches to [4 1] but cannot announce it till the timer expires. But before this happens, node 3 changes its route again to [5 1] in stage 4.

This example illustrates an interaction, which has not been previously well studied, between two BGP mechanisms: the route withdrawal process that has been shown [63] to involve path exploration of successively increasing lengths (in cliques with no policy) and the mechanism to ensure the stability of the overall infrastructure. The rest of the chapter is devoted to analyzing this interaction in detail for various topologies and BGP configuration settings and to evaluating a possible solution.

[6]The penalty value is above 2000 and it has to decay to 750 before the route can be re-used. This requires that that penalty be halved at least once. Since the half life time is 15 minutes, the route is suppressed for at least 15 minutes.

4.3.2 Announcement Triggered Suppression

A companion phenomenon is announcement triggered suppression. We show that in some topologies, a *single* route announcement can result in the route being suppressed at some node in the topology.

Consider the so-called *focus* [50] topology of size 7, shown in Figure 4.2(b). We use the same set of assumptions as in the clique case, except that instead of withdrawing the route, node 1 announces a new route to all its peers. In this case, node 7 has five routes to d of ASpath length 2. Suppose 7 prefers routes going through larger router IDs. Suppose also that the route announcements to node 7 arrive in the following order: $[2\ 1], [3\ 1], [4\ 1], [5\ 1], [6\ 1]$, separated by time intervals at least as large as the MRAI value. This means that node 7 will also announce to X these five routes in the order they are received, because the succeeding route is always preferred over the preceding one. By a similar argument to the above, when node X receives five announcements in sequence, it suppresses the route to d.

In this work, we do not explore announcement triggered suppression further. Its very occurrence depends on topology and very precise timing of update propagation. We believe it is unlikely to occur *frequently* in practice.[7] Withdrawal triggered suppression, on the other hand, depends less on precise timing, and therefore is more likely to occur. Thus, we explore the latter phenomenon exhaustively in this work.

4.4 A Simple Analytical Model

In this section, we explore route flap damping in an n-node clique (Figure 4.2(a)) using a simple analytical model. Our goal is to predict the minimum clique size for which withdrawal

[7] We validated our conjecture that announcement triggered suppression is less frequent by studying BGP update traces (Section 4.6).

triggered suppression can be consistently observed.[8] We analytically evaluate the route penalty in the clique as a function of time, $p(t)$.

Suppose that $p(0) = 0$, and that the route penalty increment is 1. We assume a simplified BGP model in which each node processes messages in lock-step order. That is, at each time step, every node processes all the routes received from all its neighbors in the previous step, selects its best route, and re-advertises that route to all its neighbors. This model approximates BGP processing where each time tick corresponds to one MRAI time interval. Labovitz *et al.* showed that in this model, at least $(n-1)$ steps are needed for the clique, before the route is withdrawn [63].

Consider a node X attached to some clique node i. We compute the penalty $p(t)$ for route d announced by i to X. Now, by our model above, at each time tick node i in the clique picks a new route and advertises it. Thus, at each time tick, node X's penalty progressively increases. To compute the penalty function, we can use simple induction. Clearly $p(1) = 1$; at $t = 1$, node X receives a new route from i and increments its penalty by 1. Then, $p(2) = e^{-\lambda} + 1$; in one unit of time, the previous penalty has decayed to $e^{-\lambda}$, and at $t = 2$ node X receives a single route. By the same logic, $p(3) = e^{-\lambda}(e^{-\lambda} + 1) + 1$, or, simplifying the expression, $p(3) = e^{-2\lambda} + e^{-\lambda} + 1$. This suggests that the general form of $p(t)$ is a geometric series:

$$p(t) = \sum_{j=1}^{t} e^{-\lambda(j-1)} \tag{4.2}$$

and a closed form for this is

$$p(t) = \frac{1 - e^{-\lambda t}}{1 - e^{-\lambda}} \tag{4.3}$$

For what value of t does $p(t)$ exceed the suppression threshold? Suppose we assume that the suppression threshold is 4, and at least 4 flaps are needed to suppress the route. Also, suppose that the half-life time H is 15 minutes (Table 4.1) and the MRAI timer is 30 seconds. Recall that

[8] It turns out that message reordering can increase the number of messages exchanged and increase the likelihood of route suppression.

in our model, one tick of time corresponds to one round of the MRAI timer; in those terms, H is 30 time ticks in our model. Now, recall that λ is the solution to the equation $e^{-\lambda H} = 0.5$; thus $\lambda = \ln(2)/H$. With our choice of parameters, then $\lambda = 0.0231$. Solving numerically, we find that the smallest value of t for which the inequalities $p(t) > (4 - 1)$ and $t >= (4 - 1)$ hold is $t = 4$. Note, we subtract 1 from the suppression threshold and maximum flap count, since the withdrawal at the end of path exploration also accounts for the additional flap with penalty of 1.[9] We also know that after the $(n - 1)$'th MRAI round, each node receives the longest path in the clique, which will cause it (at the next computation step) to withdraw that route [63]. Thus, to explore four MRAI rounds, we need a clique of size at least $five$. Hence, the smallest clique in which withdrawal triggers suppression is a clique of size five.

4.5 Simulation

While our analytic results give us some intuition for the interaction between route flap damping and BGP convergence, they cannot reveal the subtle variations that may arise from differences in BGP features (such as sender-side loop detection), topology effects, or from variations in message propagation latency. Simulation gives us more insight into the conditions under which withdrawal triggers suppression. In this section, we discuss results obtained using the SSFNet simulator [14], a Java-based simulation package with a built-in BGP simulator. The SSFNet BGP implementation is compliant with the BGP-4 specification in RFC 1771 [92]. We implemented route flap damping in SSFNet in compliance with RFC 2439 [106].

[9] Here we assume that between the last update and the final withdrawal, the penalty has not yet decreased significantly. In practice, the re-announcement of the route by node 1 after the path exploration, *i.e.,* an additional flap, will usually keep the penalty value above the suppression threshold.

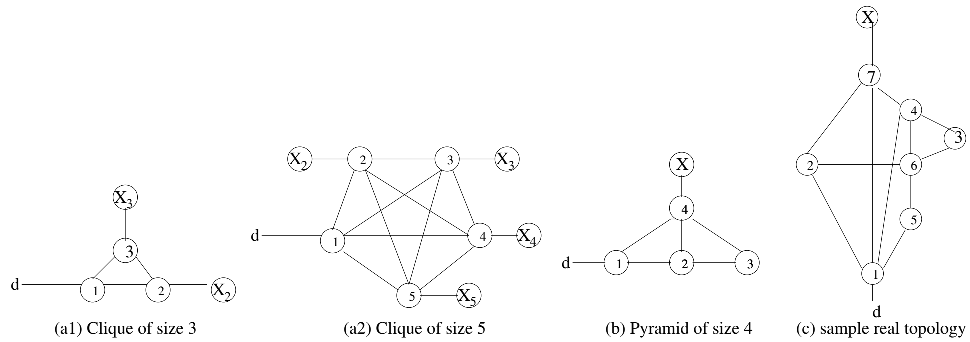


Figure 4.3: Sample topologies used in simulations

4.5.1 Simulation Methodology and Assumptions

Our simulations explore a number of scenarios with different topologies (Section 4.5.2). For tractability, we study withdrawal triggered suppression for a single prefix. In all our topologies (Figure 4.3), the origin d for this prefix is connected to node 1, and we study convergence of the route to d at another node X. In our experiments, d and X are connected by a single link to the rest of the topology. For ease of explanation, we assume that node d is always connected to node 1 in the topology. This simplification allows us to isolate the effect of the particular topology under study on the convergence times at X for routes to d.

Our simulation scenarios ignore route filtering due to policy. Certainly, for a given topology, route filtering can determine whether or not route flap damping is invoked by withdrawal path explorations. Labovitz *et al.* have already shown that there exist realistic policy and topology configurations in the Internet that exhibit delayed convergence [68]. We believe that, in these topologies as well, withdrawal triggered suppression can occur.

Our simulation scenarios treat individual nodes as routers. Withdrawal triggered suppression can occur among routers connected to an exchange point. More generally, it can also occur across multiple autonomous systems. In this setting, our simulations are admittedly unrealistic because they do not capture the internal topologies of ASes. However, we believe our conclusions will not be qualitatively affected by this simplification, since route flap damping is not invoked on I-BGP peering sessions. This precautionary measure prevents inconsistent routing and forwarding loops within an AS [59].

Unless otherwise specified, we study the following route change pattern in all our simulation scenarios. Node 1 announces a route to d at some time to all its neighbors. All nodes in the topology have converged to a route to d by some time t. At time t, node 1 detects a failure of the link to d and withdraws its route to d.[10] Then at time $t + \alpha$, node 1 re-announces the route to d to all its neighbors, because the transient failure has been repaired.

The choice of α affects whether withdrawal triggered suppression happens or not. If α is large enough, of course, the route penalties accumulated at the nodes as a result of the route withdrawal will have decayed below the reuse threshold. As a result, when it is re-announced, all nodes will converge relatively quickly to their route to d. Clearly, the largest value of α for which this happens depends on the topology and flap damping parameter setting. We have verified these qualitative observations for a clique topology of size 5 and for the base parameter set (described in the next section). We found that when α is greater than 1600 seconds, withdrawal triggered suppression does not occur in that topology. If α is smaller than the MRAI value, the withdrawal followed by the re-announcement will be aggregated by the MRAI timer, and withdrawal triggered suppression will not be invoked. We have also verified this in our simulator. In our simulations, we set α to 500 seconds; this is large enough for all topologies in our study to have converged after the withdrawal at time t.

In all our simulations, the link delay is set to be 0.01 seconds. Since only a single desti-

[10] This is a simplification. The exact mechanism by which this failure is detected depends on protocol details. For example, if node d and 1 are external-BGP peers, this detection might happen because the BGP keepalive timer expires. If, instead, d is internal to node 1's AS, the failure may be detected by the failure to receive IGP Hellos from d.

nation prefix is simulated, router workload variation is simulated using variable delay in processing updates. This delay varies uniformly from 0.01 to 1 second. In addition to this source of randomness, jitter is applied to MRAI, as suggested by RFC 1771 [92]. Each data point in our simulation results is obtained by averaging a number of simulation trials.

4.5.2 Simulation Scenarios and Metrics

The occurrence of withdrawal triggered suppression depends on topology as well as parameter settings for various BGP mechanisms. This section describes the topologies and parameter settings explored in this work.

We use the topologies shown in Figure 4.3 in our simulations. Our goal is not to enumerate all the topologies for which route flap damping can exacerbate convergence. Rather, we study this effect for very different topologies to see if there is any qualitative difference in the interaction between RFD and convergence. We also include one real topology fragment studied in the literature [68] to demonstrate that the effect can be observed in practice.

Our topologies include (Figure 4.3):

- An n-node clique. The clique has been used in the literature as a canonical topology to understand withdrawal path explorations. Furthermore, cliques are not completely unrealistic topologies. Full mesh BGP peering at exchange points does occur. Whether the routing policies at these exchanges cause these path explorations is not clear.

- An n-node pyramid. This consists of $n-1$ nodes, numbered 1 through $n-1$ connected in a chain. Node n is directly connected to each one of the other nodes. The pyramid is a contrived topology. But, we chose the pyramid because it is a qualitatively different topology from the clique. The clique is highly symmetric in that every node is connected to every other node. The pyramid is highly asymmetric, with only n being connected to every other node, and all other nodes having relatively sparse connectivity. Moreover, the pyramid is a topology where we might *expect* withdrawal triggered suppression: node n has $n-1$ alternate paths of different lengths to d, a property that has been shown to be at least one signature of topologies in which withdrawal path exploration can happen [63].

- A sample topology from a study done by Labovitz *et al.* [68]. This topology is a subgraph of the inter-AS topology that was actually observed in their experiments. We include this topology to show that withdrawal triggered suppression can occur in real topologies as well.

In addition to the topology, withdrawal triggered suppression depends on the parameter settings for route flap damping. It also depends on the configuration of two features in BGP implementations:

- Sender-side loop detection (SSLD): a BGP speaker avoids announcing routes to a peer if that peer would detect a loop in the route and discard it. SSLD has been shown to improve route convergence in many cases.

- Rate-limiting applied to withdrawals (WRATE): some implementations apply the MRAI timer to route withdrawals as well as updates, violating a recommendation of the specification.[11]

There is a third BGP implementation feature that can affect our findings. Some implementations set MRAI timers *per peer* instead of *per prefix*. This can reduce the likelihood of withdrawal triggered suppression by delaying announcement messages to peers. But, this in combination with

[11] At the time of this writing, at least one major router vendor applies rate-limiting to withdrawals.

WRATE can also further delay withdrawal messages, resulting in additional alternate paths explored, increasing the likelihood of triggering route suppression. We have left the study of this feature for future work since it required simulation of other prefixes in the system.

To understand whether and how these BGP features affect our findings, we explore the following sets of parameters:

Base case: This uses a "standard" set of parameters. MRAI timer of 30 seconds, no sender side loop detection, no withdrawal rate-limiting, no policies, and route flap damping are implemented at all nodes. This case uses the Cisco parameter set in the first column of Table 4.1, along with RIPE's recommendation of not suppressing until at least the fourth flap. The results using the Juniper parameter set are similar.

MRAI=5: This set is used to study the impact of MRAI on withdrawal triggered suppression. Here, we set MRAI to 5 secs, keeping all other parameters unchanged from the base case.

Less aggressive damping: We set the penalty increment for route attribute changes to be 250 (half the value in the base case, see Table 4.1), but keep other parameters unchanged. This penalizes route attribute changes less, and in this sense is less aggressive.

SSLD: In this set, we enable sender-side loop detection. All other parameters match the base case.

WRATE: In this set, we enable withdrawal rate-limiting, keeping all other parameters of the base case.

Damping disabled: Finally, we disable route flap damping in the base case. This parameter set is included for calibrating withdrawal triggered suppression.

The primary metric for our simulations is *convergence time*, as defined in Chapter 1 Section 1.3. We repeat the definition here. This is defined as the time between when the route to d is re-announced by node 1 till the time the node marked X sees a usable route to d. In each of the topologies depicted in Figure 4.3 except the clique, node X is always connected to the node n in an n-node topology. In the clique case, we connect a node X_i to each node i in the clique except node 1. We record the longest convergence time among all nodes X_i for each simulation run.

The secondary metric is the *total update count* previously defined in Section 1.3. This is the number of update messages seen in the topology during the entire process including the initial route announcement, withdrawal, and final announcement by node 1. It helps us explain the convergence time behavior in some cases. One may argue that we should also consider instability as a metric, since RFD is aimed at reducing routing instability. However, in our experiments, we control the route changes originated at the source: only a single withdrawal followed by one announcement. We study the routing convergence behavior for such a relatively stable route.

4.5.3 Simulation Results

In this section, we examine the convergence time behavior of different topologies in some detail. This discussion also tells us how different parameters impact withdrawal triggered suppression.

Clique

Figure 4.4 plots the convergence time as a function of clique size, averaging 50 simulation runs. The most startling observation is that, with a *single withdrawal and announcement* from node 1, withdrawal triggered suppression can cause convergence times of up to 60 minutes (3600 seconds) for a large enough clique using our base parameter set. In the “damping disabled” case, by contrast, it takes less than 30 seconds between when the route is re-announced and when the route becomes available at each X_i connected to the clique.

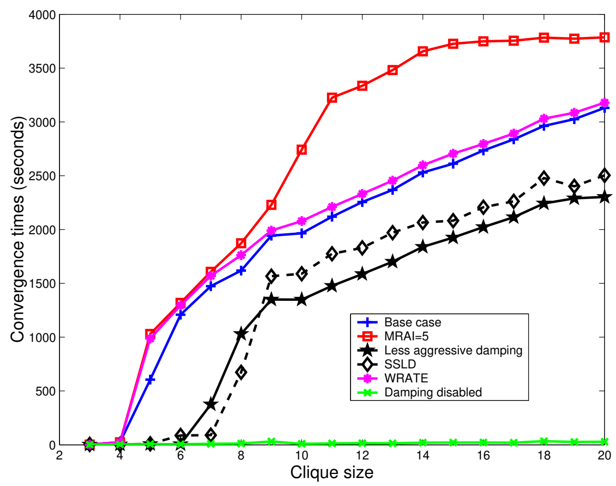


Figure 4.4: Convergence times of the clique topology

Before we analyze Figure 4.4 in any detail, we discuss some subtle but important observations about route flap damping in the clique that are not easily learned without simulation.

Damping in Cliques: The first aspect of damping in cliques is *where* in the clique withdrawal triggered suppression is invoked. Recall that with route flap damping, suppression is per-peer. Each node in a clique is connected to every other node, but in the base case we find two interesting effects: (1) Some nodes do not suppress routes from any peer. (2) No node suppresses routes from all peers. In particular, since node 1 flaps only twice,[12] and all other nodes are connected to node 1, none of them suppresses node 1. Thus, when node 1 re-announces the route to d, all nodes in the clique have at least one usable route to d. But we also observed that it is not true that these nodes suppress all other neighbors either. This is a little surprising, because, from symmetry, one would have expected uniform behavior from all nodes except perhaps node 1. The reason is that in the base case, each node sends the same message to all its neighbors. However, each message is interpreted differently due to loop detection. Some updates are counted as withdrawals because

[12]Using Cisco's parameter set, node 1 only flaps once–the subsequent re-announcement after the withdrawal is not counted as a flap. Using Juniper's parameter set, it flaps twice.

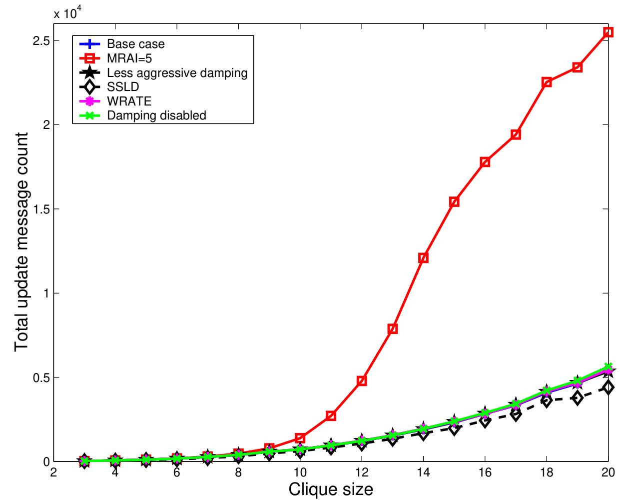


Figure 4.5: Total update count of the clique topology

the receiving node detects a loop in the ASpath. The second of two successive withdrawals is not counted as a flap. Therefore, the penalty values of different nodes accumulate differently with time. Furthermore, jitter added to the MRAI timer as well as router processing times can cause messages become reordered, resulting in different penalty values. This causes different nodes to advertise and receive routes at slightly different times. As a result, routes aggregate or "bunch" up differently. Sometimes a routing update from farther away reaches a node faster than a routing update from its neighbor.

Despite this, a node X_i that is connected to clique node i almost always (beyond cliques of a certain size) observes enough route changes that it suppresses routes from i. Thus, withdrawal triggered suppression does not manifest itself in the loss of connectivity to d from nodes in the clique, but only in nodes attached to the clique.

We also found that variable message processing and propagation delays can unexpectedly cause withdrawal triggered suppression in even a 3-node clique (Figure 4.3). This is in apparent contradiction to our results in Section 4.4, but only because our analytical model did not capture variations in message processing and propagation times. Assume that in the steady state, node X_2 has the route $[2\ 1]$ to d. When node 1 sends out a withdrawal, node X_2 first receives a withdrawal, then an alternate route $[2\ 3\ 1]$ from 2 before the final withdrawal is received. Thus, a single withdrawal results in three flaps. Now, when node 1 announces route to d again to node 2 and 3, due to variable message processing and propagation delay, node 2 sometimes announces route $[2\ 3\ 1]$ to node X_2 before announcing the preferred route $[2\ 1]$. Thus, a single announcement results in two more messages. Node X_2 thus receives a total of 5 messages from node 2, accumulating enough penalty to suppress the route from node 2.

Analysis of Results: Figure 4.4 plots the convergence time for each of our six scenarios as a function of clique size. We now discuss each scenario separately.

Base case: For the base case, withdrawal triggered suppression sets in with a five node clique, confirming our analysis of Section 4.4. This is not surprising, since four messages are required to exceed the threshold. In fact, we find from our simulations that flap damping is triggered at at least one of the X_i's in every simulation run of our five node clique. The convergence time increases monotonically as a function of clique size. The number of paths explored increases with clique size and therefore the accumulated penalty increases. As a result, for large enough cliques, convergence time increases until the maximum suppression time, which in our simulations is one hour (3600 seconds).[13]

MRAI=5: Figure 4.4 shows that compared to the base case, setting MRAI to be 5 seconds consistently increases the convergence times. Griffin and Premore have previously shown that reducing the MRAI timer value can result in many more routing updates [50]. Our simulations also

[13]The convergence time can be a little higher than 3600 seconds, as shown in MRAI=5 case, since we measure the convergence time from when the announcement was sent. The route flap damping suppress timer is set some time after that.

confirm this (Figure 4.5). In turn, this can greatly increase the route flap penalty accumulated for each peer, and thereby the time to reuse the route. We also note that except for this scenario, the number of update messages exchanged is roughly equal for all other cases.

Less aggressive damping: Unlike decreasing the MRAI timer, this scenario exhibits a *later onset* of withdrawal triggered suppression and a lower convergence time. This scenario penalizes route attribute changes (*i.e.,* when a new route differs from the previous route only in the route attributes) by only half the regular penalty. This kind of change predominates during routing convergence. As a result, the penalty accumulates slower than in the base case. Because the thresholds are unchanged, the convergence times are lower corresponding to lower penalty values. Moreover, it takes a larger topology with more alternative routes to trigger route suppression.

SSLD: Sender-side loop detection (SSLD) consistently reduces convergence times compared to the base case. As with less aggressive damping, it also exhibits a later onset of damping. Intuitively, SSLD withdraws invalid alternate paths early and reduces the number of paths explored. This is confirmed by the update message plot (Figure 4.5), showing fewer number of updates. Fewer messages correspond to lower penalty values and thus faster convergence times.

WRATE: As suggested by Labovitz *et al.*, rate-limiting withdrawals can increase convergence times, since it delays the invalidation of invalid alternate paths [63]. More alternate paths are explored as a result, causing higher penalty values and thus longer convergence times. This is evident from our simulation results as well.

In summary, we observe two qualitative classes of behavior with respect to the BGP knobs we study in this section. One class is comparable to, or worse than, our base case. The second class exhibits lower convergence times and later onset of damping as a function of clique size. However, even in the second category, the convergence times are much higher compared to the "damping disabled" case. For a clique of size 10, convergence times are more than 33 minutes. Thus, none of the BGP knobs eliminate withdrawal triggered suppression.

Pyramid

Having examined the clique, we now turn our attention to the pyramid. Recall that we chose to experiment with the pyramid because it was qualitatively different from the clique. Indeed the pyramid reveals significantly different behavior from the clique for many of our scenarios.

Figure 4.6 shows the convergence times for the base case scenario of the pyramid. These times were obtained by averaging 300 simulation runs for different sizes of pyramids. With increasing topology size, the convergence time increases and, beyond a pyramid of size seven, drops dramatically. In fact, beyond a pyramid of twelve nodes, we see almost no evidence of withdrawal triggered suppression. This is very counter-intuitive. We had assumed that since this kind of suppression was caused by BGP's exploration of different path lengths, it would be more prevalent in topologies with larger numbers of alternate paths of different lengths. In a pyramid of size n, node n has $n-1$ alternative paths of lengths from 2 to n. Thus, we expected to see monotonically increasing convergence times with the pyramid, as we did with the clique.

Non-Monotonicity in Convergence Times Explained: To understand this, consider the base case for an n-node pyramid. We evaluate the conditions that must hold for the *minimal* set of route changes to trigger flap damping at node X. We then show that this minimal set of route changes becomes increasingly unlikely due to increased message processing load on node n as the size of the pyramid increases. Note, there is one major difference between the pyramid and the clique. Although both have a large number of alternate paths of different lengths from node n to 1, all these paths in the pyramid are dependent, *i.e.,* they share common hops.

According to our parameters, to suppress a route to d, X must receive at least four route

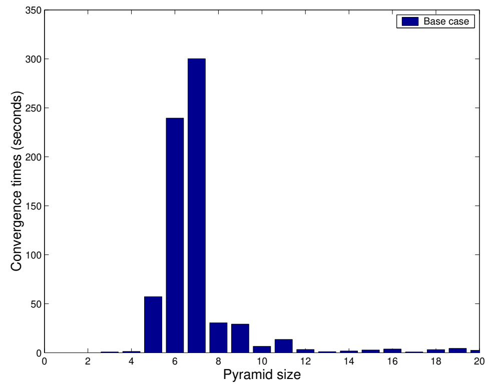


Figure 4.6: Convergence times of the pyramid topology (base case)

changes from node n. If we assume that the re-announcement of the route to d does not itself cause secondary flaps[14], the minimal set of routes needed to trigger a route change is as follows. In response to the withdrawal of the route to d, node n picks two alternate routes to d before withdrawing. These account for three flaps. The re-announcement of the route causes the fourth flap. Thus, the key to our explanation is understanding the circumstances under which node n *twice* announces an alternate route in response to a route withdrawal.

In steady state, all nodes i $(3 \ldots n-1)$ choose the shortest path by going through n: $[i\ n\ 1]$.[15] Now suppose node 1 sends a withdrawal to its neighbors 2 and n. When node n first receives the route withdrawal, it picks the next shortest route $r_n = [n\ 2\ 1]$ and announces it to X. This accounts for the first flap. Assuming comparable route propagation delays to node 2, at roughly the same time, node 2 picks its next shortest path $r_2 = [2\ n\ 1]$. Clearly, node n's choice and node 2's choice are mutually incompatible, so node n will never pick node 2's route. So, if node n has to pick a second alternate route (to account for the second flap), node 3 must choose route $r_3 = [3\ 2\ 1]$,

[14]This is the common case in our simulations, as we rarely observe announcement triggered flaps for the pyramid.

[15]Actually, node 3 can pick either the direct path $[3\ 2\ 1]$ or the path $[3\ n\ 1]$, since they are each of the same length. Here we assume node 3 picks the latter. If it picks the former, n will never explore a second alternate route. That is because 3 will only announce a route change to n, either $[3\ 2\ n\ 1]$ or $[3\ n\ 2\ 1]$, which arrives before n can send out $[3\ 2\ 1]$. In our simulation, the tie-break rules were such that for our topologies, node 3 chooses $[3\ 2\ 1]$ over $[3\ n\ 1]$.

because all other alternate routes go though this route. We discovered that whether node 3 chooses route r_3 is highly dependent on both the message processing delay and the message arrival order of r_n and r_2. Recall that these two routes are sent out roughly simultaneously in response to the withdrawal sent out by node 1. Normally the message processing order does not matter as MRAI imposes an order by preventing messages being sent out before timer expires. However, in this case, node 3 has not sent out any message within the last MRAI time period and can send out an update right away in response to any route change.

The necessary and sufficient condition for node 3 to choose r_3 is that it receives r_n and announces its own choice of r_3 to node n, before receiving r_2 from node 2, and 3 does not announce another route to n before n's MRAI timer expires. We sketch a simple argument for this statement here. It is easy to see that the condition is sufficient: if that is the order of events, then n will select $[n\ 3\ 2\ 1]$ and that constitutes the second flap we have been looking for! This condition is also necessary, because if r_2 is received before node 3 processes r_n, then it can never pick $[3\ 2\ 1]$ and its only alternate route is through node n. In that case, node n will not incur a second flap to trigger flap damping at node X.

It is not completely implausible for r_n to arrive at node 3 before r_2 does, since the path lengths are equal. Thus, whether r_n arrives before r_2 depends on the order in which they are sent out, and the message processing delay by nodes 2 and n. In addition, it also depends on the propagation delay (in our simulations, propagation delay is kept constant). Finally, it depends on whether node 3 processes and sends out r_3 before processing r_2. If it waits, the arrival of r_2 may invalidate r_3.[16] In our simulations, we add a randomly chosen jitter value between 0.01 to 1 seconds for processing each update message. This explains why for larger pyramids, withdrawal triggered suppression is

[16]Note, r_3 does not have to be physically sent out immediately, it can be placed in the waiting queue pending on the value of MRAI, as long as the arrival of r_2 does not cause the message to be deleted from the queue.

less likely to occur. Larger sizes imply that node n is connected to more nodes, and it will take n much longer to process the announcement r_n to be sent to all other nodes. Therefore, the probability of r_n arriving before r_2 is significantly lower compared to smaller topologies. We have confirmed this explanation in our simulation results.

Table 4.3: 6-node pyramid convergence behavior

Parameter setting	Convergence time (second)	Update count	Damp count
Base case	239.57	93	53
MRAI=5	528.22	98	78
Less aggressive damping	195.18	92	35
SSLD	0.77	59	0
WRATE	238.51	94	34
Damping disabled	0.80	93	0

Examining Other Scenarios: Given our observations above, we now examine the impact of the various BGP knobs on withdrawal triggered suppression in a six-node pyramid (Table 4.3), averaging 200 simulation runs. The damp count column indicates the number of simulation runs in which withdrawal triggered suppression occurred. We notice two main differences in convergence times when compared to the behavior of the clique: (1) Sender-side loop detection completely eliminates convergence-based suppression in the 6-node pyramid! We verified that it actually does so for all other pyramid sizes for which suppression is invoked in the base case. (2) Unlike for the clique, withdrawal rate-limiting actually exhibits lower convergence time than the base case. We explain these differences below.

SSLD: SSLD is very effective for the pyramid, because it invalidates all alternate routes within a single round of the MRAI timer. We show such an example for a 4-node pyramid in Table 4.4. When node 1 withdraws the route to d, node 2 picks the alternate route $[2\ n\ 1]$, but does not propagate it to n because it notices a loop. Similarly, n picks $[n\ 2\ 1]$ and does not propagate

Table 4.4: 4-node pyramid convergence behavior with SSLD

Stage	Routing Tables	Msg Processed	Msg Queued
0	steady state		steady state
	2(*1, 341, 41)		
	3(21, -, *41)		
	4(*1, 21, -)		
1	1 withdraws route		
	2(-, 341, *41)	1→{2,4}W	4→{1,3}[421]
	3(21, -, *41)		4→2 W, 2→4 W
	4(-, *21, -)		2→{1,3}[241]
2	4's msgs		
	2(-, *341, -)	4→{1,3}[421]	3→4[321]
	3(*21, -, 421)	4→2 W	3→2 W
	4(-, *21, -)		
3	2's msgs		
	2(-, *341, -)	2→4 W	4→{1,2,3}W
	3(*241, -, 421)	2→{1,3}[241]	
	4(-, -, -)		
4	3's msgs		
	2(-, -, -)	3→4[321]	2→{1,3,4}W
	3(*241, -, 421)	3→2 W	
	4(-, -, 321)		
	...	...	...

this route to 2. Instead, both node n and node 2 send withdrawals to each other (in this scenario, withdrawal rate-limiting is *not* in effect), but announce their choices to their other neighbors. When n receives node 2's withdrawal, however, n withdraws the route $[n\ 2\ 1]$ from all of its neighbors (stage 3 in Table 4.4). Similarly, node 2 withdraws from its neighbor 3 (stage 4). As a result, node 3 will withdraw the route from n after stage 4, so node X never sees enough flaps to exceed the suppression threshold.

WRATE: Table 4.3 shows that, unlike for the clique, the *WRATE* scenario can actually exhibit a lower convergence time. This is because when withdrawals are delayed by the MRAI timer, there are some cases where node n sees fewer secondary flaps compared to the base case. These cases depend on a particular sequence of route propagation. Please refer to [74] for an example of one such sequence. Intuitively, since the number of alternate routes going through n is much greater than ones that do not, withdrawal rate-limiting increases the probability of exploring the former routes.

Table 4.5: Convergence times of the sample real topology (Figure 4.3(c)) averaging 50 simulation runs

Parameter setting	Convergence time (second)	Update count	Damp count
Base case	243.45	132	11
MRAI=5	558.18	137	26
Less aggressive damping	1.73	132	0
SSLD	2.03	94	0
WRATE	410.34	135	18
Damping disabled	1.73	132	0

A Sample Topology

We take a sample real topology from the study done by Labovitz *et al.* [68] to test whether withdrawal triggered suppression can happen in real topologies. Table 4.5 shows the results, each data point denoting the average of 50 simulation runs. The damp count column indicates the number of simulation runs in which withdrawal triggered suppression occurred. Note that the impact of the various BGP knobs is consistent with our observations for the clique topology: setting the MRAI timer to a smaller value increases the number of messages and convergence times, and withdrawal rate-limiting worsens the convergence times and increases the number messages. What is interesting is that for this topology, SSLD and less aggressive damping both eliminate withdrawal triggered suppression. We found that with SSLD enabled, the number of MRAI rounds is reduced to one and thus reduces the likelihood of triggering route suppression. Note, SSLD cannot eliminate the possibility of withdrawal triggered suppression, because the route re-announcement may cause additional flaps.

4.5.4 Summary

In summary, our extensive simulations reveal several important observations about withdrawal triggered suppression: In many topologies, including at least one real topology fragment, BGP path explorations following withdrawal can trigger route flap damping after just a single withdrawal followed by a route re-announcement. In such cases, the route is sometimes suppressed for up to an hour. Even in topologies with a large number of alternate paths of different lengths, such as the pyramid, it is not always true that withdrawal triggered suppression is more likely to be invoked than in smaller topologies. No proposed or deployed BGP implementation features eliminate this phenomenon for all topologies. For certain topologies, *e.g.,* pyramid, sender-side loop detection can eliminate withdrawal triggered suppression.

4.6 Trace Analysis

We have already shown that withdrawal triggered suppression can happen in practice, by taking a realistic topology fragment from [68] and from our experiments of Cisco and Juniper routers in a 4-node clique topology [74]. How *prevalent* is withdrawal triggered suppression? This is a difficult question to answer with certainty. Instead, we scope this question by performing a simple analysis of BGP update traces to determine how often we can observe an important signature of delayed convergence—successive announcements of strictly increasing path lengths. Each such sequence of length greater than four can *potentially* trigger suppression at a damping-enabled router. For our traces analysis, we use publicly available routing update data from RIPE NCC [79] and the University of Oregon Route Views project [15]. This section corresponds to the measurement part of our overall methodology for obtaining more visibility BGP dynamics as illustrated in Chapter 3. Here, through passive measurement data, we analyze the likelihood of this phenomenon on today's Internet.

Table 4.6: Withdrawal triggered flap statistics

	RIPE00 01/10/2002	Oregon RV 11/15/2001
Total instances	8533	6828
Max num announcements per instance	8	7
Total unique peers	13	20
Total unique prefixes	2768	3040
Max prefix length	30	26
Min prefix length	8	8

Our trace analysis simply counts instances of routing message sequences with strictly increasing path lengths followed by a withdrawal, ignoring path length increases caused by AS path prepending. We only recorded sequences of length four or greater, since at least four flaps are required to trigger flap damping. Table 4.6 shows the results of our analysis on a particular day from both data sources. We find several thousand instances of such routing message sequences in our traces. Notice also that these sequences are not restricted to a particular peer, nor from a particular prefix, and they span a wide variety of prefix lengths. This indicates that the phenomenon we describe in this chapter may actually occur relatively frequently, and is therefore of considerable practical importance. As we conjectured earlier, we rarely observed update sequences indicative of announcement-triggered suppression, *i.e.,* routes of decreasing path lengths.

4.7 Selective Route Flap Damping

In this section, we consider a simple solution for both withdrawal and announcement triggered suppression. We should emphasize that our goal here is to demonstrate the existence of a relatively simple mechanism that will reduce or eliminate the occurrence of triggering route suppression during convergence. Much more evaluation and experimentation is necessary to understand the efficacy of the scheme under various topologies, as well as its incremental deployability. That is the subject of future work.

The key to our mechanism is to detect route changes due to path exploration to avoid increasing penalties. From the clique example in Section 4.3, one might conclude that one way to detect route changes due to path exploration is to avoid penalizing successive routes with non-decreasing path lengths. Thus, if a new route has the same or longer path length than the existing route, we do not increment the flap penalty.

While this works for the simple example we discussed above, it does not work well in general. In particular, policies at various nodes in the clique can, in theory, cause longer path lengths to be explored first than shorter ones (if they happen to be more preferred). So, a more general observation might be that each node, during convergence after withdrawal, selects routes in order of non-increasing preference until it finally withdraws the route. Thus, if the sender of a route includes its current preference for the route (a feature that BGP currently lacks for external peers), the receiver of the route can compare the sender's preference for the received route with that of the previous route from the sender. The preference value can be encoded in a specialized community attribute that is nontransitive, making our proposal incrementally deployable. The receiver can then increment the penalty for the route if the new route does not have a higher preference (at the sender) compared to the previous route.

This simple mechanism does not work perfectly. The sequence of route changes seen from a peer during withdrawal convergence can have route withdrawals interspersed with routing updates.[17] Furthermore, in some topologies such as the pyramid, this can happen even without SSLD (see [74] for an example). Thus, our mechanism has to deal with this situation as well.

Our proposed mechanism is a modification to route flap damping that we call *selective route flap damping*. It requires the sender to attach to each route announcement its local preference or the relative preference value compared to the previous route announcement. We keep two bits for each destination route from each peer. These two bits encode the comparative value of the last two announcements received. We call these two bits the *comparison bits*. 00 denotes the situation where fewer than two routes have been received. 01 denotes that the values of the two routes are the same. 10 means the latest route has higher degrees of preference than its previous route. And finally, 11 indicates the latest route is less preferred. When an announcement is received, comparison bits are recomputed based on the current announcement and the latest announcement. The newly computed comparison bits are compared with the stored comparison bits. If these two sets of comparison bits indicate that the direction of route preference change has altered, then we count the current

[17] This can be caused by sender-side loop detection, policies, or update reordering.

announcement as a flap. In other words, if one set of comparison bits is 10 and the other is 11, we consider the announcement received as a flap. This heuristic is used, because secondary flaps are always of either increasing or decreasing degrees of preference.

To deal with interleaved withdrawals, selective damping temporarily ignores withdrawal messages until the next announcement is received. We keep track of the *temporary* penalty corresponding to the withdrawal message and let it decay exponentially just like the regular penalty value. This temporary penalty would have been added to the penalty in the existing scheme. If the next announcement received is considered a flap, this temporary penalty is added to the penalty value in addition to the penalty corresponding to the current flap. Otherwise, the temporary penalty is discarded. Here we add another condition under which the current route is considered a flap. If the route received has the same preference value as the previous one, we do not simply discard it as a redundant update, because the announcements could be interleaved by withdrawals. Thus, we count the current announcement as a flap if it has the same value as the previous announcement *and* is preceded by a withdrawal. The goal of this slight modification is to make sure the new scheme can contain real flaps.

Selective damping is thus designed to ignore route changes caused by withdrawal exploration, yet to mimic unmodified route flap damping. It does so, but with one caveat. Because of the way it deals with withdrawals, it penalizes true route flaps to the same extent that unmodified route flap damping would, but it might do so slightly later (because it has to wait for the announcement following the withdrawal to penalize the route). At most *one* extra withdrawal message is propagated under the new scheme.

Finally, selective flap damping also eliminates announcement triggered suppression, which consists of successive announcements of increasing degrees of preference. Since our scheme does not count successive monotonic route changes as flaps, both forms of suppression are eliminated.

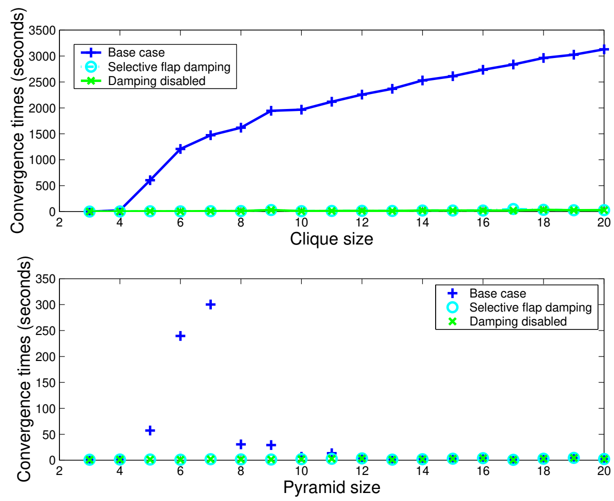


Figure 4.7: Convergence times of the clique and pyramid topology (averaging 50 simulation runs)

Following our general research methodology of combining analysis, measurement, and traffic correlation as described in Chapter 3, we have validated through simulation that selective flap damping actually eliminates withdrawal triggered suppression. In Chapter 5, we will use actual measurement trace data from the Internet to perform further trace-driven validation of the effectiveness our solution. As shown in Figure 4.7, selective flap damping exhibits convergence times comparable to the situation when damping is disabled both for the clique and the pyramid topologies. In addition, we also verified this for our realistic topology, where selective flap damping exhibits the same convergence time and number of messages as the case when flap damping is disabled.

Furthermore, we verified that selective damping can suppress actual flaps. To do this, we simulated network failures by making node 1 in each of our topologies repeatedly flap (*i.e.,* alternately withdraw and announce the route to d) with a period of 40 seconds.[18] We then observed the number of additional messages it takes for selective damping to suppress the route compared to the unmodified route flap damping implementation. Our simulation shows it takes at most 8

[18]The maximum frequency is limited by MRAI timer value.

additional messages for selective damping to suppress a continuously flapping route compared to the original RFD scheme. A scheme that does not use any form of damping will instead send an update every 40 seconds. For each topology size, the actual number of additional messages differs. For instance, for a clique of size 5, it takes on average 3 extra messages. For a clique of size 20, it takes on average 6 extra messages.

4.8 Summary

In this chapter, we analyze a previously not well-studied interaction between BGP's route withdrawal process and its route flap damping mechanism for ensuring the overall stability of the Internet routing system. This interaction can, depending upon the topology, suppress up to one hour the propagation of a route that has been withdrawn once and re-announced. We have shown that this interaction has a number of subtle features. For instance, we found that in the pyramid topology increasing the size of the topology actually improved the rate of convergence.

We have proposed a simple fix to this withdrawal triggered suppression called selective flap damping. It relies on being able to weed out secondary flaps using a monotonicity condition which selectively avoids penalizing such secondary flaps. Our selective flap damping mechanism successfully eliminates withdrawal triggered suppression in all the topologies that we have analyzed. Our work together with [63, 68] makes it clear that faster convergence does require modifying BGP. This could be done by either fixing the withdrawal path exploration phenomenon (the direction followed in [87]) or by deploying a mechanism similar in spirit to selective flap damping (as in our work). Either way, such BGP modifications could move us closer to the Holy Grail: an inter-domain routing protocol that is stable and yet reroutes traffic extremely fast after failure.

Revisiting our research methodology described in Chapter 3 and our overall goal of designing a framework bringing more visibility into BGP dynamics, we have illustrated in this chapter an example of how complex dynamics between BGP features result in much worse performance. Our use of analysis through simulations based on representative small topologies and experiments in a router testbed exposes this previously not well-known interaction between route flap damping and path-vector convergence. We thereby also demonstrate the BGP complexity due to heterogeneous default parameter settings. Thoroughly understanding the phenomenon allows us design a remedy for this undesired interaction. We validated our improved flap damping algorithms again using simulations. There is still a need to answer the questions of how prevalent this interaction is on today's Internet and how our solution works in practice. To answer these questions, in next chapter, we present an active measurement infrastructure through which we confirm the presence of this interaction between flap damping and convergence on the current Internet. The infrastructure also allows us to evaluate our improved algorithm using trace-driven simulations.

Chapter 5

BGP Beacons: A BGP Measurement Infrastructure

In the previous chapter, we analyzed the interaction between route flap damping and BGP convergence. Although we carefully studied the interaction using both simulations and a router testbed consisting of commercial routers commonly used in the core of the Internet. The question of how often such interaction occurs on today's Internet still remains. The answer to such a question is important, as it can also reveal network configurations that can potentially reduce the occurrence of such problems. To answer this question, we must be able to associate a sequence of routing updates with the originating routing change. Or we must be able to know the actual input of routing changes in terms of the location, time, and the type of change and ideally the routing changes should cover topologically diverse locations. One way to have such information is through an active measurement infrastructure where we precisely control the injected routing change. We now describe such an active measurement infrastructure called BGP Beacons to understand how often route flap damping is triggered on today's Internet. Using controlled BGP experiments we also characterize routing convergence. Furthermore, we can model convergence response to infer root causes for BGP updates. BGP Beacons is intended to be a long-lived public active measurement infrastructure to be used to calibrate BGP behavior to aid in studying BGP dynamics.

This chapter is organized as follows. Section 5.1 introduces BGP Beacons and its goals to provide more visibility into complex BGP dynamics. We explain why passive measurements are not sufficient for studying BGP dynamics due to lack of control of originating routing changes and the difficulty in inferring root causes of arbitrary BGP updates, and how our work differs from previous BGP measurement efforts. More details about the Beacons are covered in Section 5.2 including their deployment and the measurement setup through periodic routing update announcement. As the BGP Beacons are an open public measurement infrastructure for all researchers, we describe the process of processing and interpreting BGP updates generated by the Beacons in Section 5.3. We illustrate four example uses of BGP Beacons to understand BGP dynamics in Sections 5.4, 5.5, 5.6, and 5.7. First, we show how Beacons can help us indentify BGP implementation variants between Cisco and Juniper in terms of the update rate limiting behavior through average signal length and signal duration distribution. To answer the question raised by the previous chapter, Section 5.5 shows that route flap damping can occur on today's Internet, and at some locations it is triggered in close to 90% of the cases where we control injected input routing changes. This validates the importance of our finding in Chapter 4. We explore an aspect of BGP dynamics – update inter-arrival time, not considered in any previous work as the third example of using Beacons in Section 5.6. We revisit previous studies in studying BGP convergence in Section 5.7 as the last example. We conclude this chapter in Section 5.8.

5.1 What is a BGP Beacon?

Passive monitoring of BGP updates has resulted in important insights into the dynamics of BGP [65–67]. As mentioned in Chapter 3, several public sources, such as Oregon's Route Views [15] and the RIPE Routing Service [79], provide BGP updates collected from a large number of points in the Internet. Passive measurements are not sufficient for all purposes due to the difficulties in identifying the root causes of BGP updates [49] and so active techniques have also been employed in the analysis of BGP dynamics [63, 68]. With the active approach, prefixes are announced and withdrawn from the global routing domain while quantities such as convergence time are measured. The main advantage of the active approach is that the *input* to the routing system is known (for example, an announcement or a withdrawal), which allows inferences to be made that would be difficult or impossible with purely passive measurements.

To date, the route injection infrastructure of such experiments has either been temporary in nature, or has its use restricted to the experimenters. Mounting such an infrastructure is often beyond the means of many interested in this area of research, and so we feel that the routing research community would benefit from a permanent and public infrastructure for such active routing probes. We use the term *BGP Beacon* to refer to a publicly documented prefix having global visibility and a published schedule for announcements and withdrawals. A BGP Beacon is to be used for the ongoing study of BGP dynamics, and so should be supported with a long term commitment. We describe two collection of BGP Beacons that have been set up at various points in the Internet. We then describe techniques for processing BGP updates when a BGP Beacon is observed from a BGP monitoring point such as Route Views or RIPE.

We illustrate the use of BGP Beacons with four case studies. Each study relies on the fact that we are monitoring updates that have been (indirectly) generated by a Beacon event. First in

B	Prefix	Period	Upstream ASN(s)	Src ASN	Beacon ASN	Host ASN	Location	Host	Start date
1	198.133.206.0/24	2 hrs.	2914, 1239(1)	3972	3972	3130	WA,US	R. Bush	8/10/2002
2	192.135.183.0/24	2 hrs.	3701,2914	5637	5637	10876	OR,US	D. Meyer	9/4/2002
3	203.10.63.0/24	2 hrs.	1221	1221	private	1221	Australia	G. Huston	9/25/2002
4	198.32.7.0/24	various	2914,8001	3944	private	3944	MD,US	A. Partan	10/24/2002
5	192.83.230.0/24	2 hours	2914, 1239	3130	3130	3130	WA,US	R. Bush	6/12/2003

Table 5.1: The PSG Beacons

Section 5.4, we study the implementation differences between Juniper and Cisco router in terms of update rate limiting. Then in Section 5.5, we investigate the potential that route flap damping [107] has for punishing “well behaved” routes. Simulation results in [75] shown in the previous chapter, Chapter 4 have shown that route flap damping can punish “well behaved” as well as “misbehaving” routes. Here we use the BGP Beacons to validate those results in the global Internet. Even though the BGP Beacons we use have a fairly long cycle (two hours between each announce or withdraw event), we see that *even announcements* can trigger flap damping in as much as 10 percent of the time at some locations in the Internet. For withdrawals, up to 90 percent of the Beacon events trigger route suppression. For our third study in Section sec:BB:Itime, we present an analysis of the inter-arrival times of updates generated by BGP Beacons. In Section 5.7, we revisit the convergence time issues studied in [63, 68].

5.2 BGP Beacons

Currently, there are two groups of BGP Beacons that differ somewhat in implementation. There are four Beacons in the first group, called the *PSG Beacons* because the first was set up at `psg.com`, which are listed in Table 5.1. I set up these Beacons were with the help of the Beacon hosts. A web site for these Beacons at `http://www.psg.com/~zmao/BGPBeacon.html`

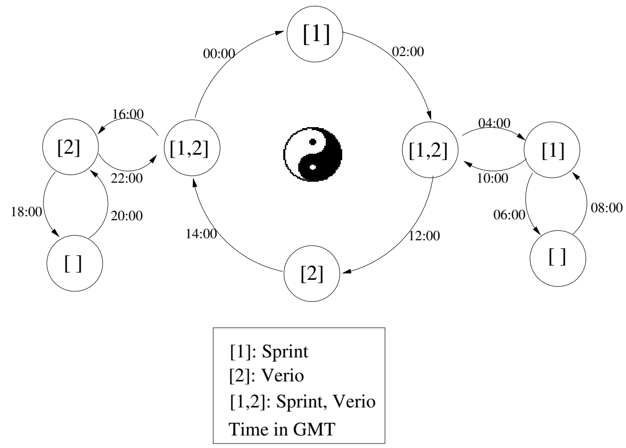


Figure 5.1: Schedule for Beacon #5

contains contains Beacon related scripts and Beacon data derived from Route Views. The Beacon *period* is the time between each event at the Beacon, where an event is either an announcement of the Beacon prefix or a withdrawal. We picked two hours as an period for the first three Beacons since by experience this is long enough for most route flap damping to expire. Beacon #5 has a special schedule as shown in Figure 5.1 to study the effect of multi-homing fail-over. It is multi-homed to Verio and Sprint and we selectively announce and withdraw to each provider to understand how fast BGP can find the route through the other provider.

In PSG Beacons, two attributes of the announcements have been "hijacked" to serve as a timestamp and a sequence number. The aggregator IP attribute, which is an IP address, is set to have the form `10.X.Y.X` where `0.X.Y.X` (in binary) represents the number of seconds since the start of the month (GMT). The aggregator ASN attribute, is a number that is incremented with each announcement and cycles through the values from 64,512 to 65,635. Note that values of both the Beacon timestamp and Beacon sequence numbers are within "private" spaces.

The second group of Beacons, called the RIPE Beacons [10], have been set up as a part of the RIPE Routing Information Services (RIS). Each BGP route monitor, located in nine different locations, is associated with a unique BGP Beacon prefix from `195.80.244.0/24` through `195.80.232.0/24`. Each RIPE Beacon has a period of 2 hours.

The implementation of these Beacons differs in the following ways. First, PSG Beacons currently have timestamps and sequence numbers, while the RIPE Beacons do not. Second, the PSG Beacons currently have what we call *anchor prefixes* (see Section 5.3) associated with them, which aid in the pre-processing of update data. Third, the PSG Beacons are not associated with BGP routing monitors, as are the RIPE Beacons.

For the rest of this study, we focus on PSG Beacons 1, 2, and 3, as Beacon 4's varying period may result in interaction between consecutive signals. We are not currently using the RIPE Beacons because of the lack of anchor prefixes needed for data cleaning.

5.2.1 Beacon Software

As mentioned in Chapter 3, the PSG Beacon daemon software is based on the open-source BGP software router written in perl available at `http://bgpd.sourceforge.net/`. The original software is purely passive and does not provide any functionality to advertise routes. We modified it to inject routing changes to an open BGP session triggered by user-defined interrupt signal. A cron job is set up to regularly send an interrupt to the Beacon daemon software, so that announce and withdraw updates are sent alternately. In the beginning, the updates are sent every 30 minutes. We quickly discovered that the prefix was not visible at all and this was due to the route flap damping mechanism [107] implemented by the upstream provider of the Beacon. Subsequently, the schedule is set to be 2 hours between consecutive updates to minimize the likelihood of route suppression.

5.2.2 Terminology for BGP Update Propagation

We use the term *input signal* to refer to any update generated at a BGP Beacon (either an announcement or a withdrawal). The network of BGP speaking routers can be thought of as a giant non-deterministic signal transducer [49], where each input signal *causes* various *output signals* to be generated at different locations in the Internet. Output signals generated by the Beacon input signals can vary considerably, depending on the Beacon, the monitor point, and the time observed. For example, here is a an output signal for an announcement from PSG Beacon 1 sent, as seen from one peer at Route-Views on January 11, 2003:

Time (GMT)	Type	AS Path
05:00:11	A	8121 19151 2914 1 3130 3927
05:00:39	A	8121 16631 174 1 3130 3927
05:01:08	A	8121 3491 1 3130 3927

This output signal contains three updates. The *signal duration* is the elapsed time from the first to the last update in the signal. For this example, the signal duration is 57 seconds. A signal containing only one update has a duration of 0 seconds.

5.2.3 Convergence Time

A BGP monitor may be receiving BGP updates from more than one neighbor. For example, Route Views currently has over 20 neighbors or peers. For any Beacon event, there will be some neighbor that sends an associated update first. The time between this first update and the last update in a signal collected at that point is called the *relative convergence time*. For example, if the first update received from any neighbor for the announcement event above was 05:00:06, then the example signal has a relative convergence time of 62 seconds. *End-to-end convergence time* is the time between the last update in a signal and the sending time of the input signal based on the Beacon timestamp in the aggregator field.

Ideally, we would like to know the end-to-end convergence times, but this requires clock synchronization. Both the Beacon machine and the monitoring sites should be NTP synchronized according to people who administer these machines. However, to our surprise, we discovered many instances where the receiving timestamp of a Beacon update is smaller than the sending timestamp of the update due to the problem with clock synchronization. As we do not have control over the machines at monitoring sites nor the machines that run the Beacon software, we sometimes use relative convergence times and signal durations to understand the convergence delays.

5.2.4 Beacon Location Terminology

As seen in Table 5.1, there are several AS numbers associated with a Beacon: Source AS, Beacon AS, and Host AS. *Source AS* is the origin AS in the AS path of the updates associated with the Beacon prefix. *Beacon* AS is the AS to which the Beacon daemon belongs. If EBGP session is used, where the BGP session is between two routers in different ASes, it can be either a private AS number or a special AS number for the purpose of Beacon experiments. In the case of IBGP session, the Beacon AS number is the same as the *Host AS* which is the AS that directly provides the network connectivity for the Beacon prefix. In contrast, the upstream AS is the closest tier-1 ISP that provides connectivity to the Host AS. The Host AS itself can be the upstream AS in the case of Beacon 3 hosted in Australia. The Source AS number is that of the Beacon AS unless it is private. In that case, the Source AS number is that of the Host AS. In general, there is no need for a special AS to create a Beacon.

5.2.5 Public Monitoring Points

There are several public monitoring points where BGP data are collected from several ISPs. Route Views [15] is one such monitoring point which peers with about 30 different networks and receive all the BGP updates. Since the Beacon prefixes are not aggregated and should be globally visible, they are visible in all the BGP feed available at the monitoring points.

5.3 Data Cleaning and Signal Identification

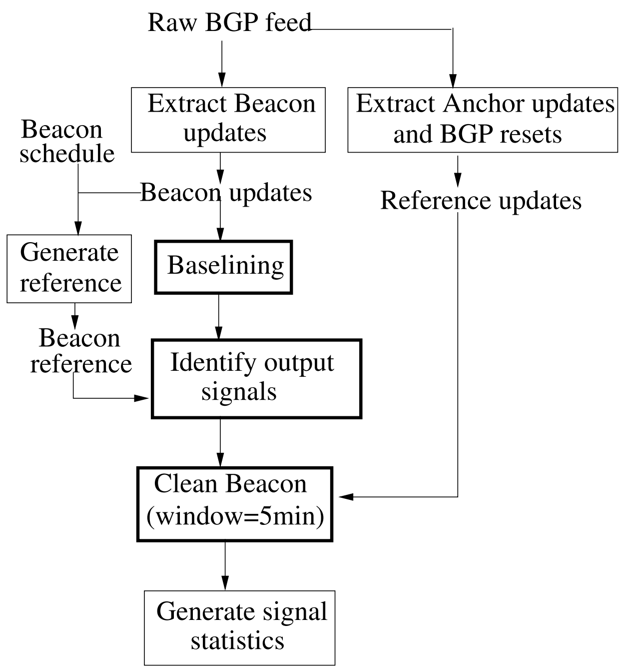


Figure 5.2: The process of cleaning Beacon data and identifying signals

For the Beacon analysis, it is important to clearly identify the output signals by associating the observed BGP updates with a single input signal. We describe in detail a novel methodology to achieve that goal. Not all updates observed related to the Beacon prefix are caused by our input signals. Some of them, for instance, may be caused by routing changes in an upstream AS from the observation point. Given a BGP feed at Route Views, for example, there are typically always some routing updates observed every second. Take a typical day, May 2 2003, there are about 70 updates observed from about 30 peers at Route Views. Such routing changes can be due to a variety of reasons such as routing policy change, link failure, congestion. BGP session requires keep-alive messages to be exchanged between the two neighbors. If timeout occurs due to congestion or link failure, the BGP session is reset. Upon session reestablishment, the entire routing table is exchanged resulting in a large number of updates observed. Such session resets can occur locally between the local router and the route monitoring software. In that case, such updates do not reflect actual routing changes that affect the forwarding plane of the data traffic. Work by Wang *et al.* [108] demonstrates the importance of filtering out updates due to local BGP session resets in analysis. Very different conclusions are drawn if such updates are not properly filtered.

We take a sequence of steps as shown in Figure 5.2 to clean the Beacon data to identify output signals and compute several signal statistics. This process consists broadly of three crucial steps: baselining, signal grouping, and noise filtering or cleaning.

5.3.1 Baselining

The goal of this step is to process the BGP data such that we can compare them from all peers in a sensible way. We first extract from the raw data the updates associated with the Beacon prefixes, *i.e.,* the Beacon updates. We found some peers at Route Views send out updates related to the BGP community and MED attribute changes. BGP policies can be set to prevent such attribute changes from being sent. These peers tend to send more updates and reveal more of the internals of the AS, as such attribute changes typically reflect the BGP dynamics within the last-hop AS. We also found that some peers at of Route Views send out consecutive updates that are identical. To do a reasonable comparison, during the baselining step, any update which is either identical to the previous update or differs only in its community or MED attribute values are eliminated. This reduces about 15% of the updates for all Beacons based on Route Views data. 12 out of 23 peers at Route Views send such duplicate updates that differ only by these two attributes from previous updates. We verified that eliminating such updates have little or no effect on our analysis of inter-arrival update time (Section 5.6) and convergence delay (Section 5.7). However, the results of route flap damping (Section 5.5) will provide a lower bound, and the analysis of signal length can also provide an underestimate.

5.3.2 Signal Identification

For the ease of analysis, we do additional processing to group updates together according to the input signals. To achieve that, we create *Beacon references* which identify the starting time of each output signal for the input signals injected. Any output signals received have timestamps greater than or equal to the Beacon reference timestamps. Such timestamps can be easily generated if we have synchronized clocks, as the Beacon schedule is recorded by the Beacon daemon software. Similarly, we can generate such timestamps by using the Beacon timestamps in the BGP update aggregator field. However, as mentioned before, the clocks of the Beacon machine and the monitoring machines are not well synchronized.

We thus resort to several algorithms to generate the Beacon references. If the monitoring sites receive the BGP feed directly from the AS that hosts the Beacon, the Host AS, then the timestamps of the messages from that AS can be used in the references. The BGP monitoring site, such as Route Views, usually establishes multihop BGP sessions with several different ISPs. The output signal coming from the Host AS almost always arrives first, as it typically travels through the fewest number of routers and smallest distance. Two of the Beacons (Beacon 1 and 3) fall into this category. Typically, the Host AS produces very clean output signals which consist of a single announcement given the input announcement signal and similarly a single withdrawal given the input withdrawal signal. It is important to point out that if there are no issues with time synchronization, the Beacon timestamps provide very accurate timestamps for the references that can be used for all monitoring sites. The time stamps based on the Host AS is specific to the monitoring site and requires that there is a BGP feed from the Host AS at the site. If other sites use such Beacon references generated from a different site, there may be offsets from the reference timestamps for the output signals, *i.e.,* some output signals may start earlier than specified due to clock differences.

If the BGP feed from the Host AS is not available at the monitoring sites, heuristics are used based on the Beacon schedule to determine the start of a new Beacon signal. The period of Beacon announcement is purposely set to be two hours for the first three Beacons. We therefore expect that most sites should already have converged on the final route long before the next input signal is injected. One simple heuristic is to look for large gaps between updates and use the Beacon schedule as a reference to identify the starting times of the output signals. It is easy to distinguish the start of a new announcement signal with the help of the sequence number in the aggregator field. The start of a withdrawal output signal is identified using the timing heuristic.

5.3.3 Noise Filtering/Cleaning

To differentiate the updates caused by our injected routing changes from updates caused by other effects, we propose the use of *anchor prefix* to detect such unexpected routing changes. An anchor prefix is a statically nailed down prefix belonging to the Beacon AS or the Host AS. Such a prefix can contain live hosts and thus does not experience routing changes caused by our experiments. It can also be an unused prefix purposed announced for the sake of reference with the Beacon prefix. Anchor prefixes serve as calibration points to identify unexpected routing changes. When there are no such routing changes, no routing updates associated with the anchor prefix can be observed. Anchor prefixes originate from the same AS as the Beacon, so that any routing change experienced by the anchor prefix must also be experienced by the Beacon prefix and is not caused by our purposely injected routing changes. In general, we found Beacons hosted by larger ISPs, e.g., AS1221 in the case of Beacon 3 to be more stable as it has more redundant network connectivity. This is evidenced by the observation that Beacon 3's anchor prefix files are typically one third smaller than other Beacons.

The Beacon signals are cleaned by deleting signals that can be affected by unexpected routing changes as experienced by anchor prefixes. For each anchor prefix update, we construct a window starting at W minutes before the update time and ends at W minutes after the update time. Any Beacon prefix signal that have an overlap with such a window is ignored, as they are likely to affected by external routing changes. Based on empirical observation, it takes on the order of minutes for a routing change to become globally visible. We set W to be 5 minutes in our analysis. We perform the same cleaning process using certain BGP STATE messages which indicate BGP session resets. Using Route Views data, we observe that the number of output signals remain fairly constant when $W >= 5$. The cleaning process deletes on average 2 to 3 percent of updates.

Tables 5.2 and 5.3 show the effect of cleaning on observed signals in terms of signal count, average signal duration, delay, and signal length. They demonstrate the importance of cleaning. For all four Beacons, less than 5% of the signals have been deleted after cleaning. Overall, the average signal delay and signal duration have decreased for both announcement and withdrawal signals after cleaning. In some cases, the decrease is as much as 50% of the original value. However, the signal length remain mostly the same after cleaning. This means that cleaning removes those outliers with large inter-arrival time or long signal duration.

For each of the four PSG Beacons, we perform this sequence of steps and finally generate a set of signal statistics such as relative convergence, signal duration, end-to-end convergence time, number of updates.

Table 5.2: Effect of cleaning on observed announcement signals: signal count, average duration, delay, and length

B	Before cleaning				After cleaning			
	count	avgDur (sec)	avgDelay (sec)	avg sigLen	count	avgDur (sec)	avgDelay (sec)	avg sigLen
1	33536	27.13	50.60	1.47	33318 (99.35%)	19.36	41.89	1.47
2	34522	9.13	29.56	1.20	33726 (97.69%)	6.75	25.21	1.17
3	32504	10.82	34.99	1.22	32188 (99.03%)	5.77	28.40	1.21
4	39044	41.95	63.66	1.52	37970 (97.25%)	22.79	43.16	1.46

Table 5.3: Effect of cleaning on observed withdrawal signals: signal count, average duration, delay, and length

B	Before cleaning				After cleaning			
	count	avgDur (sec)	avgDelay (sec)	avg sigLen	count	avgDur (sec)	avgDelay (sec)	avg sigLen
1	33443	37.88	100	2.07	33261 (99.46%)	32.98	90.09	2.07
2	33860	45.24	109.23	2.19	33344 (98.48%)	42.94	94.38	2.19
3	32379	59.16	120.64	2.55	31182 (96.30%)	56.36	114.40	2.55
4	36633	96.33	139.63	3.43	35776 (97.66%)	75.65	115.90	3.41

5.4 BGP Implementation Impact: Cisco vs Juniper

One question of some interest is how much impact the different implementations of BGP have on the results. The BGP specification (RFC 1771) [92] defines the protocols to be used, but not how they should be implemented, and in some cases BGP implementations have contained bugs which meant they did not even match this specification. There are implementation differences between different router vendors, and even between models, and software versions from the same company. In addition, there are settable parameters which may impact behavior (*e.g.,* the MinRouteAdverTimer). As specified by the BGP RFC, MinRouteAdverTimer specifies the minimum amount of time a router needs to wait before second consecutive updates for the same router to the same neighbor. In an ideal world, these would have little impact on the operation of BGP, but studies have shown (*e.g.,* [50]) that at least some of these differences may have large impacts.

In this section we consider one example of the impact that differences in implementation may have on the behavior of BGP. Namely, we consider the difference between Cisco and Juniper implementations of BGP. The decision to compare these two (out of all the possibilities) is motivated by the fact that we know the make and model of two last hop routers (as seen by Route Views). In general this information is not publicly available, but the last hop routers corresponding to Peer 147.28.255.1 and 147.28.255.2 are known to be Cisco and Juniper routers, respectively.

Table 5.4 presents a comparison between Juniper and Cisco routers (as seen from Beacon 2). The table shows that the Juniper router sends about 25% more updates, has a similar update duration, and a substantially smaller average inter-arrival time for updates (around 60% of that for Cisco routers). The most startling difference, though, is in the number of short ($<$ 26 second) inter-arrival times is much greater for the Juniper router.

		signal length		duration		inter-arrival		% of short inter-arrivals	
Peer	Type	A	W	A	W	A	W	A	W
147.28.255.1	Cisco	1.20	2.07	6.79	48.4	34.8	45.4	1.56	0.44
147.28.255.2	Juniper	1.50	2.49	7.13	44.3	14.2	29.6	12.76	4.37

Table 5.4: A comparison of the Cisco and Juniper Routers. The table shows average statistics (including the average signal length, or number of updates, the average duration, or convergence time, the average time between updates during a sequence of events, and the percentage of inter-arrival times less than 26 seconds), for announcement 'A', and withdrawal 'W' events, for the two known last hop routers.

The differences can be explained by the fact that, by default, the MinRouteAdverTimer is turned off in Juniper routers [76]. It is common practice for users to leave default settings alone, unless they have particular reason to do otherwise, and we know from discussion with the administrator of these routers that the defaults are used here. As shown in [50] having a small, or zero MinRouteAdverTimer can result in large numbers of additional updates (as seen here). Also shown in [63] was how the MinRouteAdverTimer spaces out the updates, thereby potentially increasing the convergence time – also as seen here.

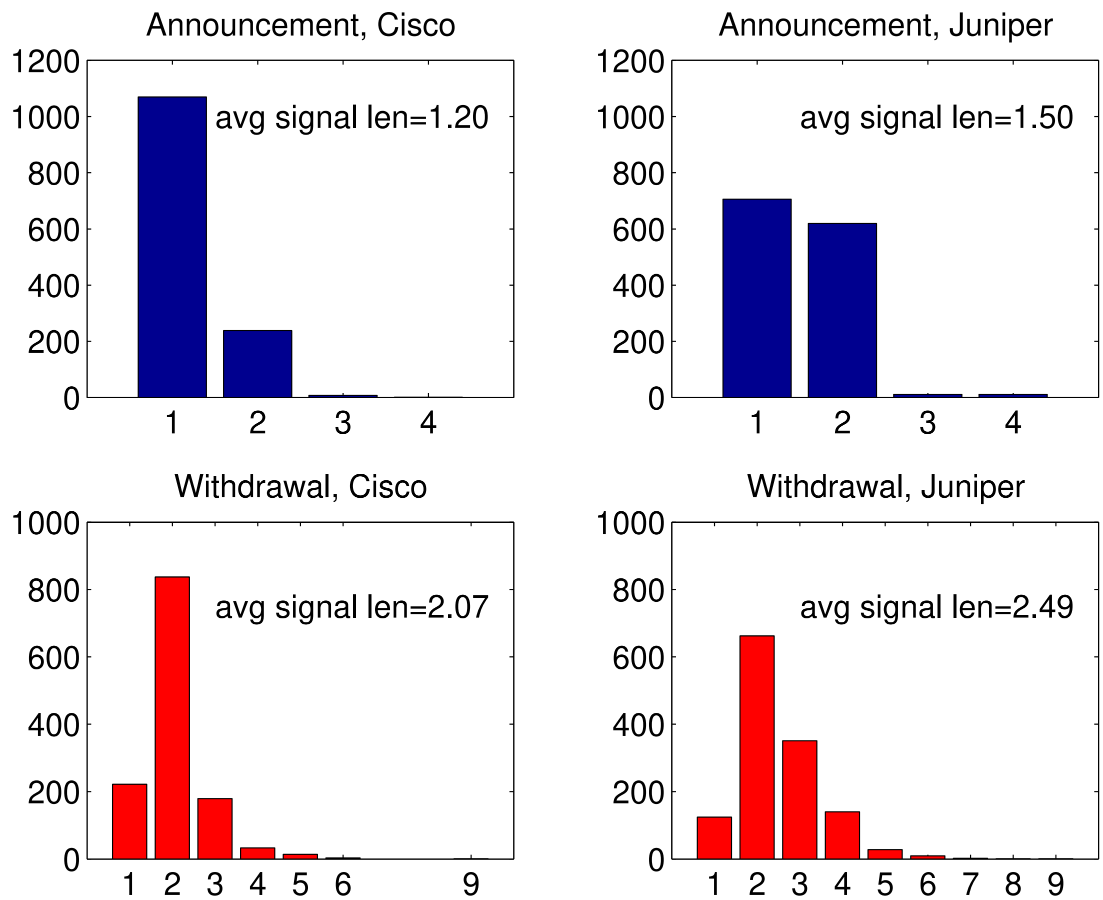


Figure 5.3: Beacon 2: Comparison of the numbers of updates for two known Cisco and Juniper routers from the same peer.

Given the obvious difference between the two router types above, the question naturally arises. Can we distinguish the other last hop routers as being of one type or the other? Figure 5.4 suggests that we can separate the two, though not perfectly. We could identify candidates from this figure, but needed to do a final verification by examining the update sequences in detail to confirm the findings. The routers that appear to be Juniper routers are marked on the plot.

The box plots in Figures 5.5 and 5.6 clearly demonstrate the difference in signal duration distribution as the signal length increases. For Juniper-like routers, there is no clear dependence

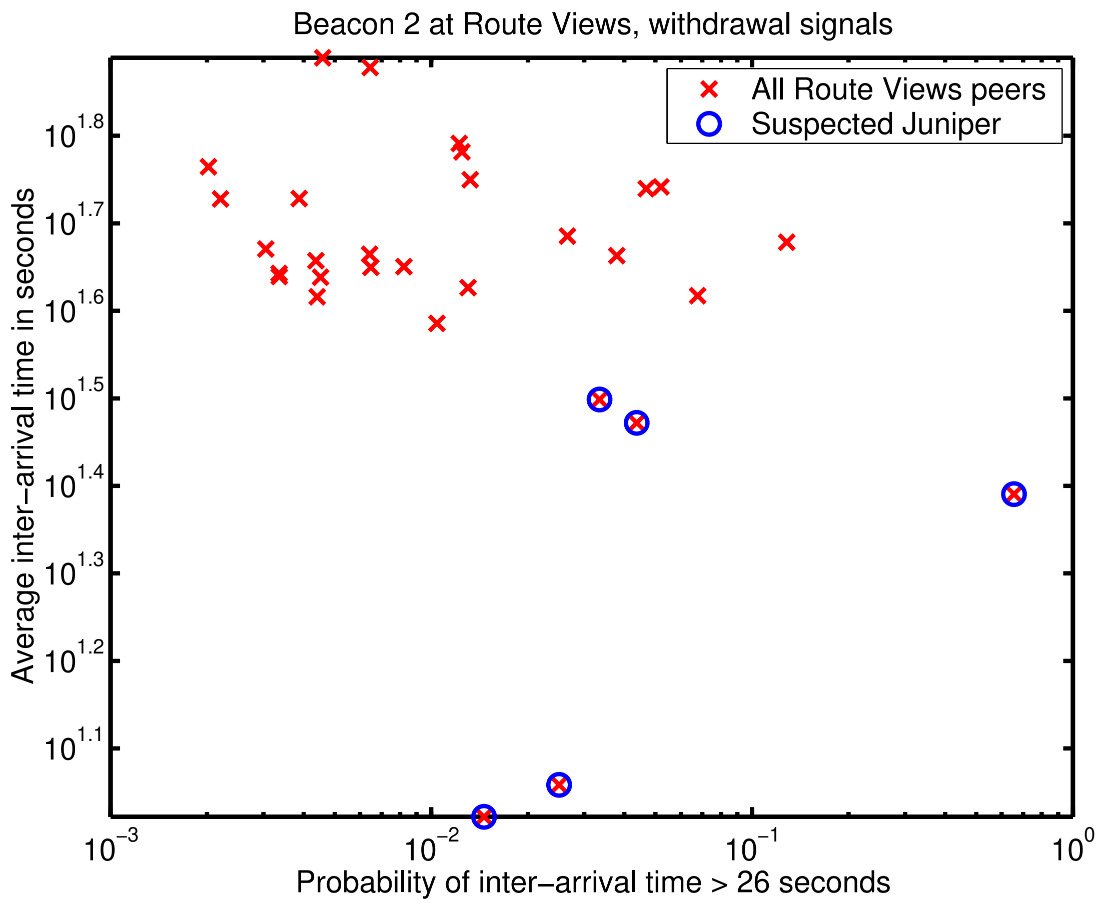


Figure 5.4: A scatter plot showing the routers classified as Juniper-like. Although there is not a completely clear distinction in the plot, detailed examination of the update sequences shows that the marked peer routers show similar characteristics to the known Juniper router.

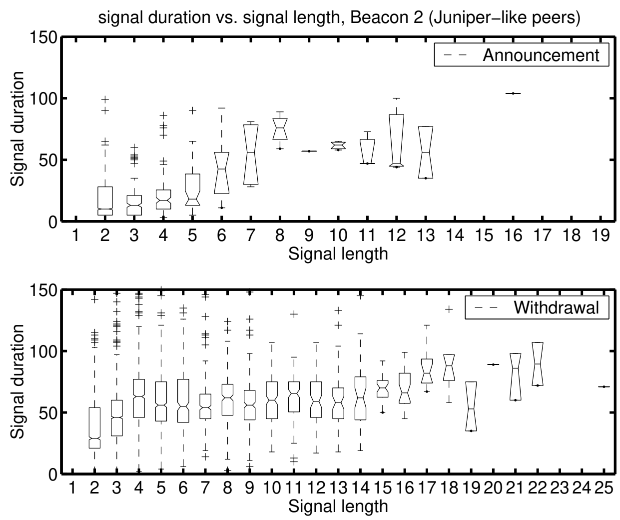


Figure 5.5: Beacon 2's signal duration distribution for each signal length for Juniper-like peers

between the signal duration and signal length. As the signal gets longer, the duration increases only slightly and keeps the values around 40 to 60 seconds. This is because Juniper's default rate-limiting algorithm allows updates to be sent in burst imposing little delay. In clear contrast, Cisco-like routers clearly show the 30 second rate-limiting behavior: there is a linear increase with slope of around 30 seconds in the duration as the signal gets longer. Furthermore, signals generated from Cisco-like peers have shorter signals compared to that of Juniper-like peers.

Beacon 1's signal duration distribution is shown in Figures 5.7 and 5.8. Again we differentiate between Cisco-like and Juniper-like peers. Announcement signals have in general shorter durations compared to withdrawal signals. In addition, we again see the signal duration for Cisco-like peers to be multiples of 30 seconds. The distribution for Juniper-like peers is much more spread out. Large number of signals converge within 30 seconds for both types of routers.

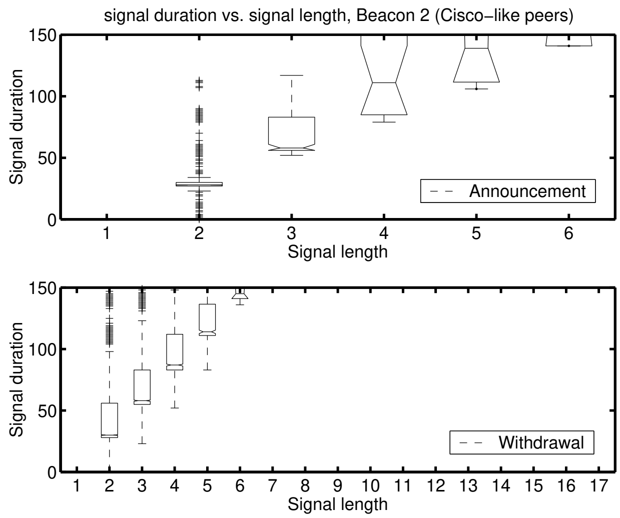
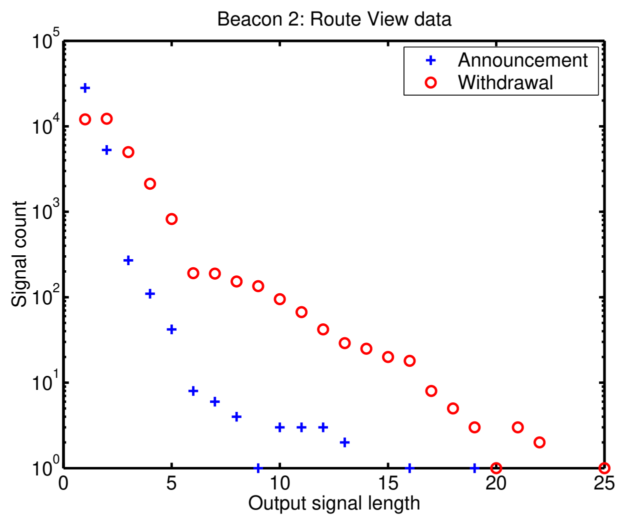


Figure 5.6: Beacon 2's signal duration distribution for each signal length for Cisco-like peers

5.5 Route Flap Damping Analysis

Route flap damping [107] is one of the two mechanisms in BGP aimed to achieve routing stability. Flap damping punishes unstable routes or routes that change frequently by suppressing them. It is designed to deal with routes that are unstable on a long time scale. In contrast, the other mechanism, the minimum route advertisement timer (minRouteAdverTimer) is designed to act on routes that are unstable on a short time scale. MinRouteAdverTimer specifies the minimum amount of time a router needs to wait before sending consecutive updates referring to the same prefix to the same neighbor. The hope of delaying the updates is for consecutive updates to be batched together to reduce update traffic. These two mechanisms can interact, as the timer determines the amount of updates that are propagated during the convergence process.

We have shown in Chapter 4 in simulations and commercial router testbed setting that in certain topologies, no matter how large the minRouteAdverTimer is, there are sufficient updates induced by a single route change to trigger route flap damping. This means that a single router



reboot, which translates to a withdrawal message followed by an announcement message, can cause the route to be suppressed somewhere on the Internet. As there is no feedback in the flap damping mechanism, it is difficult to determine which router suppresses the route. If the route suppressed is the only route to reach a destination prefix, then the destination becomes unreachable. It is thus very important to understand how likely this occurs in today's Internet.

It is very difficult to understand the extent at which route flap damping can suppress well-behaved or stable routes on today's Internet. The difficulty arises due to the complexity in inferring the root causes of BGP updates observed in passive measurements [49]. The Beacon infrastructure provides a perfect medium to do such study, as routing changes are injected at known times and locations. Assuming the Beacon prefix routes are not suppressed, we can simulate how likely a single routing change can cause the route to be suppressed. We implemented the route flap damping algorithm using the Cisco and Juniper default parameters and calculated the percentage of signals that can trigger route suppression at the monitoring sites. Based on Route Views data, we observe about 5% of input signals are suppressed across all Route Views peers as shown in Figure 5.9. Some peers are much more likely to suppress the route than others because the large number of updates

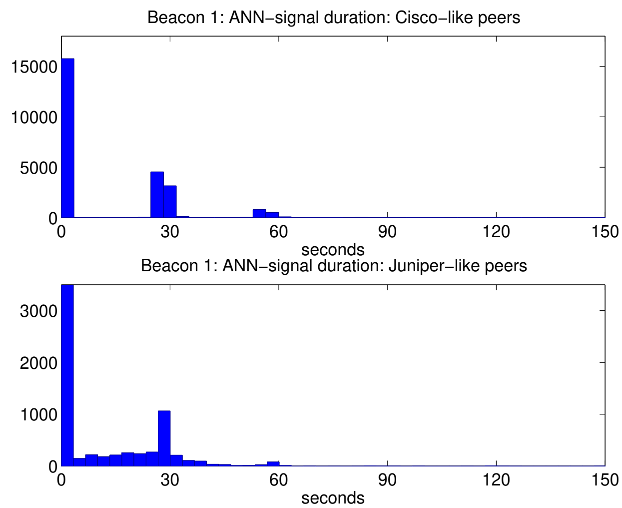


Figure 5.7: Comparing Beacon 1's announcement signal duration distribution for Cisco- like peers with that for Juniper-like peers

generated. The columns labeled with "peer max" in the Figure indicate the maximum percentage of suppressed signals on a per peer basis. Some peers suppress close to 45% of all signals received.

Figure 5.10 breaks down the suppressed signals between announcements and withdrawals. Since withdrawals typically generate more updates, naturally a much higher percentage of withdrawal signals are suppressed compared to announcement signals. In fact, at some peers, close to 90% of all withdrawal signals can trigger route suppression using Cisco's default setting. Overall, Cisco is more aggressive in suppressing routes than Juniper based on Route Views data for our three Beacon prefixes for both announcement and withdrawal signals. In fact, this may not always be the case. Although Cisco's cutoff threshold is lower compared to Juniper's, it does not punish route readvertisement or an announcement that is preceded by a withdrawal. Consequently, Cisco is more likely to suppress routes with the following update patterns: "AAAW" (A: announcement, W: withdrawal). There is no readvertisement in such patterns; therefore, Cisco's lower threshold will increase the probability of route suppression. Juniper is more likely to suppress routes with update

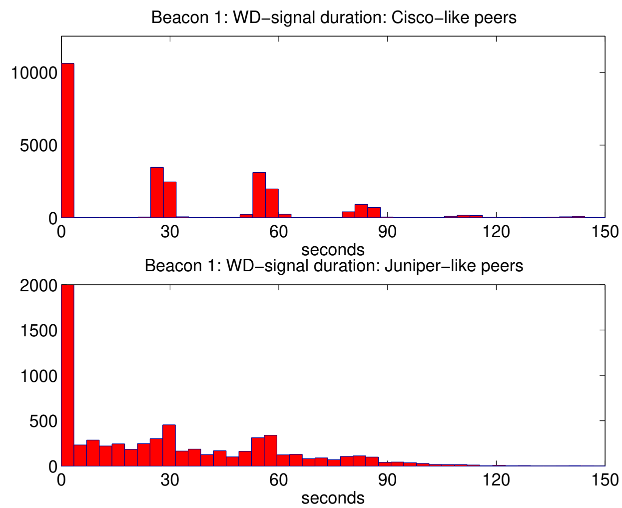


Figure 5.8: Comparing Beacon 1's withdrawal signal duration distribution for Cisco- like peers with that for Juniper-like peers

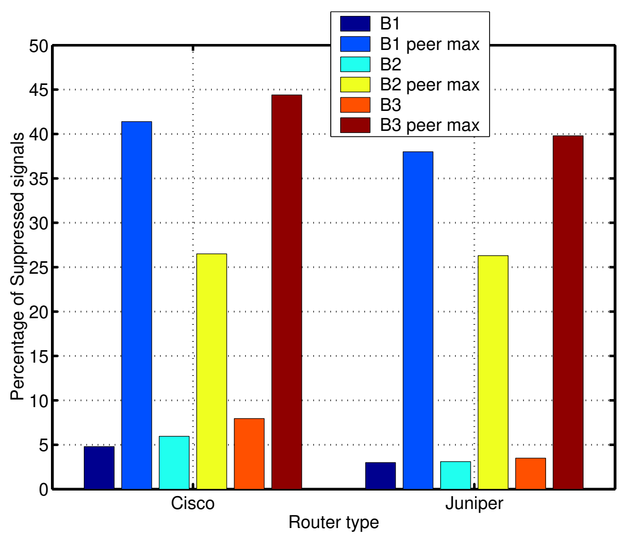


Figure 5.9: Overall percentage of suppressed signals due route flap damping for each Beacon and on a per peer basis for Cisco and Juniper.

patterns such as "AWAWA" where there are readvertisements. From our data, the former pattern is much more prevalent; therefore, Cisco is overall more aggressive in route suppression than Juniper.

Our previous analysis only provides a lower bound for the percentage of suppressed sig-

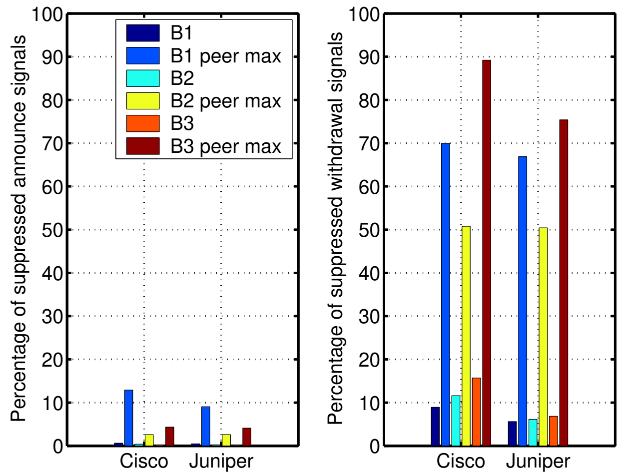


Figure 5.10: Percentage of suppressed Beacon signals due to announcement and withdrawal

nals, as some routers in the network may already have suppressed the Beacon prefix, resulting in possibly fewer updates observed at the monitoring site. Certain update patterns also indicate the presence of route suppression by some intermediate routers from a given monitoring point. For an announcement input signal, we sometime observe inter-arrival time between 1000 and 3600 seconds. After the long timeout, the update sequence always ends with an announcement. Such a long timeout is extremely unlikely to be caused by propagation delay, router processing delay, BGP path vector effects, or MinRouteAdverTimer values, even if they are accumulated along each router hop. Flap damping suppression duration is typically on the order of tens of minutes which match well with the timeout values we observe. In fact, based on the default Cisco parameter setting, the minimum suppression duration is about 30 minutes. For Juniper, the value is about 21 minutes.

If the data are cleaned properly and there are no missing updates, such timeout values provide a good indication that route suppression has occurred. The final announcement can be caused by the re-announcement of the route after its penalty has decayed below the reuse threshold. And the long break indicates the duration during which the route is suppressed. We show two such examples below (Tables 5.5, 5.6). In the first one, the last update before the long break is an announcement. In the second example, a withdrawal occurs before the long break. In both cases, the timeout value is about 40 minutes or 2400 seconds. We observe three transient routes from peer 216.18.31.102 in case 1. The final route goes through AS701, and it is very likely that the long break is due route flap damping occurring along the AS path “6539 701 1 3130 3927”. ASes 6539, 701 or 1 may have suppressed the route originated by AS3927. AS3130 could not have suppressed the route, because the alternate path used still goes through it. As soon as this route is unsuppressed, it is chosen as the preferred route by AS6539, which apparently has at least three alternate routes to reach the destination AS3927 where the Beacon prefix comes from. It very likely only suppresses the route learned from AS701; therefore, its connectivity to the Beacon is not affected. In general, if the last update before the timeout is an announcement, it indicates that there is an alternate path available from the monitoring point. In the case of withdrawal, flap damping has affected all the available paths from the monitoring location.

The latter is exemplified by case 2 shown in Table 5.6. The route with AS path “11608 2914 3130 3927” appears to be preferred over the alternate route with AS path “11608 2914 1239 3130 3927”. In this example, it is very likely that AS11608 suppresses the route received from AS2914. AS2914 is less likely to suppress the routes coming from both ASes 3130 and 1239, as this requires a router AS2914 to have received sufficient updates from both these neighbors. AS3130 is also unlikely to have suppressed the route from the origin AS 3927, as it is very close to the Beacon source and gets fewer updates. In general, ASes farther away from the origin AS is more likely to experience more updates and thus more likely to suppress the route.

In our data, we observed close to 1% such updates sequences indicative of route suppression in an intermediate router. We purposely separated consecutive updates with two hours, as the maximum suppress time is one hour. Therefore, for an announcement input signal, we should

Table 5.5: Case 1: observation from peer 216.18.31.102 on Apr 3 2003 for Beacon 1:

Time (GMT)	Type	AS Path
23:00:17	A	6539 3561 1 3130 3927
23:00:44	A	6539 701 1 3130 3927
23:01:14	A	6539 3602 16914 852 1 3130 3927
23:42:46	A	6539 701 1 3130 3927

Table 5.6: Case 2: observation from peer 207.246.129.14 on Sep 17 2002 for Beacon 1:

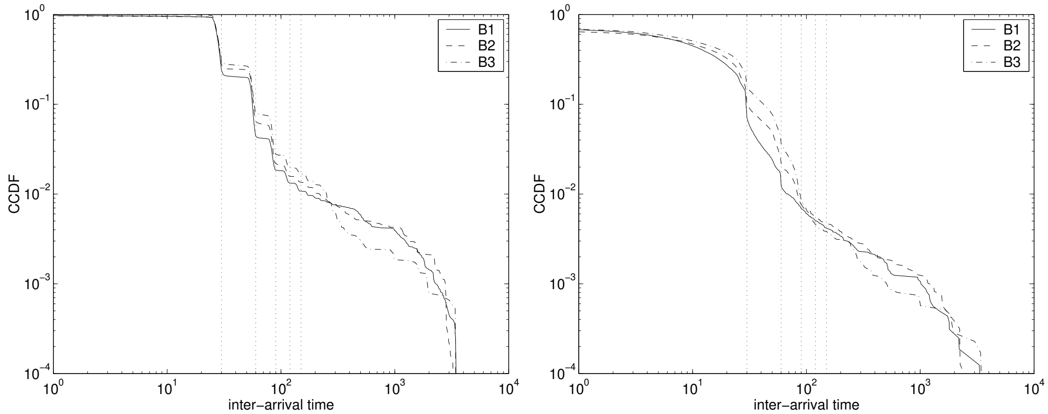
Time (GMT)	Type	AS Path
22:38:45	A	11608 2914 1239 3130 3927
22:39:13	A	11608 2914 3130 3927
22:39:40	W	
23:24:26	A	11608 2914 3130 3927

always expect to observe an ending announcement unless data are missing or anchor prefix is unstable. Similarly, withdrawal input signal should always result in an ending withdrawal in the output signals.

5.6 Inter-arrival Time Analysis

In this section we explore an aspect of BGP dynamics not considered in any previous work. A BGP update sequence has been typically considered (within this chapter as elsewhere) to be a sequence of N updates, over some duration, but little attention has been given to the distribution of events within this time interval. However, within this sequence the updates are spaced according to an *inter-arrival* distribution. We provide a brief examination of some of the properties and ramifications of this distribution as an example of how BGP Beacon data may used in the analysis of Internet routing dynamics.

As noted above, the inter-arrival times for Juniper-like and Cisco-like routers are different,



(a) Cisco-like last hop routers.

(b) Juniper-like last hop routers.

Figure 5.11: The inter-arrival time distribution for each of the three Beacons as seen from Cisco-like and Juniper-like routers. The vertical dotted lines are drawn at 30 second intervals.

and so we shall consider these separately here. First, consider the Cisco-like cases. Figure 5.11 shows log-log plots of the CCDF of the inter-arrival times for events (including announcements and withdrawals), for each of the three Beacons. The x-axis being the time between updates (in seconds), and the y-axis, the probability that an interval exceeds this time. Note that in the results here, intervals are rounded up to the nearest second, so that all intervals less than 1 second will appear as one second.

Despite the difference in the two graphs, we see two regimes in both. This is most clear in the distributions for the Cisco-like routers. For each of the Beacons we see a *body* region of step like decrease at slightly less than 30 second intervals (the vertical dashed line are drawn at exactly 30 second intervals). The 30 second intervals seems to match the typical value of the MinRouteAdverTimer described above. The second *tail* region seems to level out the distribution, followed by a sharp decrease, before the distribution is truncated at 3600 seconds by the data cleaning. The cut off between the two regions appears to be around 100 seconds. The Juniper-like routers show a similar

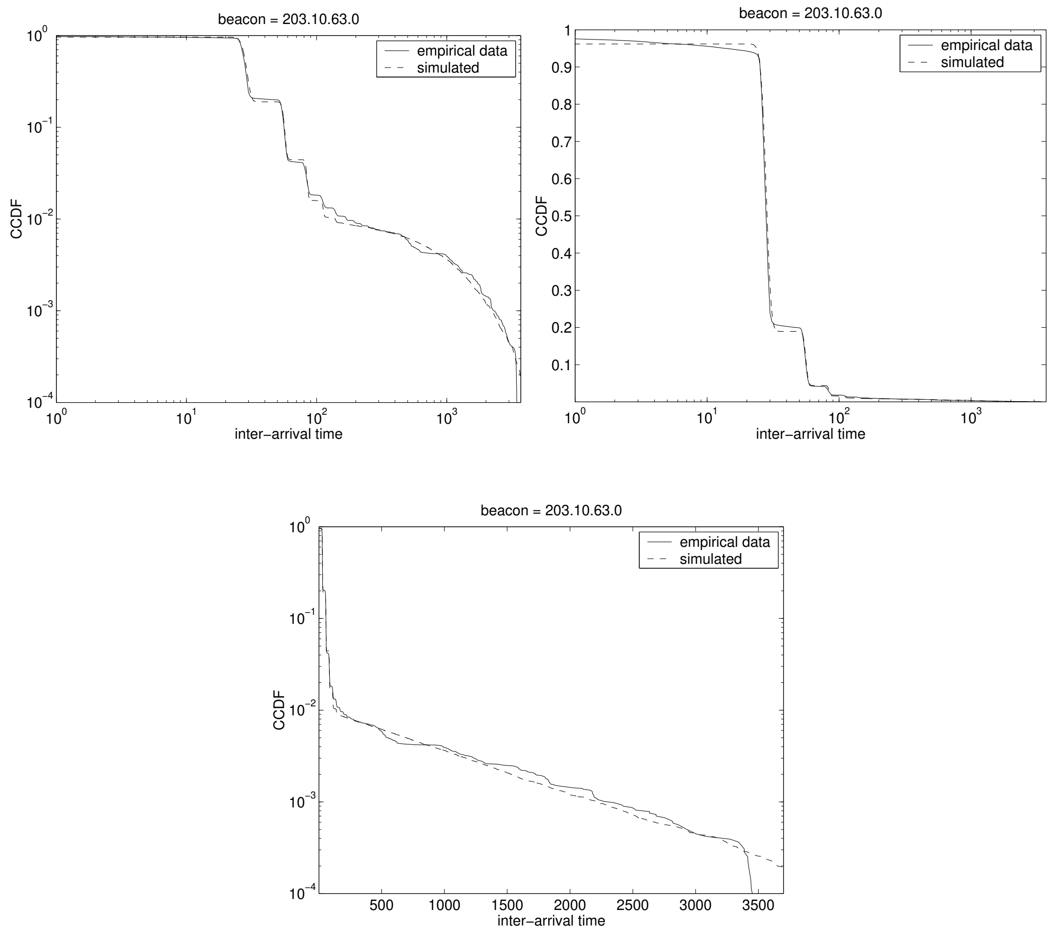


Figure 5.12: The inter-arrival time distribution for Cisco-like last hop routers, and Beacon 1

division of the distribution, though the body part is less step like.

A natural hypothesis to make is that the two components of this distribution arise from different basic causes. We shall examine this hypothesis by trying to understand what two processes might produce these results. Out first approach is to do some distribution fits to the data, to gain an understanding of what we are seeing.

Figure 5.12 shows one such fit (on loglog and semilog axes), done by eye for the Cisco-like routers. The fit combines three components. Firstly, we model the step function taking the number of steps to be given by a geometric distribution, and with the length of the steps to be 28 seconds (plus a small Gaussian jitter). The second component is a small mass at 1 second, because of the discretization of the interval times, in particular all times below one second are rounded to one. The third component is a shifted exponential distribution, which matches the tail of the distribution. The figure shows the fitted curve as the dashed line. Note that the fit on the loglog plot is very visually satisfying, as it is on both of the semilog graphs.

In this fitting we are not seeking to gain a precise knowledge of the parameters involved. In fact, given the number of parameters we have to play with here, the data are not sufficient to achieve a precise parameterization (one can always fit a sufficiently complex curve to any data set). The aim is to gain an understanding of the processes that might be involved by seeing what type of distributions they generate. However, for the benefit of the reader we provide a precise definition of the distributions and parameters used to produce the fitted distribution.

The distribution is generated using:

$$X = \begin{cases} 28 * (1 + \text{Geom}(0.81)), & \text{with probability } 0.9524, \\ 1, & \text{with probability } 0.0381, \\ 90 + \text{Exp}(970), & \text{with probability } 0.0095, \end{cases}$$

where $\text{Exp}(970)$ refers to an exponentially distributed random variable with mean 970 (seconds), and $\text{Geom}(0.81)$ refers to a geometric distribution with parameter $p = 0.81$, and therefore mean $(1-p)/p = 0.2346$.

Rather than try to consider the exact nature of the distribution above, let us try to understand the implications of this form of distribution. This type of distribution could arise as a result of a random mixing between three different random variables:

- **geometric distribution:** The first part of the distribution is generated by a series of steps, each nearly 30 seconds apart. The action of the MinRouteAdverTimer would certainly explain the first step – this timer prevents a router from sending an announcement within some time of the last prior announcement of the prefix. The typical default (rarely changed in practice) of a Cisco router is 30 seconds, and the router adds jitter to this amount to prevent any possible synchronization effects. Hence, one would naturally expect a delay of around 30 seconds between announcements – hence the first step in the distribution. The second and further steps can then be explained as multiple MinRouteAdverTimer intervals. We can suggest a simple reason why one might see such gaps: Cisco's implementation of the MinRouteAdverTimer is not 'per prefix', but rather 'per peer'. That is, the router will send a series of announcements to a peer, and then wait for the MinRouteAdverTimer Hence, an announcement which arrives 'late' due to delays in prior transmission (through a series of AS's) can miss the next batch of transmissions, and be delayed for a step. This can happen multiple times as an announcement traverses the network, and so we see multiple missed steps. The interesting thing is that this process can be so simply modeled by a geometric distribution, in which the probability of missing the next step does not depend on how many steps have already been missed.

- **mass at zero:** The discretization of timestamps to integer seconds results in discretization of the inter-event times – hence times between zero and one will tend to be lumped into one point at zero. Thus we need to include a probability mass at zero, which is indicative of the number of very short inter-arrival times.

- **shifted exponential:** This is the most puzzling part of the distribution, partly because we have the least data in this region (less than 1% of the distribution falls into the tail). The lack of a large data set, and the fact that these are truncated (by only using data sets for which the total time is less than one hour), means that one can model the tail almost as accurately using

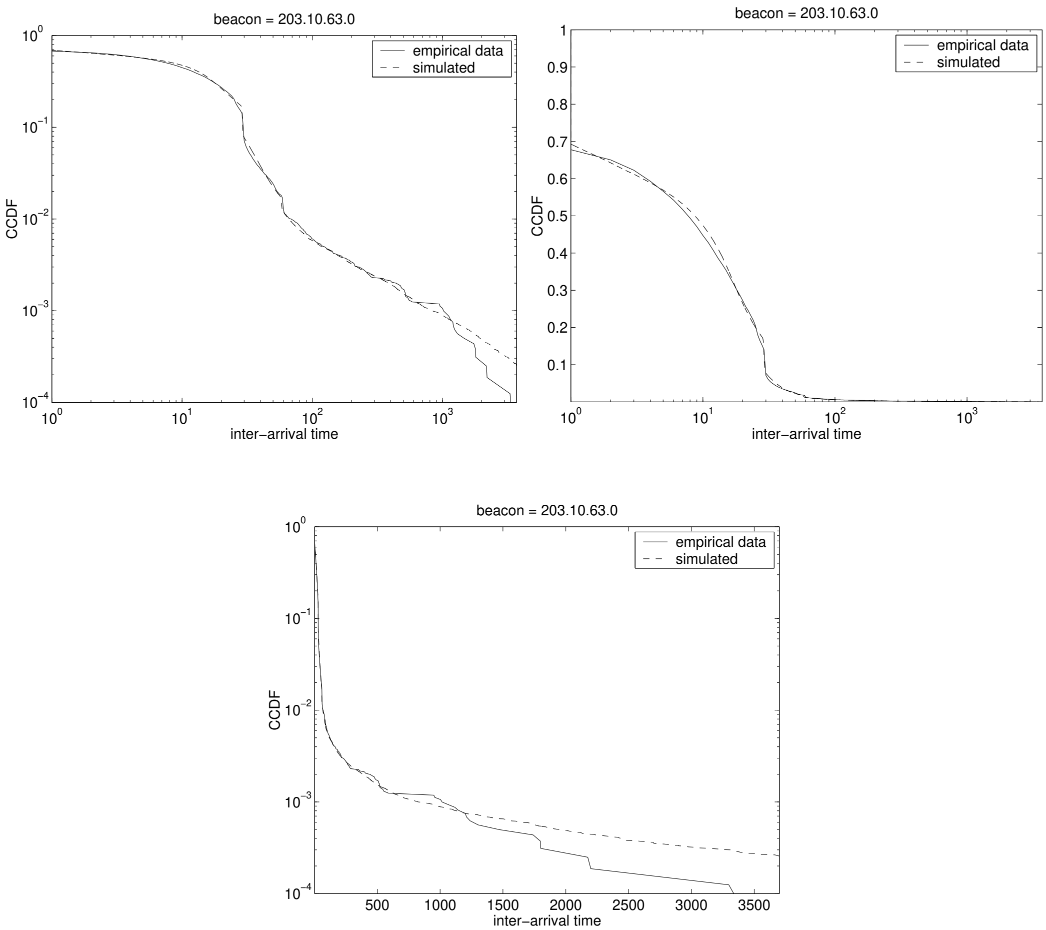


Figure 5.13: Empirical inter-arrival time distribution for Juniper-like routers comparing with simulated values

a power-law distribution. However, of the known BGP mechanisms, the most likely source of these delays is route flap damping, discussed above.

There is a natural test for the cause of the step like body of the distribution above. That is to consider the Juniper-like last hop routers, for whom we suspect the MinRouteAdverTimer is not used.

Figure 5.13 shows plots of the inter-arrival time distribution for the Juniper last hop routers (on loglog and semilog scales). This appears to be somewhat different from Figure 5.12. We can

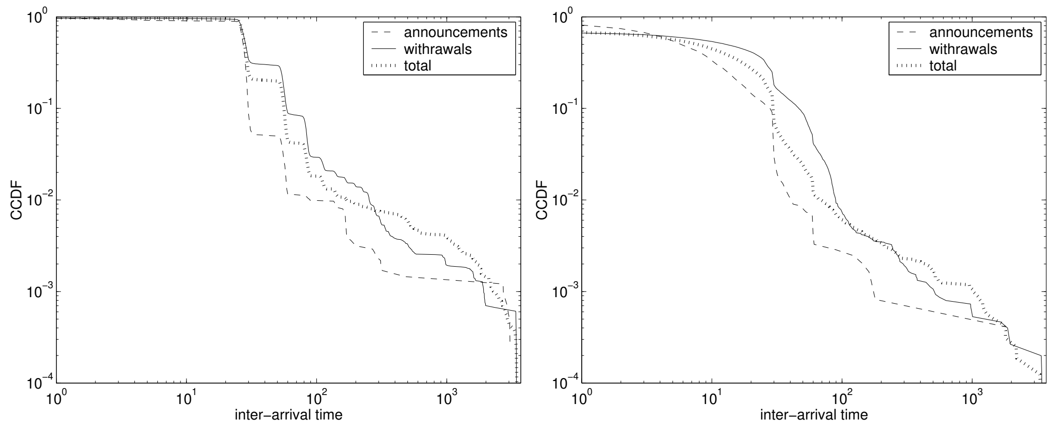


Figure 5.14: Inter-arrival time distribution for Juniper-like routers separated out by announcement and withdrawal signals

see much less evidence of a step-like decrease, and significantly more of the mass of the distribution appears before 30 seconds. Figure 5.13 also shows a fit to the distribution, though this time the fit has four components:

- a geometric distribution (step size ~ 30),
- mass at one (probability 0.1765);
- a convolution of a uniform (over $[0, 16)$) and exponential distribution (mean 10),
- a power-law tail (with $\alpha = 0.85$.).

In this case the much smaller (probability 0.0882), residual geometric component of the distribution can be easily explained by Cisco routers earlier in the path of the updates. The first part of the distribution, formed from the convolution of uniform and exponential distributions appears to be the fundamental difference between the two. It seems clear form these results the the MinRouteAdverTimer is responsible for the inter-arrivals times typically being multiples of approximately 30 seconds.

We display the power law fit to the tail for the Juniper-like routers, as in this case it appears to be a little better than the exponential tail, but note that it is not really possible to rigorously distinguish the two given the small amount of data in the tail, and the narrow range of scales across which it traverses. It will be interesting to study this as more data becomes available, to determine which model is better, as well as better determining the cause of this tail.

Finally, the above graphs lump two components together, the inter-arrivals for announcements, and for withdrawals. It might be possible that the two components seen are actually derived from these two separate types of events. In Figure 5.14 we show the separate distributions for announcements and withdrawals separated (and compared to the overall distribution). Note that the three curves all retain the same basic characteristics, though the curves for the announcement events both drop more sharply, and level off more than those for the withdrawals. The overall curve is more like the withdrawal curve, largely because withdrawal events generate more updates, and therefore more inter-arrival measurements.

The above analysis provides us with one more insight that might not be immediately obvious. Regardless of the form of the tail of the distribution, it can be considered a 'heavy-tailed' distribution in the sense that there is a significant probability of an event several orders of magnitude larger than the typical events (thousands of seconds as opposed to around 30). This is important because it immediately explains our earlier finding that the convergence times are not well correlated with the number of updates seen.

The sum of a series of heavy-tailed random variables is well known to also have a heavy-tail, but a less well known result is the fact that the heavy-tail of the sum arises not from a sum of medium size events, but from single large events. The intuitive explanation for this effect is that "rare events happen in the most likely way". In this context, we may interpret this to mean that the longer convergence times are not typically the result of a long series of updates that take a long time to converge, but rather the result of a single long inter-arrival time between updates. The data seem to validate this intuition. If removing the heavy-tail from convergence times is considered important, then one must first concern oneself with the causes of the heavy-tail in the inter-arrival time (for instance flap damping), rather than trying to reduce the number of updates.

5.7 Convergence Redux

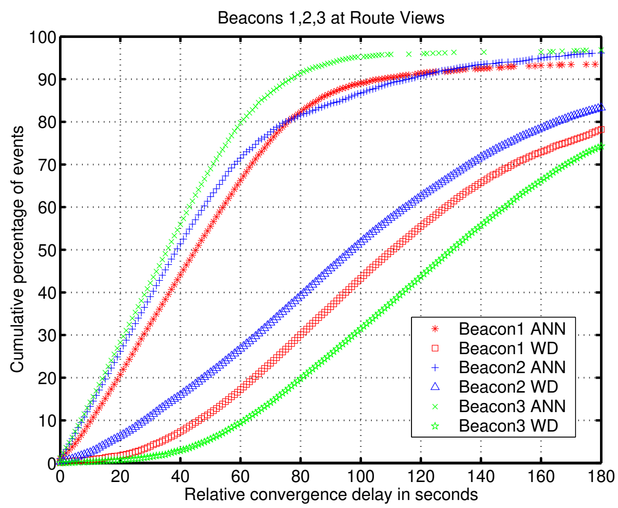


Figure 5.15: Cumulative distribution of relative convergence times for all three Beacons for both announcement and withdrawal signals

Given the insight we gained from our previous three studies, we now revisit the work by Labovitz *et al.* [63] conducted about three years ago. We go into more details as needed here than when we first described these studies in Chapter 1. They analyzed BGP convergence behavior of four types of events: announcement (Tup), withdrawal (Tdown), fail over to a shorter route (Tshort), fail over to a longer route (Tlong). We only focus on the first two cases and leave it to future work to study the latter two cases. To study Tshort and Tlong, we need to modify our experiment setup to inject a withdrawal to one of the upstream ASes rather than to both ASes in the case that the Beacon

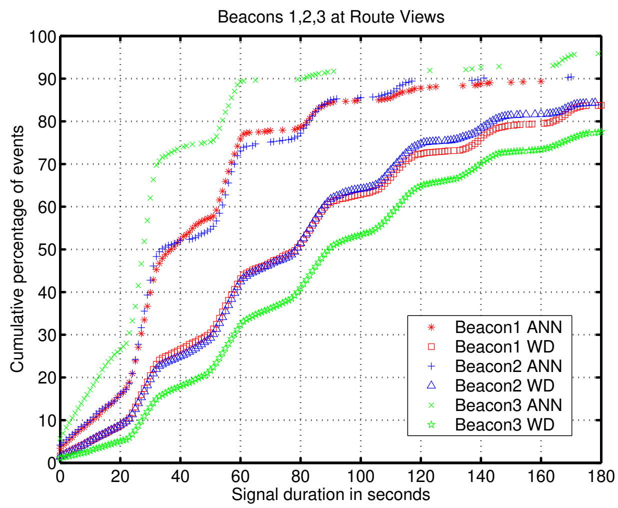


Figure 5.16: Cumulative distribution of signal duration for all three Beacons for both announcement and withdrawal signals

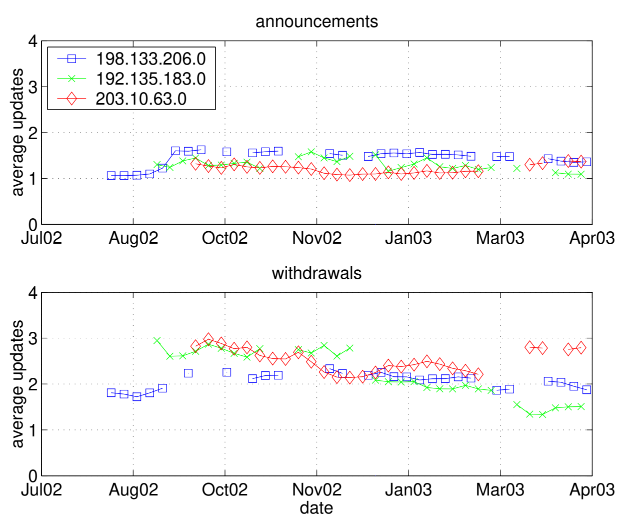


Figure 5.17: Variation in average signal length over time (Beacons 1,2,3).

is multihomed. As pointed out by Labovitz, the observed behavior of Tshort is very similar to Tup and that of Tlong is quite similar to Tdown.

Figures 5.15 and 5.16 present a cumulative distributions of the relative convergence times

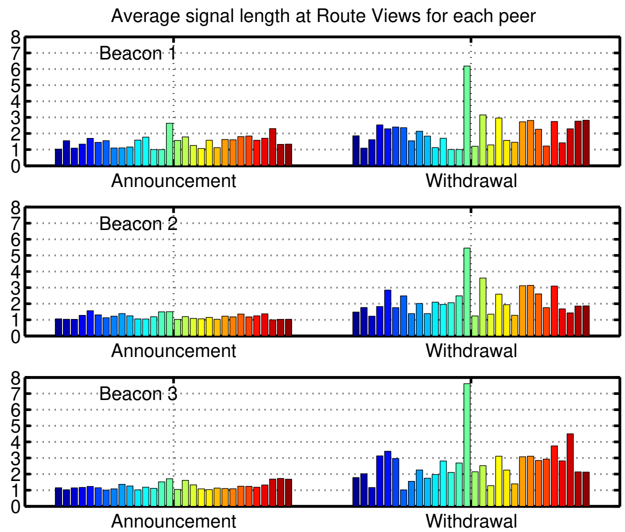


Figure 5.18: Average signal length for each peer

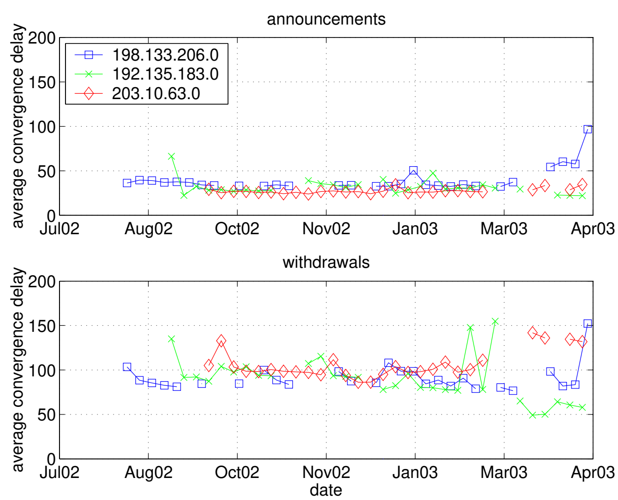


Figure 5.19: Variation in average relative convergence delay over time (Beacons 1,2,3).

and signal durations, for PSG Beacons 1, 2 and 3. These results are entirely consistent with the results of [63], showing that these characteristics appear not to have changed significantly in the last few years. Our analysis is consistent also with a preliminary study of the RIPE Beacons, recently

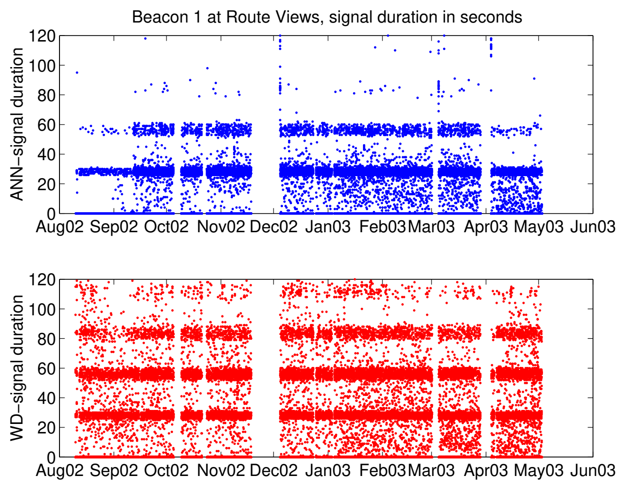


Figure 5.20: Beacon 1's signal duration variation over time, cutoff at 120 seconds. Max duration is 3525 seconds.

presented [104].

Figure 5.17 present the variations over time in signal length. These averages mask the wide variations seen among peers, which is presented in Figure 5.18. As in [63], we see more updates in signals associated with announcements and withdrawals. Figure 5.19 shows the variation in the relative convergence delay over time — there is large amount of variation but no clear trend.

Figure 5.20 illustrates an interesting time series of Beacon 1's duration during the course of our study. The gaps indicate the time periods during the Beacon was down. In the month of August 2002, the Beacon was single-homed: with only one upstream provider AS2914. In September 2002, it became multi-homed to AS2914 and AS1. There is little change for withdrawal signal duration; however, the duration doubled from around 30 seconds to 60 seconds. Upon further analysis, we found that the average announcement signal length also doubled from around 1 to 2. After Beacon 1 is multihomed, the announcement signal is very likely to explore the alternate less preferred route first before settling on the final route. There is another interesting change occurred in

April 2003, during which one of Beacon 1's upstream providers is changed from AS1 to AS1239. Apparently AS1239 seems to be better connected than AS1 and resulting in shorter announcement signal durations.

5.8 Summary and Open Problems

This chapter describes an active BGP measurement infrastructure consisting of a set of BGP Beacons that have been set up for public use, along with techniques for obtaining clean and useful data from these Beacons. Used in conjunction with public route monitors they provide a mechanism for performing controlled experiments with global Internet routing. We present several examples of how data from such experiments may be used to understand BGP routing dynamics. Among the examples, we validated our conjecture in the previous chapter, Chapter 4 that route flap damping can significantly delay BGP convergence. This is a clear example of how active measurements on the actual Internet with controlled routing input can reveal insight into complex BGP dynamics. BGP Beacons is the first public active measurement infrastructure for studying interdomain routing dynamics. It provides an venue for bringing more visibility into the BGP run-time behavior and the correlation between BGP and the data plane behavior. To perform the correlation studies, we describe in the next chapter a tool that provides *AS-level* forwarding path information, a necessary requirement for the correlation.

Chapter 6

AS-level Traceroute: Correlating BGP With The Data Plane

Chapter 5 describes an active measurement infrastructure for studying BGP dynamics where experimenters can precisely control injected input routing changes. Such an infrastructure is very valuable for understanding the root causes of observed BGP update sequences. Since the routing infrastructure ultimately serves the purpose of providing routes for the packet forwarding plane or the data plane, its effectiveness should be measured in terms of application performance. As shown in Chapter 3, where our research methodology was described, the third component of our framework for understanding BGP dynamics is correlating BGP with traffic measurements. To perform such correlation, BGP paths and data paths need to be compared. However, data paths are typically in the form of hop by hop IP-level paths. Such IP-level forwarding paths need to be translated to AS-level paths to be compared with BGP paths. This chapter describes a tool for characterizing the AS-level forwarding paths of data packets based on both traceroute and BGP data to address the following two challenges: BGP may not be accurate in predicting the forwarding paths of data packets; the traceroute path is difficult to be directly translated to an AS path. The contribution of this work is to combine both information with additional information such as DNS, whois database, BGP data from multiple vantage points to more accurately predict a packet's forwarding AS paths. This in turns allows us to correlate the routing plane with the data plane to understand the BGP dynamics in the context of improve application performance.

The chapter follows this organization. In Section 6.1 we introduce the motivation for building an accurate AS-traceroute tool for the purpose of characterizing the forwarding behavior of routers, explain why simple approaches of using whois, BGP paths do not work, and give an overview of the measurement methodology. Complementing related work in Chapter 2, we also describe some measurement studies relevant to our work of developing an AS-traceroute tool. Section 6.2 describes in detail how BGP and traceroute data are used to generate the initial IP-to-AS mappings. In the two subsequent sections: Sections 6.3 and 6.4 we analyze the traceroute paths and resolve the incomplete paths due to either traceroute problems of inaccurate IP-to-AS mappings. Three sets of heuristics (IXP, sibling and shared address space inference) described in Section 6.5 constitute the first set of techniques to obtain accurate IP-to-AS mappings of infrastructure addresses, critical to developing an accurate AS-traceroute tool. Section 6.6 presents an initial analysis of the cases where the forwarding paths and BGP paths do not match due to routing anomalies and various operational practices. The rest of the chapter, Sections 6.7, 6.8, 6.9, 6.10 present a more systematic approach of improving the IP-to-AS mappings assuming that most forwarding paths match BGP paths and the mismatches can be explained by inaccurate IP-to-AS mappings. We take an approach of iteratively applying a dynamic programming algorithm to reduce the amount the mismatches between BGP and traceroute paths. A summary of the chapter is presented in Section 6.12.

6.1 Introduction

Network operators and researchers would benefit greatly from an accurate tool for reporting the sequence of Autonomous Systems (ASes) along the path to a destination host. Designing a useful "AS-level traceroute" tool depends on having an accurate way to map the IP addresses of network equipment to the administering ASes. This problem is surprisingly difficult and existing approaches have major limitations, due to the operational realities of today's Internet. We propose a way to improve the IP-to-AS mapping of the infrastructure by comparing traceroute and BGP (Border Gateway Protocol) paths collected from multiple vantage points. This improved IP-to-AS mapping can be used as seed input for a tool that maps traceroute output to an AS-level path.

6.1.1 Motivation for AS Traceroute

Traceroute [60] is widely used to detect and diagnose routing problems, characterize end-to-end paths through the Internet, and discover the underlying network topology. It identifies the interfaces on a forwarding path and reports round-trip time statistics for each hop along the way. Despite its many well-documented limitations, traceroute is the only effective way to determine how packets flow through the Internet without real-time access to proprietary routing data from each domain. The tool is invaluable for network operators in identifying forwarding loops (loops in the data plane), blackholes (traffic not delivered to destinations), routing changes, unexpected paths (forwarding and routing path mismatch) through the Internet, and, in some cases, the main components of end-to-end latency. Researchers rely heavily on traceroute to study routing protocol behavior [85], network performance [96], and the Internet topology [13, 21, 45, 99].

In addition to the IP forwarding path, operators often need to know which ASes are traversed en route to the destination. Upon detecting a routing or performance problem, operators need to identify (and notify!) the responsible parties—often their compatriots in other ASes. This is a crucial part of diagnosing and fixing problems that stem from misconfiguration of the routing protocols [70] or serious equipment failures. For example, suppose that customers complain that they cannot reach a particular Web site. The operator could launch traceroute probes toward the destination and determine that a forwarding loop is to blame. However, correcting the problem requires a way to determine which AS (or set of ASes) has routers forwarding packets in the loop. Inaccurate information leads to delays in identifying and correcting the problem.

Researchers use the AS path information to construct AS-level views of the Internet topology [17] and to study the properties of the AS paths traversing this graph [102]. These AS-level "outputs" are used as "inputs" to research in a variety of areas, such as the placement and selection of Web content replicas. The accuracy of these studies hinges on having a sound way to determine the sequence of ASes on a forwarding path. However, a recent paper [19] demonstrated that the various techniques for identifying the AS-level forwarding path lead to very different results for basic properties of the Internet topology, including path asymmetry and node degree. Having a good AS-level traceroute tool in the research community would make these studies more accurate. In addition, new research could focus directly on the properties of the AS-level forwarding path, such as identifying the ASes most responsible for forwarding anomalies and performance problems.

6.1.2 Difficulty of the Problem

Identifying the ASes along the forwarding path is surprisingly difficult. Contrary to common wisdom, BGP path does not necessarily reflect the actual forwarding paths due to complex BGP dynamics resulting in routing anomalies. Traceroute infers the path by transmitting a sequence of TTL-limited packets and extracting the interface IP addresses from ICMP responses sent by the hops along the way. However, some hops do not return ICMP replies, and successive TTL-limited packets do not necessarily follow the same forwarding path. Mapping the IP-level hops to AS numbers is complicated and existing approaches have major limitations:

BGP AS path: A seemingly natural way to determine the AS path is to observe the routes learned via BGP. However, timely access to BGP data is not always possible from the vantage point of interest. Perhaps more importantly, BGP provides the *signaling* path (the list of ASes that propagated the BGP update message), which is not necessarily the same as the *forwarding* path (the list of ASes traversed by data packets). Although the two AS paths usually match, they may differ for various reasons such as route aggregation/filtering and routing anomalies [53]. In fact, the two paths may differ *precisely* when operators most need accurate data to diagnose a problem.

Internet route registry: Instead, the ASes in the forwarding path can be derived directly from the traceroute data by associating each traceroute hop with an AS number. The popular "NANOG traceroute" [7] and prtraceroute [9] tools perform *whois* queries to map each interface to an address block allocated to a particular AS. However, *whois* data are often out-of-date or incomplete, since institutions do not necessarily update the database after acquisitions, mergers, and break-ups, or after allocating portions of their address blocks to customers.

Origin AS in BGP routes: A more accurate and complete IP-to-AS mapping can be constructed from BGP routing tables by inspecting the last AS (the "origin AS") in the AS path for each prefix [23]. However, some traceroute hops map to multiple origin ASes (MOAS) [110] or do not appear in the BGP tables. The notion of "origin AS" blurs the many reasons why ASes introduce prefixes into BGP. In addition to originating routes for its own infrastructure, an AS may inject routes on behalf of statically-routed customers. Some ASes do not advertise their infrastructure addresses and others may announce the addresses of shared equipment at boundary points between domains. As a result, some traceroute AS paths appear to have AS loops, or extra or missing hops relative to the corresponding BGP paths.

In this chapter, we identify the root causes of the differences between the traceroute and BGP AS paths, and propose techniques for identifying the "real" AS-level forwarding path. This is critical to understanding the effect of BGP dynamics on the data plane. Unexpected forwarding paths can give us insight into routing anomalies such as *routing deflection* where a router's chosen forwarding path to an egress point to be deflected by another router on that path [53].

6.1.3 Our Approach to the Problem

In practice, the signaling and forwarding AS paths do not always agree, due to route aggregation and forwarding anomalies. However, we believe that most discrepancies between the BGP and traceroute AS paths stem from inaccuracies in the IP-to-AS mapping applied to the traceroute data. We propose to improve this mapping by comparing BGP and traceroute paths from multiple vantage points. Our algorithms analyze measurement data to identify cases where a single "explanation" would account for the differences between many pairs of BGP and traceroute AS paths. These explanations build on an understanding of common operational practices, such as the presence of Internet eXchange Points (IXPs), where multiple ASes connect to exchange BGP routes and data traffic. The results of our algorithms are used to tune an initial IP-to-AS mapping derived from the BGP routing tables. We envision this as a continuous process where traceroute and BGP data are collected from many vantage points and used to compute an accurate IP-to-AS mapping as it changes over time. An AS traceroute tool running on end hosts would periodically download the latest IP-to-AS mapping and use it to compute and display the AS path associated with each traceroute probe the user launches. The chapter makes five main contributions toward this end:

Measurement methodology: Our techniques depend on collecting traceroute probes, BGP update messages, BGP routing tables, and reverse DNS lookups, as discussed in Section 6.2.

Traceroute analysis: In Section 6.3, we analyze traceroute and BGP paths from eight locations, and construct an initial IP-to-AS mapping from BGP routing tables. Then, we present an initial comparison of the BGP and traceroute AS paths.

Resolving incomplete paths: Section 6.4 presents three simple techniques for resolving most traceroute hops that do not map to an AS number. We also introduce our approach of using internal router configuration data for checking our results.

Improved IP-to-AS mapping: Many mismatches between BGP and traceroute paths can be explained by IXPs, sibling ASes managed by the same institution, and ASes that do not advertise routes to their equipment. Section 6.5 proposes techniques that identify and "fix" some of these cases.

Legitimate mismatches: The traceroute and BGP AS paths may differ for valid reasons such as route aggregation, interface numbering at AS boundaries, the choice of source address in ICMP, and routing anomalies. Section 6.6 discusses how these factors may explain some of the remaining differences between the traceroute and BGP AS paths.

Validating our techniques is difficult without knowing the actual AS-level forwarding paths. Where possible, we compare results with publicly-available data, such as *whois* data and lists of known IXPs. We conclude in Section 6.12 with a summary of our contributions and a discussion of ongoing work.

6.1.4 Related Work

Recent measurement studies have quantified the differences between BGP and traceroute AS paths. The analysis in [19] showed that these differences have a significant impact on the characterization of the Internet topology. In parallel with our work, the work in [57] used publicly-available data (such as *whois*, lists of known IXPs, and other Web sites) to test the hypothesis that many of the mismatches stem from IXPs and siblings; in contrast, our work proposes heuristics for *identifying* IXPs, siblings, and other causes of mismatches to improve the IP-to-AS mapping. To improve the accuracy of AS graphs derived from traceroute, the work in [31] proposed techniques that identify border routers between ASes to correct mistaken AS mappings; this is an alternate approach that handles some of the inaccuracy introduced by IP-to-AS mappings derived from BGP tables. Traceroute data have been used in other studies that measure router-level topologies and map routers to ASes [45, 99]. Except for handling certain traceroute anomalies such as unmapped IP address, these studies did not focus on improving the accuracy of the IP-to-AS mapping derived from the BGP routing tables. Focusing solely on BGP AS paths, the work in [42, 101] presented algorithms for inferring AS-level commercial relationships, including siblings; however, these studies did not consider the influence of sibling ASes on the accuracy of traceroute AS paths.

In contrast to previous work, our work focuses on automated techniques for improving the IP-to-AS mapping applied to the traceroute paths. Although we use publicly-available information for validation purposes, the techniques we propose do not depend on the availability of such data. Our work capitalizes on traceroute paths and BGP updates collected from multiple vantage points to a large number of destinations throughout the Internet. The techniques we apply to pre-process the measurement data limit possible inaccuracies from transient routing changes and unmapped hops in the traceroute paths. Our algorithms for identifying IXPs, siblings, and unannounced infrastructure addresses allow us to produce a more accurate estimate of the AS-level forwarding path from the raw traceroute data. This, in turn, enables us to focus our attention on the legitimate mismatches between the AS-level signaling and forwarding paths.

6.2 Measurement Methodology

This section presents our methodology for collecting traceroute and BGP paths from multiple vantage points, as shown in Figure 6.1. Taking advantage of data from multiple locations is part of our general research methodology described in Chapter 3. This strategy provides more complete view of the data, reduces any bias due to limited topologies. We select candidate prefixes and ultimately individual IP addresses to cover the routable address space. For each prefix we measure the forwarding path with traceroute and extract the BGP AS path from the routing table of the border router. We discard data for cases where the BGP AS path cannot be meaningfully compared with the traceroute path. We compute an AS-level traceroute path by mapping traceroute hops to AS numbers using the origin ASes extracted from a large set of BGP routing tables.

6.2.1 Selecting Candidate IP Addresses

Starting with a list of routing table entries, we first identify the prefixes that cover the routable address space and then select two IP addresses within each prefix for traceroute probing.

Select prefixes: Ideally, we would like to learn the forwarding path to each live destination address from each vantage point. However, identifying all live IP addresses is challenging and sending traceroute probes to each destination would be prohibitively expensive. Instead, we select a set of prefixes that cover the routable address space to sample a wide range of forwarding and signaling paths. For each vantage point, we extract a list of prefixes from the BGP routing table of the (single) border router that connects this site to the Internet. However, some prefixes are never used to route traffic because of more specific subnets in the routing table. For example, no packet would use the 192.0.2.0/23 route if nested entries for 192.0.2.0/24 and 192.0.3.0/24 were available, due to the longest-prefix match forwarding paradigm. Other prefixes may be partially covered by subnets. For

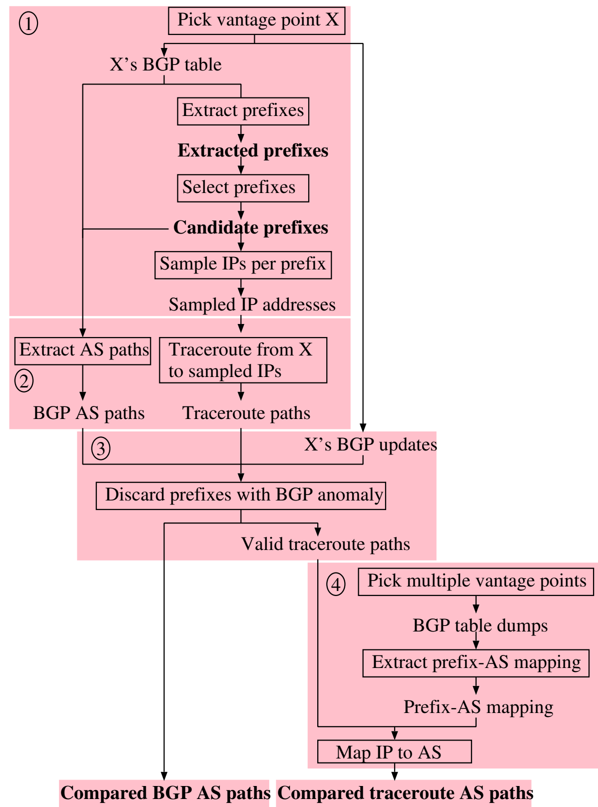


Figure 6.1: BGP and traceroute data collection.

example, a table with routes for 8.0.0.0/8 and 8.128.0.0/10 would only use the 8.0.0.0/8 routing entry for destinations in 8.0.0.0/9 or 8.192.0.0/10; all other addresses in 8.0.0.0/8 would match 8.128.0.0/10. To identify these cases, we sort the list of prefixes based on the numerical values and mask length; this ensures that each prefix is followed immediately by all of its subnets. For each prefix, we identify the portion of the address space that would match this routing table entry and represent it as a list of address blocks. The algorithm runs in $O(n^2)$ time in the worst case that all n routing entries are subnets of a single prefix. In that case, the difference between any prefix and all its preceding prefixes in the sorted order is calculated. Prefixes like 192.0.2.0/23 that are covered by their subnets do not correspond to any portion of the address space, and are excluded from the candidate prefixes.

Sample IP addresses: Each candidate prefix has one or more IP addresses that match the routing entry using longest prefix matching. We select two IP addresses for each prefix for the sake of comparison; this is especially useful for studying the effects of route aggregation and filtering. Limiting ourselves to two addresses reduces the time required to collect the data. For each prefix, we arbitrarily select the first address block in the representation computed by our algorithm (*e.g.,* 8.0.0.0/9 for {8.0.0.0/9, 8.192.0.0/10}). We select the two IP addresses from the beginning and the middle of the block. That is, for a block with address Q and mask length N, we select IP addresses "$Q + 1$" and "$Q + 2^{32-N-1} + 1$" (*e.g.,* 8.0.0.1 and 8.64.0.1). Note that the addresses do not necessarily correspond to live hosts; some may be unused or assigned to parts of the infrastructure.

6.2.2 Obtaining Traceroute and BGP Paths

After selecting the IP addresses, we obtain both the traceroute and BGP paths from each vantage point.

Collect traceroute paths: We configure the traceroute software to send a single UDP packet for each TTL value and wait two seconds for an ICMP reply. In lieu of sending multiple packets per TTL value, we modified the traceroute source code to send a second packet only if the first attempt did not produce an ICMP response; for the second packet, we wait five seconds for a reply. To further reduce the overhead, we apply the modified traceroute with DNS resolution disabled; after completing the traceroute experiments, we perform a reverse DNS lookup for each unique IP address that appears in the traceroute output. For each destination address, we record a timestamp and the traceroute output.

Extract BGP AS paths: For each candidate prefix, we extract the corresponding AS path from the most recent (daily) dump of the local BGP table that occurred before the traceroute. If the table has no BGP route for the prefix (say, due to a BGP withdrawal since the initial table dump), we extract the AS path of the longest matching prefix. We preprocess the BGP AS path to collapse consecutive repeating ASes (*e.g.,* converting "701 88 88" to "701 88") that stem from AS prepending.

6.2.3 Discarding Based on BGP Properties

In some cases, comparisons between the BGP and traceroute paths are not meaningful, and we discard both paths:

BGP routing changes: Routing changes introduce uncertainty in the local BGP AS path during the period of a traceroute experiment. Starting with the most recent BGP table dump, we apply the sequence of update messages to track changes in the BGP AS path for each prefix over time. After a traceroute completes, we identify the most recent BGP route update for that prefix and use this AS path in our subsequent analysis. We also inspect a window of time before and after the traceroute for update messages for the prefix. If a BGP routing change occurs during this window, we exclude this prefix from our analysis. In our study, we apply a 30-minute window before and after each traceroute to account for delays in BGP routing convergence [64]. Still, we cannot ensure that the *forwarding* path remains the same throughout the traceroute experiment. For example, the forwarding path may fluctuate due to an *intra*domain routing change. In addition, some downstream AS might experience a BGP routing change for some subnet of the prefix that is not seen at local collection point. We can only ensure that the BGP AS path seen at our collection point is stable during the traceroute experiment.

Null AS paths: Each BGP table has a few routes with a null AS path. These routes correspond to prefixes belonging to the institution where we collected the BGP and traceroute data.

Private AS numbers: Some BGP AS paths contain private AS numbers in the range of 64512–65535. This can arise when a customer (using a private AS number) mistakenly leaks BGP routes learned from one upstream provider to another.

Apparent AS loops: BGP has a loop-detection mechanism where a router discards a route when its own AS number appears in the AS path. However, apparent AS-level loops can arise if a router is (mis)configured to prepend an arbitrary AS number that already appears elsewhere in the AS path.

AS_SET: In the usual case, the AS path information is encoded as a sequence of ASes. However, occasionally, when a router aggregates multiple BGP routes, the resulting AS path may include an *unordered* set of ASes from the original paths (to prevent loops). This makes it impossible to determine the sequence of ASes in the path(s).

6.2.4 Computing Traceroute AS Paths

Computing the AS-level traceroute path requires mapping the IP addresses in the path to AS numbers. We construct an initial mapping by combining BGP routing tables from multiple locations and extracting the last hop in the AS path (the "origin AS") for each entry. An individual IP address is mapped to the longest matching prefix. If a prefix has routes with multiple origin ASes (MOAS), we map the IP address to the *group* of ASes. After mapping each traceroute hop to an AS (or group of ASes), we collapse hops with the same mapping to produce the AS-level traceroute path.

6.3 Traceroute Analysis

In this section, we apply our measurement methodology to traceroute and BGP routing data collected from eight sites. We analyze the diverse ways the traceroute experiments can *end* and explain how we preprocess the data. We quantify the limitations of using Internet routing registry data to map the IP addresses to AS numbers and evaluate our approach of using BGP routing tables

Organization	**Location**	**Dates in 2003**	**Upstream Provider (AS Number)**
AT&T Research (AS 6431)	NJ, USA	June 6-9	UUNET (701), AT&T (7018)
UC Berkeley (AS 25)	CA, USA	June 6-8	Qwest (209), Level 3 (3356)
PSG home network (AS 3130)	WA, USA	April 30 - May 8	Sprint (1239), Verio (2914)
Univ of Washington (AS 73)	WA, USA	June 4-8	Verio (2914), Cable & Wireless (3561)
ArosNet (AS 6521)	UT, USA	May 1-6	UUNET (701)
Nortel (AS 14177)	ON, Canada	May 1-6	AT&T Canada (15290)
Vineyard.NET (AS 10781)	MA, USA	June 4-9	UUNET (701), Sprint (1239), Level 3 (3356)
Peak Web Hosting (AS 22208)	CA, USA	May 1-8	Level 3 (3356), Teleglobe (6453) Global Crossing (3549),

Table 6.1: Traceroute probing locations

collected from multiple vantage points. Still, many of the hops in the traceroute paths have IP addresses that map to multiple ASes or do not appear in the BGP table.

6.3.1 Collecting Traceroute and BGP Updates

We collected detailed routing data from eight locations in the North America, as summarized in Table 6.1. The sites were chosen based on their topological diversity and our ability to collect both traceroute and BGP update data. At each location, we ran traceroute on machines one or more hops behind a single border router; the traceroute data were preprocessed to remove the initial hops between the probe machine and the border router. In AS 6431 and AS 25 we had root access to a Linux machine that ran our modified traceroute software to send a second TTL-limited probe upon receiving a "*" response from an intermediate hop; sending a second packet resulted in a successful ICMP reply in 7% of the cases. In other locations, we used a standard traceroute configured to send a single probe for each hop to reduce overhead and delay. At each site, we collected BGP updates in MRT format through a BGP session with the border router, along with daily dumps of the BGP routing table. At each location, the machines sending traceroute probes and logging the BGP updates had their clocks synchronized using NTP. For brevity, we present the results from the first three locations only; the results from other locations are similar.

Despite the topological diversity of our measurement points, our analysis would benefit from a larger number of data sets from different countries. On the surface, using the publicly-available traceroute servers would seem like a natural solution to this problem. However, BGP update messages are not available from these servers, although some support querying of the BGP routing table; this would have allowed us to poll a prefix's BGP route a few minutes before and after each traceroute experiment, in the hope of catching relevant BGP routing changes. However, the public traceroute servers typically impose a rate limit on requests issued from the same host, making it difficult to probe a large number of addresses in a reasonable amount of time. The long delay could span significant changes in the Internet topology and in the mapping of prefixes to ASes. In addition, the GUIs at the public servers typically do not support changes to traceroute parameters (*e.g.,* number of probes per hop, timeout for ICMP replies, and disabling DNS resolution). As such, although our methodology can be applied to an arbitrary number of vantage points, the analysis in this chapter focuses on a smaller number of data-collection points under our direct control; where relevant, we comment on how the limited vantage points may affect our results.

Our analysis focuses on one set of traceroute experiments from each location; results from other dates produced very similar results. The eight traceroute data sets were collected between May and June in 2003. Table 6.2 reports the number of prefixes *extracted* from the local BGP routing table at the first three sites, following the steps outlined earlier in Figure 6.1. The table also lists the

	AS 6431	AS 25	AS 3130
Extracted	121259	124295	120996
Candidate	119550	122487	119340
Compared	118345	112120	117195

Table 6.2: Number of prefixes in the three datasets

	AS 6431	AS 25	AS 3130
Routing changes	0.3802%	0.5809%	0.3105%
Null AS paths	0.0058%	0.0064%	0.0000%
Private ASes	0.0000%	0.0000%	0.0008%
AS loops	0.0000%	0.0000%	0.0155%
AS_SET	0.0214%	0.0233%	0.0248%

Table 6.3: Prefixes excluded due to BGP properties

number of *candidate* prefixes used in the traceroute experiments (with two destination addresses per prefix), after applying the algorithm in Section 6.2.1; the other 1.3–1.4% of the BGP prefixes were not the longest matching route entry for any destination addresses. The *compared* prefixes excludes the cases where comparisons with the BGP AS paths were not meaningful. Table 6.3 presents a more detailed breakdown of the five cases, which account for less than 1% of the prefixes probed in the traceroute experiments. The rest of the candidate prefixes that are not compared are due to BGP table changes causing some long prefixes to disappear and failed traceroutes caused by routing problems. The compared prefixes form the basis of the analysis in the remainder of the chapter.

6.3.2 Characterizing the Traceroute Results

Ideally, traceroute returns a complete list of IP addresses up to and including the destination. This requires each hop to return an ICMP TIME_EXCEEDED message with the address of the corresponding interface and the destination host to return a PORT_UNREACHABLE message.

In practice, the traceroute paths *end* in five different ways, as summarized in Table 6.4:

Expected final address: Only around 11% of the paths end with a PORT_UNREACHABLE message from the target IP address. In a way, this is not surprising because the destination address does not necessarily correspond to a live machine and some networks have firewalls that discard the UDP traceroute probes. Still, around 95% of the traceroute paths reach an address with the same origin AS as the target destination.

Unexpected final address: About 15% of the paths end in less than 30 hops (the default maximum number) with an address that differs from the intended destination. This can occur when the destination is a device (such as a router) that has multiple interfaces with different IP addresses, or if an intermediate component (such as a firewall) sends a PORT_UNREACHABLE message upon receiving unsolicited packets for a downstream host.

Ending with "*": More than half of the paths end with one or more "*" characters, implying that no ICMP reply was received. This can occur when the TTL-limited probes are discarded (say, by a firewall), the components along this part of the path do not participate in ICMP (or apply rate-limiting), or the ICMP messages are lost along the reverse path.

Ending with "!": Around 12% of the traceroute results end with a "!" symbol indicating that the last component in the path was unable or unwilling to forward the packets toward the destination. The two most common scenarios are !H (host unreachable) and !X (communication administratively prohibited), with !N (network unreachable) a distant third.

Ending after 30 hops with an IP address: About 7% of the paths continue to the maximum length (30 hops) and end with an IP address. The vast majority (95%) of these paths have forwarding loops, where some addresses appear multiple times in the path. A small fraction of the paths do not contain loops and appear to represent paths that continue beyond 30 hops.

Although most loops persisted till the end of the traceroute path, a few paths had tempo-

	AS 6431	AS 25	AS 3130
Expected	11.21%	11.24%	11.21%
Unexpected	14.37%	14.17%	15.00%
"*"	54.79%	55.48%	53.92%
"!"	12.15%	12.09%	12.48%
30 hops	7.47%	7.02%	7.40%

Table 6.4: Ending of the traceroute experiments

rary loops and some loops that ended with a "*". In total, 7–8% of the paths contained a forwarding loop. This may stem, in part, from IP addresses that have not been allocated to any operational network or machine. Some routers may be configured with default routes that direct the traffic back to an upstream router. We do not expect traceroutes to known "live" addresses to uncover such a large percentage of forwarding loops. Overall, the combination of forwarding loops, unreachable hosts, discarded probe packets, and devices with ICMP disabled resulted in a relatively small number of probes that traversed the entire path to the destination. To enable comparisons with the BGP data, we preprocessed the end of each traceroute path to remove forwarding loops and trailing "*" and "!" characters; then we converted the (partial) forwarding path to an AS path by mapping each hop to an AS number, where possible. As such, we did not expect a complete match between the BGP and traceroute AS paths. Instead, we compared the two paths up to and including the end of preprocessed traceroute path. For example, if the traceroute AS path is "4006 16631" and the BGP AS path is "4006 16631 22476," we considered this a successful match.

6.3.3 Comparing BGP and Traceroute Paths

To map IP addresses to AS numbers, we first applied the *whois.ra.net* data that form the basis of the "NANOG traceroute" tool [7]; the *whois.arin.net*, *whois.ripe.net*, and *whois.apnic.net* data were not appropriate for our purposes since these services do not provide the AS number

	Whois Data			Combined BGP Tables			Resolving Incompletes		
	6431	**25**	**3130**	**6431**	**25**	**3130**	**6431**	**25**	**3130**
Match	44.7%	44.7%	46.1%	71.7%	73.20%	73.4%	77.8%	78.0%	81.6%
Mismatch	17.1%	29.4%	23.0%	6.1%	8.3%	7.2%	6.6%	9.0%	7.1%
Incomplete	38.2%	25.9%	30.9%	22.1%	18.5%	19.4%	15.6%	11.1%	11.3%
unmapped hop	33.4%	20.5%	25.9%	1.5%	2.7%	2.5%	0.3%	0.6%	0.3%
* hop	8.7%	7.2%	8.5%	9.1%	7.6%	8.7%	6.4%	4.6%	5.5%
MOAS hop	0.0%	0.0%	0.0%	13.0%	9.8%	9.7%	10.0%	6.9%	6.4%
Match/mismatch	2.62	1.52	2.00	11.70	8.79	10.20	11.74	8.96	11.43

Table 6.5: BGP vs. traceroute AS paths for different AS mapping techniques

associated with an IP address. Unfortunately, the *whois.ra.net* data are out-of-date and incomplete. The statistics in "Whois Data" columns in Table 6.5 show that the BGP and traceroute AS paths matched less than half of the time. Incorrect IP-to-AS mappings may be responsible for many of the "mismatches" with the BGP AS path. Many traceroute paths were "incomplete" because no mapping exists in the whois database for some of the router hops. Around 20–33% of the traceroute paths had "unmapped" IP addresses that *whois* could not associate with an AS; this is partially explained by ASes that have not updated *whois* to reflect their current address assignments.

To improve the IP-to-AS mapping, we combined BGP routing table data from many vantage points. Combining multiple routing tables provides (i) a richer view of different subnets that may be aggregated at other locations, (ii) a more complete picture of prefixes associated with multiple origin ASes, and (iii) a lower risk of missing certain prefixes due to transient reachability problems at any one router. Table 6.6 lists the number of prefixes in each BGP routing table, along with the number of prefixes with more than one origin AS. The RouteViews data [16] consisted of BGP routes learned from 23 participating ASes, mostly in the United States. The data from the RIPE-NCC Routing Information Service project [11] provided BGP routes from 75 ASes, mostly in Europe. The SingAREN routers [12] had BGP routes from ASes in the Asia-Pacific region. Each of the other tables provided BGP routes seen from one vantage point. All of the BGP tables were collected around May 29, 2003, in the middle of our traceroute experiments, to limit the effects of changes in the mapping of prefixes to origin ASes over time. Combining all of the tables produced a mapping with more than 200,000 prefixes and 16,000 ASes. About 10% of the prefixes mapped to multiple origin ASes.

Using the collection of BGP tables increased the "match" rate and substantially decreased the fraction of paths with "unmapped hops," as shown in the "Combined BGP Tables" columns in Table 6.5. This occurred because the BGP tables from the operational routers provide a more complete and up-to-date view of the "ownership" of the IP addresses appearing in the traceroute paths. Still, the BGP and traceroute AS paths agreed less than 73% of the time, even under our relatively liberal notion of "matching" (*i.e.,* after trimming the end of the traceroute paths). Less than 8.3% of the traceroute AS paths differ from the corresponding BGP AS path. In the remaining cases, the traceroute path was "incomplete" because one or more hops did not map directly to a single AS number:

Unmapped hop: In a few ($<$ 3.0%) of the paths, some hops had an address that did not match any prefix in the set of BGP tables. Private IP addresses accounted for less than 40% of the cases. Unmapped hops can arise when interfaces are assigned addresses that are not advertised to the larger Internet.

"*" hop: Many traceroute paths had one or more "*" characters, even after removing trailing "*" characters at the end of the path. A "*" hop may stem from a lost probe or ICMP packet, or from an intermediate node that does not participate in ICMP.

Multiple origin AS hop: Around 9–13% of the traceroute paths had at least one interface address that mapped to multiple AS numbers, making direct comparisons with the BGP path impossible. MOAS prefixes occur for various reasons including misconfiguration, multihoming, or exchange points [110].

	Extracted Prefixes	**Origin ASes**	**MOAS Prefixes**
AS 6431	120997	15105	0
AS 25	124202	15213	0
AS 3130	121054	15086	0
AS 73	123583	15194	0
AS 6521	121096	15099	0
AS 14177	121135	15104	0
AS 10781	121669	15103	0
AS 22208	125050	15136	0
RouteViews	134095	15294	860
RIPE(00–08)	128960	15328	3400
SingAREN	6744	862	25
Potaroo	142348	16112	211
Verio	105381	13778	116
AT&T	128411	15171	109
Combined	203698	16367	8827

Table 6.6: BGP tables for IP-to-AS mapping around May 2003

The three cases are not mutually exclusive; a single traceroute path may have hops with one or more of these properties.

6.4 Resolving Incomplete Paths

This section describes and evaluates three simple techniques for analyzing a large fraction of the "incomplete" traceroute AS paths, as summarized in the "Resolving Incompletes" columns in Table 6.5. We discuss how to use internal router configuration files to validate the results, using data from a large service provider (AT&T, AS 7018) as an example. The configuration data enables us to verify whether certain interfaces belong to a particular AS and what lies on the other side of a link. We also can identify static routes that are used to direct traffic to specific customers. Our validation scripts could be used to compute statistics for other networks, without requiring these ASes to divulge their raw configuration files.

6.4.1 Unresolved Hops Within an AS

Many of the incomplete paths have "*" or unmapped hops in between two hops that map to the same AS; for example, a path may have one or more "*" hops between two interfaces that both map to AS 1239. We assume that such "*" and unmapped hops belong to the same AS as the surrounding hops; that is, we convert a path with hops "1239 * 1239" to a single AS-level hop of 1239. This is similar to the approach in [102] of clustering routers in a graph based on the AS number and associating each "*" interface with the nearest cluster. This simple heuristic reduced the number of incomplete paths with "*" hops by 30–40%. For unmapped hops, it reduced the incomplete paths by about 40%.

To test our hypothesis, we investigated the traceroute paths that appear to have one or more "*" hops within AS 7018 (*i.e.,* path segments such as "7018 * 7018" or "7018 * * 7018"). We inspect the IP address of the last hop in the path segment—the first hop after the "*" hops. We assume that this IP address corresponds to one end of the link from the previous router; the other end of the link should have the same network address. For example, a point-to-point link with the prefix 192.0.2.156/30 would have two interfaces with addresses of 192.0.2.157 and 192.0.2.158; 192.0.2.156 and 192.0.2.159 would correspond to the network and broadcast addresses, respectively. Upon seeing a hop with IP address 192.0.2.157, we look for another interface on a different router with IP address 192.0.2.158. In 98.1% of the cases, we are able to identify the router associated with this interface and verify that this router belongs to AS 7018. The remaining 1.9% of cases may have stemmed from transient routing changes where the hops in the traceroute path did not represent a single consistent path through the network.

6.4.2 Unmapped Hops Between ASes

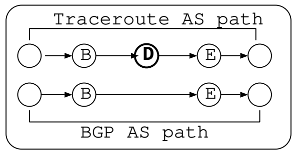
Most of the unmapped hops appeared between interfaces that are mapped to different ASes (*e.g.,* "1239 ? ? 64"). We attempted to associate the unmapped hop(s) with the previous or subsequent AS, using DNS and *whois* data. First, we considered the suffix of the domain names associated with the interfaces (*e.g.,* converting "sl-gw9-ana-4-0-0.sprintlink.net" to "sprintlink.net"), including the country domain if present; reverse DNS lookups were successful for 59% of the IP addresses in the traceroute results. If an unmapped hop had the same DNS suffix as a neighboring (mapped) interface, we associated the unmapped hop with that AS. This is similar to the approach in [45] of using DNS names to identify routers belonging to the same service provider. However, DNS did not always return a name for the unmapped hop; if some other interface in the same /24 address block had a successful reverse-DNS lookup, we used the DNS suffix for that interface. Second, we used *whois* to identify the AS responsible for the unmapped interface; we used this AS mapping only when it matched one of the adjacent ASes in the traceroute path. These techniques reduced the number of paths with unmapped hops by over 50%; these "resolved" traceroute AS paths had about the same proportion of "matches" with the BGP AS paths as the initial "complete" traceroute paths did, increasing our confidence in these additions to the IP-to-AS mapping.

For this heuristic, validating with configuration data involved checking that each "?" hop mapped to AS 7018 actually corresponded to the IP address assigned to an interface in that network. However, the AS 7018 network numbers its interfaces out of an address block that is advertised to the rest of the Internet; as such, these interfaces did not appear as unmapped hops in the traceroute paths, and we could not use the configuration data to test the heuristic. Configuration data from other ASes would have been more useful. Overall, though, the fraction of "?" hops resolved by this set of heuristics was relatively low because we did not try to map "?" hops to other AS numbers (*e.g.,* besides adjacent AS hops like 1239 or 64). We experimented with other heuristics but did not believe that the DNS and *whois* IP-to-AS mappings were accurate enough to warrant a more liberal approach.

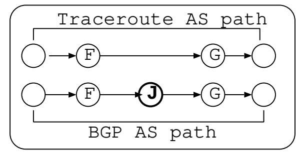
6.4.3 MOAS Hops at the End of the Path

Interface IP addresses that map to multiple origin ASes appeared in 10–13% of the traceroute AS paths. About 3% of all traceroute AS paths *end* with hops that map to multiple ASes. This can occur due to edge networks that connect to multiple providers without using BGP (or using private AS numbers) or due to misconfigurations [110]. We envision that an AS traceroute tool should report that these hops map to multiple ASes for diagnostic purposes. For the rest of the chapter, we include these traceroute paths in our comparison with the corresponding BGP AS paths. We consider these traceroute hops a "match" with the corresponding BGP hop if the AS in the BGP path matches any *one* of the ASes associated with the traceroute MOAS hops. These "resolved" traceroute AS paths had about the same proportion of "matches" with the BGP AS paths as the initial "complete" traceroute paths.

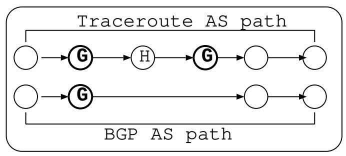
Using the configuration data, we investigated the traceroute AS paths where the last hop was mapped to AS 7018 and at least one other AS. In particular, we inspected the IP prefixes used to map these hops to multiple origin ASes to see if they actually corresponded to customers of AS 7018. In all of the cases involving AS 7018, the prefix was specified in a static route associated with one or more access links to a customer. That is, AS 7018 originated the route to this prefix on behalf of a customer and, as such, the prefix referred not to equipment inside the backbone but rather to addresses in the customer's network.



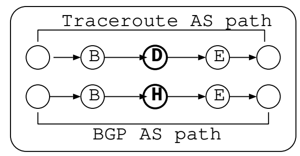
(a) Extra AS hop



(b) Missing AS hop



(c) Two-hop AS loop



(d) Substitute AS hop

Figure 6.2: Mismatch patterns for the traceroute AS paths

6.5 Improved IP-to-AS Mapping

After applying the techniques in Section 6.4, about 6–9% of the traceroute AS paths do not match the corresponding BGP AS path and another 6–10% have hops that map to multiple origin ASes. We suspect that inaccuracies in the IP-to-AS mapping are responsible for many of these cases. After a brief discussion of the causes of mismatches, we propose and evaluate algorithms for detecting IXPs, sibling ASes, and networks that do not announce routes for their infrastructure. The coverage of some of our techniques is limited by the fact that our measurement data come from only eight vantage points mostly in the United States, all directly connected to large providers in North America. The techniques discussed here and in the previous section are very efficient – linear with respect to the number of paths. The algorithms require on the order of a few minutes to run on traceroute paths to about 200,000 addresses. In Section 6.8 we present a more systematic approach of improving IP-to-AS mappings by iteratively applying a dynamic programming algorithm, complementing the set of heuristics described in this Section.

	AS 6431	AS 25	AS 3130
Extra intermediate hop	33%	40%	41%
Missing intermediate hop	22%	20%	20%
Two-hop AS loop	9%	7%	8%
Substitute AS hop	3%	3%	2%
Other	33%	30%	29%

Table 6.7: Statistics on mismatched traceroute paths

6.5.1 Patterns and Causes of Mismatched Paths

At least two-thirds of the differences between the BGP and traceroute AS paths fell into one of four simple patterns:

Extra AS hop: For about 30–40% of the mismatches, the traceroute AS path had one extra intermediate hop that does not appear in the corresponding BGP AS path, as shown in Figure 6.2(a).

Missing AS hop: About 20% of the mismatches came from traceroute AS paths that were missing one intermediate hop compared to the BGP AS path, as shown in Figure 6.2(b).

Two-hop AS loop: Around 10% of the traceroute AS paths had an AS-level loop with two AS hops, such as the "H G" segment in Figure 6.2(c).

Substitute AS: In 2–3% of the cases, the two paths had a different AS for one intermediate hop, such as AS D for the traceroute path and AS H for the BGP path in Figure 6.2(d).

Table 6.7 summarizes the statistics, focusing on the first mismatch between each pair of AS paths. In each case, the "mismatch" between the two AS paths was nested within the path, starting with an initial matching hop.

Our heuristics look for common occurrences of these "differences" across many AS paths to identify possible mistakes in the IP-to-AS mapping applied to the traceroute AS paths. Finding multiple instances of each pattern increases the confidence in our explanation for why the paths

	Extra	Miss	Loop	Subst	Other
Exchange point	X				
Sibling ASes	X	X	X	X	
Unannounced IP	X	X	X	X	
Aggregation/filtering					X
Inter-AS interface		X			X
ICMP source address	X	X		X	X
Routing anomaly	X	X	X	X	X

Table 6.8: Patterns and possible causes of mismatched AS paths

differ and also makes our algorithms more robust to transient routing changes that may affect the accuracy of some of the traceroute paths. In practice, some traceroute paths may be affected by the results of multiple techniques, since we apply the improved IP-to-AS mapping across all of the traceroute paths. Our algorithms are based on the patterns we expect from common operational practices. Table 6.8 summarizes the seven root causes we consider, and the kinds of mismatch patterns they can create. The first three cases introduce mistakes in the IP-to-AS mapping and are the focus of this section. The remaining four cases are "legitimate" mismatches that do not necessarily stem from an incorrect mapping; we defer discussion of these cases to the next section. In practice, most of the items in Table 6.8 do not fall naturally into a single "mismatch pattern"; therefore, our algorithms need to look carefully across multiple instances of mismatch paths to draw meaningful conclusions.

6.5.2 Internet Exchange Points (IXPs)

IXPs are junction points where multiple service providers meet to exchange BGP routes and data traffic. An IXP typically consists of a shared infrastructure, such as an ATM switch or a FDDI ring, with physical connections to routers in each of the participating ASes. An IXP may have its own AS number and originate routes to its infrastructure; alternatively, the address of the

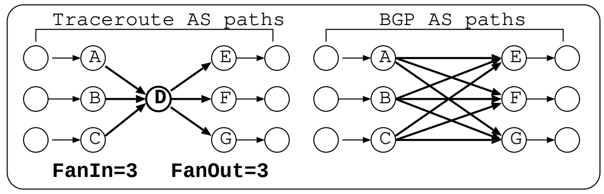


Figure 6.3: Traceroute vs. BGP AS paths through an IXP

shared infrastructure may be originated into BGP by one or more of the participating ASes. In either case, different pairs of service providers establish dedicated BGP sessions over the shared physical infrastructure. At the IP level, the forwarding paths traverse the shared equipment as shown in the left side of Figure 6.3. Yet, at the BGP session level, the participating service providers connect directly to each other, as shown in the right side of Figure 6.3. As a result, the AS-level forwarding path appears to have an extra AS hop relative to the corresponding BGP AS path, as shown earlier in Figure 6.2(a).

The patterns in Figure 6.3 for the AS-level forwarding and signaling paths drive our algorithm for detecting IXPs. First, we inspect cases where the traceroute AS path has an extra hop compared to the corresponding BGP AS path; the extra hop could be a single AS D or an individual prefix that maps to multiple origin ASes. In practice, we do not expect to see the AS for an IXP to appear in any BGP AS paths, except as the origin AS for the paths for the shared equipment at the site. As such, the second step of our algorithm removes from consideration any AS D that appears as a transit AS in any BGP AS path. Finally, we expect an IXP to provide service to several pairs of ASes. As such, we check the number of unique ASes appearing just before and just after D; the example in Figure 6.3 has a fan-in and fan-out of 3. For robustness, we apply a threshold for the minimum fan-in and fan-out; in this work, we apply a relatively small threshold of 2 since we only have measurement data from eight vantage points. Ideally, a larger threshold might be preferable for avoiding "false positives."

We also apply an additional requirement that for AS pairs consisting of the AS preceding and following the suspected IXP AS, there must at least two pairs with no AS in common. In other words, AS D is not considered as an IXP AS if it only appears as an extra AS in traceroute AS paths such as XDB and BDY, where X and Y are arbitrary ASes. As described in Section 6.5.3, AS B and D are likely to be siblings. This requirement is to assure the path diversity of selected IXPs and prevent mistaking a sibling AS for an IXP AS.

Applied to our measurement data, this algorithm found 477 cases (of an AS or a prefix) with a fan-in and fan-out of 1 or more with corresponding AS appearing in traceroute AS paths but not BGP paths. Only 25 cases had fan-in and fan-out of at least 2 and satisfy our criteria of an IXP; these cases are listed in Table 6.9 in decreasing order of fan-in and fan-out. To verify our results, we first queried *whois* using the AS number or prefix to see if the description contained the words "exchange point" or "Internet exchange"; for example, AS 5459 was listed as "London Internet Exchange" in *whois.ripe.net*. This check succeeded for 18 of our 25 inferences. Then, we compared our results against a list of known IXPs [8]. This confirmed 16 of the 25 inferences. Together, 19 of the 25 inferences passed at least one of these checks. Some of the remaining cases (highlighted in italics) may be IXPs, too; for example, CalRen is an exchange point for universities in California.

Inspecting the list of known IXPs, we find that we missed 13 known IXPs. Among them, all but one had a *fan-in* of 1; for example, the PAIX Seattle exchange point had a fan-in of 1 and a fan-out of 5. The 13 cases include 2 NAPs (in Seattle and Miami), 4 European IXPs, 1 Asian IXP, 2 Equinix sites, and 4 small IXPs in the exchange point block 198.32.0.0/16. We believe that our algorithm missed these cases due to the small number of measurement locations; in addition, our measurement sites connect directly to large tier-1 providers in the U.S. except for one site

	In	Out
California Research & Education Network (AS2151)	6	5
London IXP (AS5459)	4	7
Japan IXP (AS7527)	3	7
SANDY Network (AS5471)	2	2
PAIX (198.32.176.0/24)	9	50
Amsterdam IXP (193.148.15.0/24)	7	9
Seattle IXP (198.32.180.0/24)	6	32
Chicago Ameritech (206.220.243.0/24)	4	37
Equinix IBX San Jose (206.223.116.0/24)	4	20
Japan IXP (JPIX) (210.171.224.0/24)	4	9
London IXP (LINX) (195.66.224.0/19)	4	7
Hong Kong IXP (HKIX) (202.40.161.0/24)	4	6
Equinix Ashburn (206.223.115.0/24)	3	7
Tokyo Network Service Provider IXP (202.249.2.0/24)	3	5
Western Australia (WAIX) (198.32.212.0/24)	3	2
Hutchison Telecommunications, HK (210.0.251.0/24)	3	2
MAE West ATM San Jose (198.32.200.0/24)	2	13
Equinix IBX Secaucus (206.223.117.0/25)	2	4
MAE East (198.32.187.0/24)	2	3
Japan Network Information Center (202.249.0.0/17)	2	3
SI-TELEKOM-193-77, Slovenia (193.77.0.0/16)	2	3
Mae-West Moffet Field (198.32.136.0/24)	2	2
Lipex Ltd, Telehouse Network, UK (193.109.219.0/24)	2	2
Comite Gestor da Internet no Brasil (200.187.128.0/19)	2	2
ROSTELECOM-NET, Russia (213.24.0.0/16)	2	2

Table 6.9: AS numbers and prefixes inferred as IXPs

connecting to a large provider in Canada, limiting the number of ways the traceroute paths could reach the IXPs. In the end, some of these remaining IXPs are potentially mistakenly placed in other categories by the techniques described later in this section.

Using the list of IXPs generated by our algorithm, an AS-level traceroute tool could indicate which IP-level hops map to exchange points. We used our results to map these IP addresses to null ASes; that is, we remove the IXP ASes and prefixes from the traceroute AS paths. For example, a traceroute AS path with "B D E" would become "B E" after removing AS D. The results of

	Exchange Points			Sibling ASes			Unann Address		
	6431	**25**	**3130**	**6431**	**25**	**3130**	**6431**	**25**	**3130**
Match	78.2%	84.4%	85.4%	86.0%	85.9%	87.0%	90.0%	90.6%	91.0%
Mismatch	6.4%	8.7%	7.1%	6.4%	7.8%	6.2%	2.7%	3.5%	2.6%
Incomplete	15.4%	6.9%	7.5%	7.6%	6.3%	6.8%	7.4%	6.0%	6.6%
Match/Mismatch	12.20	9.70	12.06	13.42	11.00	14.08	33.51	25.95	35.41

Table 6.10: The results of using the three techniques to tune the IP-to-AS mapping

applying the new IP-to-AS mapping across all of the traceroute paths is shown in the "Internet Exchange Points" columns in Table 6.10. Compared with the earlier results in Table 6.5, the number of matched paths increased to 78.2-85.4%, corresponding to an increase of 1–4 percentage points. This occurs due to a decrease in both the number of mismatched paths and the number of incomplete paths. For the AS 6431 data, the IXP algorithm resolved more than half of the incomplete paths with MOAS hops. We would expect more dramatic results for sites that connect to smaller providers that tend to route more of their traffic through IXPs rather than private peering links.

6.5.3 Sibling ASes

In some cases, a single organization owns and manages multiple ASes, sometimes as a result of mergers and acquisitions. The ASes may share address space, with one AS numbering some of its equipment using part of an address block originated by another. This affects the mapping of traceroute hops to AS numbers, and can lead to ambiguity about which AS actually carries the traffic; in some sense, the distinction between the two ASes may not be important since they "belong together." In the example at the top of Figure 6.4, the traceroute AS paths includes ASes B and D though the BGP AS path includes only one of the two ASes, as shown in the bottom of the figure. This phenomenon can result in traceroute AS paths that have an extra AS hop (B or D) relative to the corresponding BGP paths. Sibling ASes can also produce traceroute paths with other patterns,

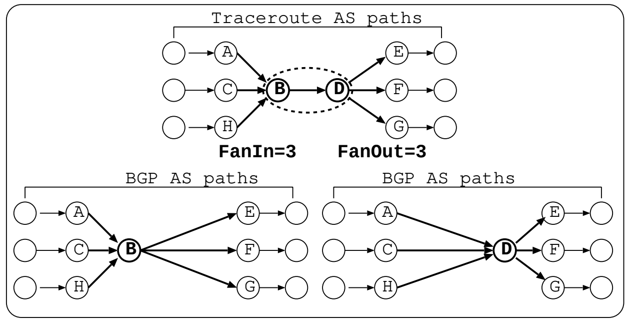


Figure 6.4: Traceroute and BGP AS paths with siblings

as discussed in the next subsection.

The patterns in Figure 6.4 suggest a way to identify cases where sibling ASes affect the traceroute AS path. Similar to the IXP algorithm, we consider the fan-in and fan-out of traceroute AS paths traversing a two-hop segment "B D" that corresponds to a single AS hop in the corresponding BGP paths. For robustness, we apply a threshold to the fan-in and fan-out; in this work, we enforce a minimum fan-in and fan-out of two. In addition, we focus on cases where one of the two ASes (say, AS D) never appears in a BGP AS path, except as an origin AS. That is, we assume that one AS (B) is using the address space originated by the other AS (D), rather than trying to capture cases where each AS borrows from the other.

In applying this algorithm to our data, we identified 28 pairs of sibling ASes. The fan-in and fan-out were as large as 10 and 31, respectively. To check our results, we inspected the *whois* entries for the ASes and found that in 15 cases the two ASes had the same organization name (*e.g.*, ASes 1239 and 1791 belonged to Sprint and ASes 1299 and 8233 belonged to TeliaNet). In the remaining seven cases, the AS pairs appeared together as originating ASes for one or more prefixes in the BGP routing tables, adding extra credibility to the conclusion that they are siblings. As

	AS 6431		AS 25		AS 3130	
Number of vantage points	**3**	**8**	**3**	**8**	**3**	**8**
Match	88.5%	90.0%	89.2%	90.6%	88.5%	91.0%
Mismatch	4.0%	2.7%	4.7%	3.5%	3.8%	2.6%
Incomplete	7.5%	7.4%	6.1%	6.0%	6.7%	6.6%
Match/Mismatch ratio	22.11	33.51	18.89	25.95	22.99	35.41

Table 6.11: The effect of multiple vantage points: comparing using the first three with all eight probing locations.

part our future work, we plan to compare our sibling inferences with the results of algorithms for inferring AS relationships from BGP AS paths [42, 101].

We modified the IP-to-AS mapping based on these results to treat sibling ASes as a single network. That is, we replaced every occurrence of B or D in the IP-to-AS mapping with the set $\{B, D\}$. We considered the traceroute and BGP AS hops a "match" if the BGP AS hop was the same as either of the two siblings in the traceroute AS path. After applying the new IP-to-AS mapping to all of the traceroute paths, 85.9-87.0% of the traceroute AS paths matched the corresponding BGP AS paths. This increase came from up to a 12% reduction in the mismatched paths and up to a 50% reduction in the incomplete paths. As a result, the mismatched and incomplete paths became as low as 6.2% and 6.3% of the total number of paths, respectively, as shown in the "Sibling ASes" columns of Table 6.10.

6.5.4 Unannounced Infrastructure Addresses

An AS does not necessarily announce the addresses assigned to its equipment via BGP. This can lead to "unmapped" addresses, as discussed earlier in Section 6.3.3. However, sometimes these addresses fall into larger address blocks originated by the AS's sibling or provider. This can cause several patterns of mismatches between the BGP and traceroute AS paths. In the example in Figure 6.5, AS C connects to two upstream providers A and B. AS A has allocated a subnet of its

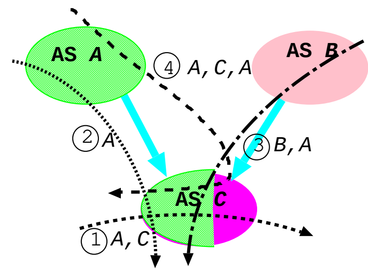


Figure 6.5: Mismatches caused by unannounced IP addresses

address space to AS C and originates the supernet in BGP to the rest of the Internet. AS C uses its part of the address block to number some of its equipment but C does not advertise the subnet in BGP. As a result, some traceroute hops in AS C are mistakenly mapped to AS A. Figure 6.5 shows four example paths:

Extra hop: Path 1 traverses some hops in AS C that (mistakenly) map to A and others that (correctly) map to C, resulting in a traceroute path of "A C" rather than "C".

Missing hop: Path 2 traverses both A and C, resulting in a BGP path of "A C." However, the hops in C are (mistakenly) mapped to A, resulting in a traceroute path of "A".

Substitute hop: Path 3 traverses both B and C, resulting in a BGP path of "B C." However, the hops in C are (mistakenly) mapped to A, resulting in a traceroute path of "B A."

AS loop: Path 4 traverses ASes A and C, resulting in a BGP path of "A C." However, some of the hops in C are (mistakenly) mapped to A, resulting in a traceroute path of "A C A."

Focusing first on AS loops, our algorithm looks for the loop patterns in Figure 6.6(a). We count the number of times ASes G and H appear together in this pattern, where the traceroute AS path has a loop and the corresponding BGP path has a single hop for each AS. In analyzing our data, we found that small number of AS pairs appeared in many such paths, and these accounted for the vast majority of the loops. Our algorithm applies a threshold of 50 occurrences before inferring that ASes G and H "share" address space and changes the mapping of the second G hop to an H; that is, once a traceroute AS path appears to "enter" an AS H, we assume that the path continues in this AS. In effect, we assume that H "owns" the addresses of these traceroute hops but did not advertise them in BGP. However, we do not know the size of the address block allocated to H. We inspect the IP addresses of the individual traceroute hops involved and add the corresponding /24 prefix to our IP-to-AS mapping (with H as the associated AS). In applying this method, we found 20 unique AS pairs responsible for 830 unannounced /24 prefixes; many of these prefixes were adjacent, suggesting that some larger subnets were involved. Furthermore, the matched prefixes of the corresponding IP addresses tend to have shorter length, indicating that there may be smaller subnets missing in our prefix to AS mapping.

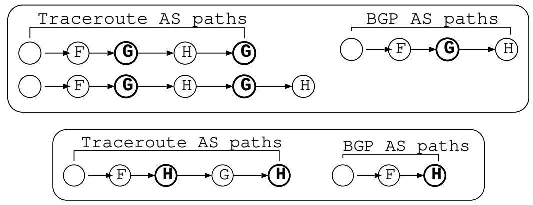
To check our results, we inspected the *whois* entries for these ASes and confirmed that in half of the 20 cases the two ASes belonged to the same institution (*i.e.,* the two ASes are siblings). In two other cases, the AS pairs could be classified as siblings based on their Web sites—AS 174 (PSINet) and AS 16631 (Cogent Communications), and AS 209 (Qwest) and AS 3908 (Supernet). These two examples are cases where the *whois* data do not capture acquisitions or mergers. Six more cases appeared to have a provider-customer relationship, in that *whois* showed one AS (the "customer") responsible for a subnet of an address block assigned to the other AS (the "provider"). In these cases, *whois* had address assignment information that was not available from the BGP routing tables since the "customer" subnet was not visible in any of our datasets. We were unable to verify the remaining two AS pairs.

For extra and substitute ASes, we follow a similar approach to the algorithms for IXPs and siblings. Focusing on patterns like Figure 6.3 and Figure 6.6(c), we apply a threshold of fan-in and fan-out of two to infer that an AS pair "shares" address space. Unlike the IXP and sibling algorithms, we apply these checks at the *prefix* level, assuming that some /24 prefix that has not been announced. For the "extra hop" case, we identified 308 such /24 prefixes; for the "substitute hop" case, we identified 25 prefixes. The case of a "missing hop," shown in Figure 6.6(b), is more complicated. By applying the fan-in and fan-out thresholds, we identified 77 AS pairs that appeared to "share" address space. However, we do not have a reliable way to determine which parts of the address block should be associated with the "missing" AS. Therefore, we do not use these results to modify our IP-to-AS mapping in any way. In ongoing work we are exploring ways to handle "missing" hops.

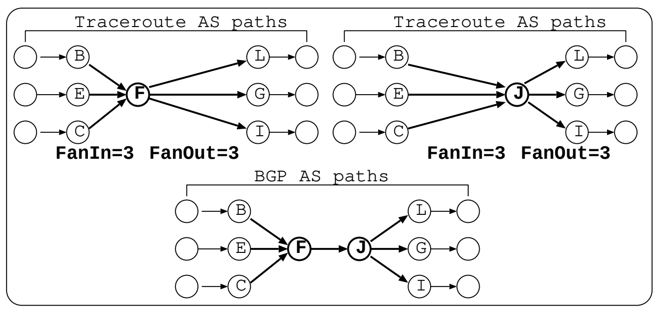
After identifying the unannounced addresses and the owning AS, we modify the IP-to-AS mapping to add a new entry for each /24 prefix. Applying the new IP-to-AS mapping across all of the traceroute paths reduced the number of mismatched paths by as much as a factor of two. In addition, the new mapping slightly reduced the fraction of incomplete paths. Ultimately, after applying all three of the techniques in this Section, the "match" rate exceeded 90% for each data set and the ratio of matches to mismatches ranged from 25-35. Still, a small fraction (2.6–3.5%) of the traceroute AS paths did not agree with the BGP AS paths; Section 6.6 explores possible explanations for the remaining mismatches.

6.5.5 Diversity of Probing Locations

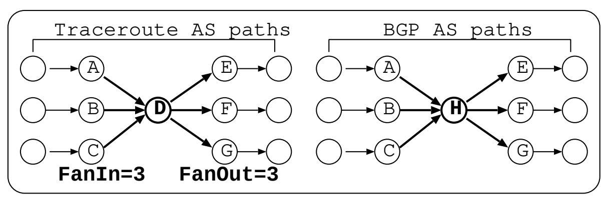
Our techniques rely on the topology diversity of the traceroute measurements. Increasing probing locations increases the likelihood that a different AS-level path is used to traverse pairs of siblings, Internet eXchange Points, and unannounced address spaces. This, in turn, reduces the probability that they would be missed in an AS-level traceroute tool based on our techniques. Both the geographic location and the upstream connectivity have an impact on the diversity of AS-level paths. Previous work [22] studied the marginal utility of discovering network topology using



(a) AS loop



(b) Missing intermediate AS hop



(c) Substitute intermediate AS hop

Figure 6.6: ASes not announcing their infrastructure addresses

traceroute. They concluded that increasing the number of sources in traceroute experiments has low utility beyond the second source. Increasing the number of sources is admittedly more important for our purposes, though, since our heuristics rely on fan-in as well as fan-out counts.

In our study, we try to cover all the destination prefixes in the local BGP table. For each source, the set of destination probed is roughly the same. We found that adding additional sources in our study significantly increases the fan-in and fan-out counts across both sibling and IXP ASes.

We compare the inference results based on measurements from the first three vantage points with all eight locations. For example, the fan-in and fan-out count going through PAIX, the Palo Alto Internet eXchange Point, increased from 5 and 14 to 9 and 50 respectively. Four known IXPs (Equinix San Jose, London IXP, Mae-West San Jose, and Mae-East) were missed using the first three locations due to insufficient fan-in and fan-out count, but they are correctly inferred using all eight data sets. As several newly added locations are in California, exchange points in San Jose are therefore more likely to be inferred.

Table 6.11 compares the match between traceroute AS paths and BGP AS paths using data from the first three locations with the complete data from all eight locations. The improvement is due to newly discovered IXPs, siblings, and unannounced address blocks as result of increased path diversity. The increase in matched paths is only between 1.5 and 2.8%; however, the reduction in mismatched paths ranges between 25–30%. This eliminates the false positives for potential routing problems that network operators need to investigate further. The table also shows that the match to mismatch ratio of comparing local BGP table AS paths with traceroute AS paths increased by 35–50%. We believe that adding vantage points in Europe and Asia would offer further advantages.

6.6 Legitimate AS Path Mismatches

In this section, we discuss four "legitimate" reasons why the traceroute and BGP AS paths may disagree, and speculate on whether the cases might explain some of the remaining "mismatches." Where possible, we look for evidence of these cases in our routing data and in the configuration files for AS 7018. We also propose additional measurement that would help classify these mismatches more precisely.

	AS 6431	AS 25	AS 3130
Extended path	22%	18%	19%
Missing hop	24%	25%	27%
Extra hop	9%	12%	13%
Other	45%	45%	41%

Table 6.12: Remaining mismatches with BGP AS path

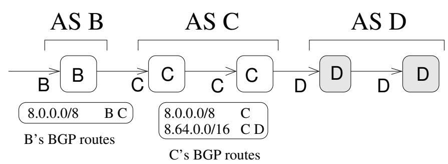


Figure 6.7: Extended traceroute path due to filtering by AS C

6.6.1 Route Aggregation /Filtering

At each of our eight measurement locations, the local BGP table does not have a complete view of the IP prefixes throughout the Internet. To limit protocol and storage overhead, routers may be configured to filter routes for certain subnets or combine multiple subnets together into a single aggregated route [33]. For example, Figure 6.7 shows an AS C that has the address block 8.0.0.0/8 and assigns the subnet 8.64.0.0/16 to its customer, AS D. Although AS C has BGP routes for both prefixes, only the route for 8.0.0.0/8 is propagated to AS B. Packets from AS B to the destination 8.64.0.1 would have a longest-matching prefix of 8.0.0.0/8 (with an AS path of "B C" in the local BGP routing table). However, the forwarding path would actually continue beyond AS C through one or more hops in AS D. Whether these traceroute hops are mapped correctly to AS D depends on whether the addresses of D's interfaces (which may or may not fall within the 8.64.0.0/16 block) are announced into BGP and are seen from the vantage points where we collect BGP routing tables.

Since many of our traceroute experiments do not traverse the entire forwarding path to the destination, we may significantly *undercount* the cases where route aggregation results in a

BGP AS path that "ends early" relative to the forwarding path for destinations in a smaller (unseen) subnet. Yet, across the three data sets, extended traceroute AS paths still account for 18–22% of the mismatches with the BGP AS paths, as shown in Table 6.12. To test our hypothesis that route aggregation is responsible for some of these cases, we compare the AS-level forwarding paths for the two IP addresses in each prefix (*e.g.,* 8.0.0.1 and 8.64.0.1). Across all of the prefixes where both forwarding paths are "complete," the two IP addresses have the same AS-level forwarding paths more than 99% of the time. However, when we focus on cases when either (or both) of these IP addresses has an "extended" path, this number drops below 75%; in more than 20% of the cases, one address has a forwarding AS path that matches the BGP AS path and the other has an extended path. The differences in the pairs AS-level forwarding paths are consistent with the effects of route aggregation/filtering.

6.6.2 Interface Numbering at AS Boundaries

Traceroute reports the IP addresses of *interfaces* rather than routers. In practice, interfaces to the same link are assigned addresses from the same prefix (*e.g.,* interfaces 192.0.2.157 and 192.0.2.158 forming a single point-to-point link with prefix 192.0.2.156/30). This introduces a potential problem for a link between two ASes—the interfaces are typically assigned an address block belonging to one of the two ASes, not both. In some cases, the path may enter and leave a router in some AS C where the two hops have addresses "owned" by the adjacent ASes, such as B and D, as shown in Figure 6.8. In this example, the traceroute AS path appears to have a segment "B D" when the path actually traverses a single router in AS C; in contrast, the "B C D" in the BGP AS path is correct. As such, interface numbering at AS boundaries can result in a traceroute AS path that has a "missing" AS hop when compared to the corresponding BGP AS path. About 25–27% of the remaining "mismatches" between the BGP and traceroute AS paths stem from a single "missing"

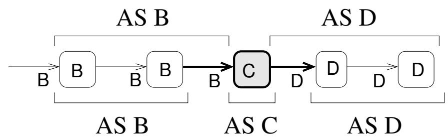


Figure 6.8: Missing AS hop C due to interface numbering

AS hop in the traceroute path, as shown in Table 6.12; we speculate that interface numbering at AS boundaries may be partially responsible.

To quantify these effects, we inspected cases where AS 7018 appeared as AS B, C, or D in a BGP path where the corresponding segment of the traceroute path "B D." AS 7018 never appeared as AS D and appeared only once as AS C; as such, we focused our attention on the case where AS 7018 corresponded to AS B. We first extracted the IP address of the last hop in the traceroute path that mapped to AS 7018; then, we generated the IP address of the other end of the link (*e.g.,* converting 192.0.2.158 to 192.0.2.157) and looked for an interface with this IP address in the configuration files from the same day. Then, we looked in the same configuration file to see if the interface was associated with a BGP session to a neighboring domain; if so, we extracted the remote AS number associated with this BGP session and compared it to the AS C in the BGP AS path. In more than 97% of the cases, we found that the last hop in AS 7018 was an interface associated with a BGP session to AS C rather than AS D or any other AS. In Section 6.12, we discuss how router-level graphs of the Internet [13, 45, 99] could help resolve these kinds of ambiguities.

6.6.3 Outgoing Interface in ICMP Message

Traceroute "discovers" a hop along the forwarding path from the source address of the ICMP TIME_EXCEEDED message sent in response to a TTL-limited probe. Ideally, the address corresponds to the *incoming* interface where the packet entered the router. However, the ICMP

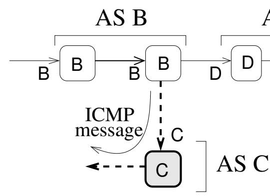


Figure 6.9: Extra AS hop C due to outgoing interface in ICMP

RFC [91] does not explicitly state which IP address the router should use. In practice, some routers may assign the source address based on the *outgoing* interface used to forward the ICMP message back to the host initiating the traceroute [19]. Since routing is not necessarily symmetric, the interface receiving the traceroute probe and the interface sending the ICMP message are not always the same. When this happens, traceroute reports the wrong forwarding path which can, at times, result in an incorrect AS-level path. Figure 6.9 shows an example where the actual forwarding path traverses ASes B and D, though traceroute reports an incorrect hop that maps to AS C. This can result in a traceroute AS path with “B C D” when the corresponding BGP path is simply “B D.” About 9–13% of the remaining “mismatches” have a single extra AS hop in the traceroute AS path; we speculate that ambiguity about the source IP address in the ICMP reply may be responsible for some of these cases.

The work in [19] checked the source code for several IP stacks and tested the behavior of a Cisco 7500 router; only the Linux IP stack used the address of the outgoing interface in the TIME_EXCEEDED message. We evaluated several other popular commercial routers and operating system versions in our test lab. Routers using the address of the incoming interface included the Cisco GSR (IOS 12.0(21)S3), Cisco 7200 (IOS 12.2.(10a)), Juniper M10 (JunOS 5.3R2.4), and Avici TSR (4.2.1A); however, the Cisco 3660 running IOS 12.0(7)XK1 used the IP address of the outgoing interface in its TIME_EXCEEDED replies. From our tests and the results in [19], we believe that the outgoing-interface problem might affect some of the traceroute paths, particularly for hops in smaller ASes that use lower-end routers. Determining whether this phenomenon explains some of our “extra intermediate hop” cases is difficult in practice. Ultimately, additional active measurements may be necessary to probe a suspicious router from multiple vantage points to infer its behavior.

6.6.4 Routing Anomalies

When the underlying route is changing, the “hops” returned by traceroute do not necessarily represent a single path through the network. This problem arises because each hop in the traceroute output corresponds to a separate TTL-limited probe that might not traverse the same forwarding path as the other probes sent toward the destination. In our preprocessing, we eliminated traceroute experiments where the corresponding BGP-level path was changing, so we may not see as many cases where routing changes occur. Still, the forwarding path may fluctuate even if the BGP path does not. Intradomain routing would tend not to alter the AS-level path but we cannot dismiss this possibility entirely. In addition, the AS-level forwarding path may change if some downstream AS experiences a BGP routing change for some subnet of the advertised prefix. To increase our confidence in the forwarding path, we could repeat the traceroute experiments in cases where the BGP and traceroute AS paths disagree to make sure that transient changes in the forwarding path are not to blame.

In addition, some routing anomalies can cause the forwarding and signaling paths to differ even when both are stable. This can arise due to “deflections,” where a router directs a packet to an intermediate node that has a different view of the “best” BGP route for a destination. The work in [53] describes how certain internal BGP (iBGP) configurations can be vulnerable to deflections; these scenarios would be extremely difficult for an operator to detect and debug. In many cases, a deflection would not change the AS-level path since the "best" AS path at different points in the network might exit via the same neighboring AS. Still, in some cases the two routers may pick different (equally good) best paths, such as AS B selecting a path through AS C (*e.g.,* "C E F") at one peering point and a path through AS D (*e.g.,* "D G F") at another. In such situations, deflections may cause the packets to traverse one of these paths despite the router having a BGP table with the other route. These kinds of anomalies could produce a variety of patterns in how the BGP and traceroute AS path differ.

6.7 Evaluating IP-to-AS Mappings

In this section we discuss our approach to evaluating a given mapping of IP prefixes to AS's by measuring how consistent it is with our given set of traceroute/BGP path pairs.

Based on the discussion in the previous section, a *mapping* A can be described by a set P_A of IP prefixes and, for each $x \in P_A$, a non-empty set $A(x)$ of AS's. Most such sets will be of size 1, although larger sets are possible if x corresponds to a MOAS. The set P_A is assumed to be complete in that every IP address has at least one prefix in P_A. For any IP address I, let $P_A(I)$ be the longest prefix in P_A that matches I. To simplify our notation in what follows, we extend the definition of A to IP's by letting $A(I)$ stand for $A(P_A(I))$.

Consider a traceroute/BGP path pair (p, q), where $p = (p_1, p_2, \ldots, p_n)$ is a sequence of IP's and $q = (q_1, q_2, \ldots, q_m)$ is a sequence of AS's. A function $a : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, m\}$ is a *matching* for p and q if $a(i) \leq a(i + 1)$, $1 \leq i < n$. (Implicitly, we also view a as matching p_i with $q_{a(i)}$.) Given a mapping A, the *error* of a given matching a for p and q is

$$E_A(a, p, q) = \left|\left\{i \leq n : q_{a(i)} \notin A(p_i)\right\}\right|$$

$$+\ a(n)\ -\ |\{a(i) : 1 \leq i \leq n\}|$$

Note that there is a penalty of 1 both for matching an IP p_i to an AS not in $A(p_i)$ and for failing to match any IP to an AS q_j that precedes the last matched AS in q. We do not impose a penalty for failing to match AS's after the last matched AS since traceroute paths can often be truncated due to timeouts, etc.

The *unavoidable error* $E_A(p, q)$ for mapping A and pair (p, q) is defined to be

$$\min \left\{ E_A(a, p, q) : a \text{ is a matching for } (p, q) \right\}.$$

A mapping A is said to be *consistent* with a pair (p, q) if $E_A(p, q) = 0$. This suggests two metrics for evaluating the quality of a mapping A:

1. $|\{(p, q) : E_A(p, q) = 0\}|$, the total number of pairs with which A is consistent, and

2. $\sum_{(p,q)} E_A(p, q)$, the total number of unavoidable errors for A.

In this work we shall consider both metrics. Their evaluation is facilitated by the fact that we can compute $E_A(p, q)$ efficiently using dynamic programming. In particular, given A, p, and q, we can in $O(nm^2)$ time find a matching a such that $E_A(a, p, q) = E_A(p, q)$. The details of the algorithm are provided in Appendix 6.11. When there is more than one optimal matching a, we break ties in favor of the one that matches IP's to the last possible AS in the path. This is motivated by the observation that mismatches that occur towards the end of the AS path due to aggregation should be resolved by mapping the IP address in question to the customer network, *i.e.,* to the AS later in the path.

The $O(nm^2)$ running time bound is quite small in our context, given that the average values of m and n are 3.8 and 7.4 respectively, with the maximum values encountered being 11 and 25. The algorithm takes only 0.2 milliseconds to complete for each traceroute-BGP pair on a 900 Mhz Sparc processor. However, to consider the entire data set – 1,860,448 pairs, it takes more than

6 hours. This provides a strong motivation for reducing the data needed to obtain accurate results, as discussed later in Section 6.9.

As an illustration of the effectiveness of the dynamic programming approach, consider what happens when we apply it to the initial mapping obtained from the BGP tables (in Table 6.6) with the entire data set of about 1.8 million pairs. Whereas the heuristics in [73] was only able to identify roughly 79% of these pairs as consistent, we find that 85.3% of them are. (A major factor in this improvement was our success in handling pairs with truncated traceroute paths, which the heuristics of [73] could not deal with at all.) When we apply our approach using the heuristically optimized mapping based on the heuristics presented in [73] to the full data set, we find that 90.7% of the pairs are consistent. This figure is much closer to the figure reported in [73], roughly 91%, but that figure ignored the results for several classes of defective pairs due to problems in traceroute, while we include them because our dynamic programming algorithm allows us to deal with the defects, even though most continue to have unavoidable errors. For the first mapping, the total number of unavoidable errors was 410,357 while for the second it was 265,459, which in both cases was roughly 1.5 errors per inconsistent pair.

6.8 Improving IP-to-AS Mappings

Our dynamic programming algorithm, by producing an optimal matching for each pair, can also be used to *improve* the mapping by helping us identify places where the mapping is accurate. For example, the mapping from the BGP tables assigns prefix 154.54.10.0/24 to AS2149, but in the 7972 pairs in which an IP appears for which this is the longest prefix in P_A, our optimal matchings never match it to AS2149, but match it to AS174 91.8% of the time (even though there is a penalty involved). This suggests that the actual mapping should have been to the latter AS.

Indeed, if we change the mapping accordingly, the optimal matchings for the relevant pairs now match 154.54.10.0/24 with AS174 100% of the time!

In this section, we consider a simple scheme for improving mappings by systematically exploiting the information from our optimal matchings. Other schemes are possible, but the current results already show significant improvement over the previous approach and provide a starting point for other more sophisticated rules. Suppose we are given a mapping A, the results of computing optimal matchings under A for all pairs (p, q), and a subset of the prefixes designated as *confirmed MOAS's* (typically confirmed by external data). We "improve" A as follows.

Say a pair (p, q) is *good* if $E_A(p, q) \leq 2$. We restrict attention to those prefixes x that fail to match with a member of $AS(x)$ in at least one good pair. (There is no need to change $AS(x)$ for the other prefixes.) For each prefix $x \in P_A$, let S_x denote the set of all IP addresses whose longest prefix in P_A is x, and let n_x denote the number of good pairs involving members of S_x. For each AS y, let $n_x(y)$ be the number of good pairs in which our dynamic programming algorithm matches a member of S_x with y. Let z_0 be the AS currently assigned to x that has the smallest value of $n_x(y)$ (ties broken arbitrarily), and let $r_0 = n_x(z_0)/n_x$. Similarly, let z_1 be the AS not assigned to x that has the largest value of $n_x(y)$, and let $r_1 = n_x(z_1)/n_x$. Apply the first of the following four rules that is relevant (and only that rule) to x:

- **Rule 1: Delete from MOAS-pair**

  If $|A(x)| = 2$, x is not a confirmed MOAS, and $r_0 < 0.1$, delete z_0 from $A(x)$, making it a singleton.

- **Rule 2: Replace a singleton**

  If $A(x)$ is a singleton and $r_1 \geq 0.55$, set $A(x) = z_1$.

- **Rule 3: Create a MOAS-pair**

If $A(x)$ is a singleton and $0.2 < r_1 < 0.55$, add z_1 to A(x), making it a pair.

- **Rule 4: Add to MOAS**

  If $|A(x)| \geq 2$ and $r_1 > 0.1$, add z_1 to $A(x)$.

All the rules that add items to $A(x)$ must of necessity reduce the number of unavoidable errors. The rule that deletes items can increase the number of errors, but is included to reduce the probability of creating fictitious MOAS's. Similarly, the threshold ratios for additions to $A(x)$ restrain the creation and expansion of MOAS's. (If for every x we let $A(x)$ be the set of all AS's, we would get a mapping with no unavoidable errors, but this would provide no insight into the true correspondence between prefixes and AS's.)

The above scheme can only increase the size of $A(x)$ by one. This is motivated in part by the conservative arguments of the previous paragraph, but also by the fact that we envision applying the scheme iteratively, with corrections made in one step helping to prevent mistakes in later ones. Here is the iterative procedure we use:

1. Let A be our initial mapping.

2. Repeat until done:

   (a) Compute optimal matchings for the current mapping A and all traceroute/BGP path pairs.

   (b) Apply the improvement scheme to A.

   (c) If A remains unchanged, we are done.

3. For each prefix x with $|A(x)| > 2$, add to $A(x)$ all those AS's that were matched to x more than 2% of the time and more than 10 times.

Given the rules in the scheme, it is unlikely that more than 10 iterations of the inner loop will be needed. This is because the last rule is the one that is most likely invoked over multiple iterations, and it cannot be invoked more 10 times given the threshold value, unless any deletion occurs. The final step is designed to expand the identified MOAS's of size greater than 2 to include most of the likely candidates for membership. This is still far more restrictive than the heuristics used in [73], which assumed that for every prefix x identified as a MOAS, $A(x)$ contained *all* the AS's.

In the next section we will discuss how the above procedure performs given various choices for the starting mapping and the set of traceroute/BGP path pairs. We will show that the algorithm performs very well and finishes within 10 iterations while making a very small number of changes to the initial IP-to-AS mapping.

6.9 Experiments

6.9.1 Experiment Selection and Design

In this section, we thoroughly evaluate the effectiveness and accuracy of the dynamic-programming-based approach on the data set in Table 6.1. The results of several experiments are discussed to show its robustness against the changes in the initial mapping as well as the reduction in the number of BGP and traceroute path pairs. The former analysis is important as it may be difficult to collect BGP tables from a large number of vantage points to construct a complete and accurate initial mapping. The latter evaluation is critical for a practical AS-level traceroute tool, as one cannot afford to do large amounts of probing.

Table 6.13 illustrates the experimental design space along the dimensions of (1) initial mapping and (2) set of traceroute-BGP path pairs. In the base case for the initial mapping, we

	BGP-traceroute pairs		
Initial mapping BGP Tables –	Full data set	Omitting Sources	Omitting Destinations
Unmodified	DP-BGP	DP-OS	DP-OD
Heuristically Optimized	DP-HO	-	-
Omitting Assignments	DP-OM	-	-

Table 6.13: Experiment design space and selected experiments

use the mapping obtained from the large collection of BGP tables shown in Table 6.6. The second option for the initial mapping is the "heuristically optimized" mapping obtained in [73] using ad-hoc heuristics. The third option is obtained by randomly deleting the assignments for 10% of the prefixes in the BGP-table-based mapping and instead initially assigning these to a non-existent AS, thus simulating the effect of added noise and inaccuracy in the initial mapping.

The base case for traceroute pairs contains all 1.8 million pairs from our original set. To study the robustness against smaller data sets, our second and third cases use smaller sets obtained by deleting all pairs with certain sources/destinations. More details on the construction of these subsets will be presented when we describe the relevant results.

The prefixes in the initial BGP tables have varying lengths. Among the 200K prefixes, more than 60% them have length of 24 or longer. For the purpose of applying our dynamic programming algorithm and improvement scheme, we subdivide all prefixes that are in the initial mapping and encountered in the traceroute data into /24's with one exception described below. The imposed limit at length 24 is motivated by the fact that most ISP's filter out prefixes longer than /24. This finer-grained partition of the IP-address space is motivated by the hope of allowing the mapping to account for the effects of address aggregation. A large supernet, for instance the one corresponding to a /16 prefix, might be assigned to a single AS by our initial mapping while it matches multiple AS's when we correlate traceroute paths with the BGP AS paths. If we divide the /16 into the corresponding /24's, the conflicting assignments may be significantly reduced without having to create new MOAS's – each /24 might match to a single AS in most of the pairs (although different /24's might match to different AS's).

The initial mapping for the newly created /24's is inherited from the mapping for the parent supernet. Any MOAS prefix in the original table is considered to be a "confirmed MOAS" in the sense of Section 6.8 since the origin AS's in the BGP tables are deemed to be reasonably accurate in this regard. However, a newly created /24 of such a MOAS prefix is not considered to be a "confirmed MOAS," thus allowing for potential deletions from the assigned set when we apply our improvement rules. For ease of programming, we carry out the subdivision into /24's for all encountered prefixes except those that already contain a subnet with length 24 or longer. Our results might improve if we carried out the subdivision for all prefixes, but very few prefixes and pairs are affected by this exception. The total number of prefixes in our experiment after subdivision into /24's is 105,068, more than half of which are newly created.

Recall that our improvement rules ignore traceroute-BGP path pairs with more than 2 unavoidable errors. Based on the initial BGP table mapping, 85.30% of the pairs have no errors, 10.67% have a single error, and 2.28% have 2 errors. The remaining pairs all have more than 3 unavoidable errors but make up only 1.75% of the pairs. After going through the dynamic programming based optimization, only 0.55% of the pairs have more than 2 errors. Most likely these represent defective pairs where the two paths fail to correspond due to forwarding path changes during traceroute probing, BGP routing changes, or other causes.

6.9.2 DP Evaluation: DP-BGP and DP-HO

The dynamic programming algorithm performs optimization on each pair of traceroute and BGP AS path. Subsequently, we re-optimize the assignment for each IP prefix by tabulating the results of the derived matchings for the prefix and applying the four rules of Section 6.8. We then iterate the procedure. Table 6.14 summarizes the results for all five experiments: DP-BGP, DP-HO, DP-OM, DP-OD, and DP-OS. It shows the benefit of iterative optimization. Each row is for one iteration of the improvement procedure. Iteration 0 evaluates the initial mapping. In general, iteration i evaluates the mapping created in iteration $i-1$. Recall that the final iteration simply expands all MOAS's with more than 2 elements to contain all AS's that match the prefix at least 2% of the time.

We now describe the meaning of each column in the table. The first column is the iteration number, 0 being the starting point. The "Mismatch" column indicates the percentage of traceroute-BGP path pairs with at least one unavoidable error. "Error count" is the total number of unavoidable errors in all the pairs. An error is either a mismatch between an AS in the BGP path and the assigned AS of a traceroute IP address using the dynamic programming algorithm or a skipped AS in the middle of a BGP path. Note, we do not penalize for AS's skipped toward the end of BGP path as traceroute may not reach the destination AS. Most pairs with unavoidable errors have 1 or 2 errors. The column labeled "Matched prefixes" presents the percentage of prefixes encountered in the data set that do not have conflicting matchings. The number of changed assignments in this iteration is shown in "New maps" column. The "Rule" columns contain the number of errors that are expected to be corrected by application of the corresponding rule during the current iteration. The values in parentheses indicate the number of prefixes to which the rule is applied during the current iteration. Only the first rule can increase the number of errors as it deletes an AS from an AS pair in the mapping. The total number of errors to be corrected or "Expected gain" is not shown due to lack of space. The last column, marked by "Actual gain," is the actual reduction in the total number of unavoidable errors compared to the previous iteration. This number is at least as large as the value in the "Expected gain" in the previous row. It is often significantly larger in the first

DP-BGP: using initial BGP-table-based mapping, 1,860,448 traceroute measurements									
i	**Mis-match**	**Error count**	**Matched prefixes**	**New maps**	**Rule 1 gain (map)**	**Rule 2 gain (map)**	**Rule 3 gain (map)**	**Rule 4 gain (map)**	**Actual gain**
0	14.70%	410357	90.82%	2853	-8 (6)	74071 (1850)	43371 (862)	28652 (135)	0
1	8.17%	212880	93.77%	442	-17 (3)	8908 (120)	3697 (221)	10271 (98)	197477
2	7.42%	187121	94.23%	47	0 (0)	16 (2)	1961 (16)	7010 (29)	25759
3	6.96%	177297	94.26%	10	0 (0)	6 (2)	4 (1)	451 (7)	9824
4	6.94%	176833	94.26%	2	0 (0)	0 (0)	4 (1)	2 (1)	464
5	6.94%	176827	94.26%	1	0 (0)	0 (0)	0 (0)	2 (1)	6
6	6.94%	176825	94.26%	27	0 (0)	0 (0)	0 (0)	32535 (27)	2
7	5.23%	143697	94.27%	-	-	-	-	-	33128
DP-HO: using Heuristically Optimized mapping, 1,860,448 traceroute measurements									
0	9.27%	265459	91.67%	2539	-122 (165)	27286 (1298)	17347 (592)	46891 (484)	0
1	4.96%	142129	94.19%	323	0 (5)	6691 (68)	1566 (149)	7112 (101)	123330
2	4.54%	125688	94.53%	29	0 (1)	12 (2)	119 (7)	3552 (19)	16441
3	4.36%	121945	94.55%	3	0 (0)	0 (0)	4 (1)	114 (2)	3743
4	4.35%	121827	94.55%	1	0 (0)	0 (0)	0 (0)	2 (1)	118
5	4.35%	121825	94.55%	1	0 (0)	0 (0)	0 (0)	2 (1)	2
6	4.35%	121823	94.55%	64	0 (0)	0 (0)	0 (0)	24428 (64)	2
7	3.08%	96786	94.59%	-	-	-	-	-	25037
DP-OM: omitting 10% of randomly selected BGP-table-based mapping, using 1,860,448 traceroute measurements									
0	54.11%	1700492	81.70%	9153	-4 (5)	728816 (8025)	83330 (1018)	17197 (105)	0
1	17.98%	476275	91.49%	1866	0 (9)	35649 (995)	45904 (483)	41894 (379)	1224217
2	12.01%	289402	93.30%	509	0 (1)	6582 (226)	4888 (74)	23019 (208)	186873
3	10.15%	247750	93.73%	97	0 (0)	131 (21)	237 (16)	4176 (60)	41652
4	9.88%	242295	93.79%	23	0 (0)	39 (6)	82 (3)	232 (14)	5455
5	9.86%	241919	93.81%	1	0 (0)	0 (0)	0 (0)	12 (1)	376
6	9.86%	241907	93.81%	159	0 (0)	0 (0)	0 (0)	60911 (159)	12
7	6.57%	175977	93.88%	-	-	-	-	-	65930

DP-OD: omitting traceroute probing destinations, using 242,836 traceroute measurements									
0	13.10%	46653	91.37%	1169	0 (0)	7448 (775)	4566 (337)	3406 (57)	0
1	7.73%	25728	93.97%	198	0 (2)	895 (49)	492 (83)	1786 (64)	20925
2	6.89%	22265	94.33%	42	0 (0)	1 (1)	102 (8)	1034 (33)	3463
3	6.46%	21081	94.39%	16	0 (0)	1 (1)	11 (3)	146 (12)	1184
4	6.41%	20917	94.42%	4	0 (0)	0 (0)	6 (2)	58 (2)	164
5	6.38%	20853	94.42%	1	0 (0)	0 (0)	0 (0)	1 (1)	64
6	6.38%	20852	94.42%	9	0 (0)	0 (0)	0 (0)	3986 (9)	1
7	4.79%	16749	94.42%	-	-	-	-	-	4103
F	7.12%	190511	91.93%	-	-	-	-	-	-
DP-OS: omitting traceroute probing sources, using 938,377 traceroute measurements									
0	18.45%	264925	91.56%	2721	-4 (4)	35063 (1753)	26137 (835)	19322 (129)	0
1	11.61%	149943	94.51%	436	-9 (3)	7683 (135)	8422 (207)	7935 (91)	114982
2	9.88%	124373	94.98%	56	-2 (3)	7 (2)	431 (19)	6340 (32)	25570
3	9.20%	116817	95.02%	13	0 (0)	14 (4)	476 (4)	85 (5)	7556
4	9.14%	116217	95.03%	1	0 (0)	4 (1)	0 (0)	0 (0)	600
5	9.14%	116213	95.04%	17	0 (0)	0 (0)	0 (0)	31783 (17)	4
6	5.94%	84194	95.04%	-	-	-	-	-	32019
F	6.34%	165630	94.09%	-	-	-	-	-	-

Table 6.14: DP-BGP, DP-HO, DP-OM, DP-OD, DP-OS: iterative optimization using dynamic programming.

several iterations, because the dynamic program re-optimizes the matchings at each iteration and the reassignment of one IP prefix may help us correct other errors as well.

We now focus on the results of the first experiment: DP-BGP. All 5 experiments conducted converge within 6 iterations. The last iteration is where the final step (4) of the iterative procedure adds likely AS's to some of the MOAS assignments. DP-BGP uses the initial IP-to-AS mapping based on 200K prefixes from BGP tables in Table 6.6. Initially, only 14.70% of the pairs have unavoidable errors, indicating that the BGP tables provide a fairly good starting point. Most modifications to the mapping are performed in the first iteration, as is the case in the other four experiments. Typically the most common change in assignment is replacement (Rule 4). For example, close to $2,000$ prefixes have their assignments replaced in DP-BGP during the first iteration. The replacement rule usually corrects for the effects caused by sharing of addresses between sibling AS's, inaccuracy of ICMP source IP address, and routing anomalies. This provides a good indication that the dynamic programming algorithm is performing fine-grained correction. The previous study [73] did not consider replacement as a way to correct IP-to-AS mappings, as it did not optimize over all the paths systematically. Our algorithm is a significant improvement in this regard. We observe that the "Actual gain" is almost always much larger than the "Expected gain" during the first several iterations. For instance, the gain obtained in iteration 1 is $197,477$, which is 35% more than the expected gain of $146,086$. This shows that value of reoptimizing and iterating.

Based on the results of DP-BGP, we conclude that the dynamic programming algorithm is very effective in reducing the mismatches between traceroute and BGP AS paths, while making a relatively small number of corrections in the mapping. As shown, the number of paths with errors is reduced by more than 60% from 14.7% to 5.23%. Similarly, the total number of errors is cut by close to 65%. The number of prefixes with errors is reduced by 37%. All this is accomplished by making changes to the assignments for only about $3,000$ prefixes. A close look at the distribution of the changes made based on each rule indicates that about 59% of the corrections made are replacements, 33% involve creating a pair of AS's, 8% involve adding an AS to a MOAS, and a negligible number involve deletion from a pair of AS's.

To compare our algorithm with the previous study [73], look first at the final line for DP-BGP and the first line for DP-HO, which evaluates the heuristically obtained mapping from that study. There are 9.27% mismatched pairs initially as shown in Iteration 0 for DP-HO, as compared to the 5.23% for the final DP-BGP mapping. Moreover, our approach, when applied to the heuristically obtained mapping, reduces the number of mismatched pairs even further, to 3.08%. This further improvement can probably be attributed to the fact that the heuristically obtained mapping is more liberal in its assignment of MOAS's than is the BGP-table-based mapping. It allows identified IXP's to map to any AS and any prefix that maps to one of a pair of sibling AS's is assumed to map to both. It is noteworthy that during Iteration 0 of DP-HO, 165 prefixes are affected by Rule 1 which deletes an AS from an AS pair. This is a significantly higher number than that for DP-BGP and hence an indication that the liberality of the heuristically obtained mapping is not totally warranted.

Table 6.15 compares the distribution of mapping sizes for all the 105,068 prefixes encountered in the data set for DP-HO and DP-BGP. The mapping size is the number of AS's a prefix maps to. A small number of prefixes in DP-HO map to all AS's as they are identified to be IXP's. We note that significantly more prefixes map to a single AS in DP-BGP than in DP-HO – 93.23% compared to 84.31%, as DP-BGP provides more fine-grained correction and is less liberal in adding AS's to mappings.

6.9.3 Robustness in Initial IP-to-AS Mapping: DP-OM

The third experiment DP-OM in Table 6.14 illustrates the resilience of our algorithm with respect to errors in the initial IP-to-AS mapping. The errors are introduced by assigning a randomly chosen 10% of the prefixes to a single non-existent "dummy" AS. The effect is quite obvious as

Mapping size	DP-HO	DP-BGP
all AS's	56 (0.05%)	0
1	88583 (84.31%)	97958 (93.23%)
2	16278 (15.49%)	6762 (6.44%)
3	135 (0.13%)	289 (0.28%)
4	13 (0.01%)	42 (0.04%)
5	3 (0.00%)	4 (0.00%)
6-15	0	13 (0.01%)

Table 6.15: Distribution of mapping size: number of AS's each prefix maps to.

more than 50% of the pairs contain unavoidable errors in Iteration 0, and the number of errors is more than four times that for the initial BGP-table-based mapping! Remarkably, the iterative algorithm steadily reduces the errors in the mapping and converges with only 6.57% of the pairs having unavoidable errors, fairly close to the final result in DP-BGP. We observe that, as with DP-BGP, the replacement rule (Rule 2) is the most frequently invoked, although here with much higher frequency. Typically, the rule when invoked replaces the dummy AS with the correct mapping.

6.9.4 Robustness in the Set of Pairs Used: DP-OD, DP-OS

So far all our analysis is based on the entire traceroute and BGP data set collected in the previous study [73], where an extensive set of traceroute measurements were performed as a first attempt to study techniques needed for an accurate AS-level traceroute tool. In that study, traceroute targets were selected based on the local BGP table; two IP addresses were probed from each prefix, yielding at least 200,000 IP addresses from each vantage point chosen. From all eight vantage points, we have over 1.8 million traceroute results. Clearly, if one can reduce the number of probes without sacrificing the accuracy of the resulting mapping, it becomes more practical to perform the probing regularly, as will be necessary if we want to keep our mapping up to date. Minimizing the number of probes is also an important requirement for the scalability and efficiency of our desired

AS-level traceroute tool. Previous work [22] suggested that increasing probing sources has less benefit than adding probing destinations for the purpose of discovering network topology. We study the effect of reducing both probing sources and destinations on our final mapping.

Each traceroute measurement provides the correlation between a prefix level path and its corresponding BGP path from the local BGP table. Some quick analysis shows that there is significant redundancy in our overall set of pairs, and so smaller sets might suffice. One way to reduce redundancy is to simply probe one destination IP address for each unique BGP path at a given source, even if that path corresponds to multiple prefixes. We simulate this by randomly selecting just one of our pairs for each BGP path at each source. This reduces our work: DP-OD uses only 242,836 pairs – 13% of the measurements in the full data set. However, it sacrifices coverage of the address space, since many measurements sharing the same BGP path to different IP addresses differ in their prefix-level paths. Applying our dynamic-programming-based approach to this set of pairs and the BGP-table-based mapping (DP-OD in Table 6.14) we get surprisingly good results. The first eight rows indicate iterative improvement in reducing errors and mismatched paths similar to that for DP-BGP. However, we should focus our attention on the last row, marked with "F." This shows the result of taking the final mapping obtained from the reduced set of pairs and evaluating it against the entire data set of all 1.8 million BGP-traceroute pairs. The result is quite close to the final result in DP-BGP with only slightly more errors and mismatched pairs.

DP-OD focuses on reducing the probing destinations. It is equally important to understand the impact of probing sources on the usefulness of measurement data. To study this impact, we picked four vantage points (AT&T Research, Univ of Washington, Nortel, and Peak Web Hosting) out of the eight locations in Table 6.1, based on their diversity in topology and network connectivity. This roughly halved the numbers of measurements and pairs. The results are included under DP-OS in Table 6.14. Once again, the final matching, when evaluated against the full set of pairs, is almost as good as that obtained using the full set. Indeed, it is slightly better than that obtained for DP-OD, perhaps reflecting the fact that here we used a bigger subset of pairs. Both results together, however, help make the case that our approach is robust with respect to the set of input pairs we use, although at least a small price must be paid for using less complete sets. In both cases we still get substantially fewer mismatches than did the heuristically optimized mapping of [73], even though the latter used the full set of pairs.

6.9.5 Comparing All Experiments

We now compare the mappings resulting from all five experiments (DP-BGP, DP-HO, DP-OM, DP-OD, and DP-OS), the initial BGP-table-based mapping (BGP), and the heuristically optimized mapping of [73] (HO-BGP). Table 6.16 shows for a given combination of mapping and comparison target, how many assignments are the same and how many are completely disjoint. Some of the mappings may overlap and constitute the third case. Let us first look at how DP-BGP and HO-BGP compare to BGP, the mapping from which they are both derived. Surprisingly DP-BGP modifies the BGP assignments for less than 3% of the prefixes compared to 10% for HO-BGP. This comparison shows that our dynamic programming based algorithm makes significantly fewer changes to BGP, while reducing the number mismatched pairs by a significantly larger amount. (There are no disjoint prefixes between HO-BGP and BGP, as HO never deletes any AS's from an assignment.)

The rest of the table shows comparisons with DP-BGP, to illustrate the effect of perturbing the initial mapping or reducing the set of pairs. The mappings from all four experiments – DP-HO, DP-OM, DP-OD, and DP-OS agree with that for DP-BGP in over 90% of the assignments. DP-HO is the worst, as it starts from HO-BGP rather than BGP. DP-OS does remarkably well, as the agreement is close to 100%. This explains why the mapping works so well over the entire data set.

Mapping	Identical	Disjoint	Comparison target
DP-BGP	97.10%	1.76%	BGP
HO-BGP	90.00%	0.00%	BGP
DP-HO	90.28%	0.04%	DP-BGP
DP-OM	95.84%	3.40%	DP-BGP
DP-OD	97.80%	1.16%	DP-BGP
DP-OS	99.56%	0.10%	DP-BGP

Table 6.16: Comparing the similarity of newly created IP-to-AS mappings.

Category	Existing prefixes	Created /24's
Rule 1: Delete from MOAS-pair	0.00%	0.23%
Rule 2: Replace a singleton	12.92%	46.00%
Rule 3: Create a MOAS-pair	10.69%	25.34%
Rule 4: Add to MOAS	1.77%	3.05%

Table 6.17: DP-BGP: distribution of final 3050 mapping changes.

In the future, we plan to develop a scheme combining DP-OD and DP-OS to further reduce probing.

6.10 Validation

In general, validating an IP-to-AS mapping is hard due to the lack of information in network configurations. We must rely on information obtained from other sources, both public and internal, which themselves are incomplete. Also, since our approach is different in nature from previous ones, different sorts of validation are needed. An important distinction between our new algorithm and previous work is that the changes to the IP-to-AS mappings are all based on prefixes and what our dynamic program matches them to, and not directly based on inferred relationships between AS's or on inferred IXP's. Therefore, our validation is based on finding explanations for the changes rather than on confirming the inferred relationships. Typically, one type of change can have several explanations.

As in [73], we first use router configuration data from AT&T's network, AS 7018, and the $whois$ entry for the corresponding AS's (using organization names) to validate changes of IP-to-AS mappings where AS 7018 is involved. We found a total of 54 such cases. Recall that the IP-to-AS mappings resulting from the dynamic-programming-based approach could have two categories of prefixes: prefixes existing in the original IP-to-AS mapping and newly created /24 prefixes. Changes can be applied to either an existing prefix or a new /24 prefix.

As seen in Table 6.17, replacement is the most common change applied to an IP-to-AS mapping. Mismatches are fixed by replacing the AS in the original mapping, commonly by its sibling or customer AS. Sibling AS's are owned and managed by a single organization. They may share address spaces, with one AS numbering some of its equipment using part of an address block originated by another. In addition, an AS does not necessarily announce the addresses assigned to its equipment via BGP. Sometimes these addresses fall into larger address blocks originated by its sibling or provider.

In our validation, we consider the replacement 7018 by another AS X to be valid if AS X is a sibling or a customer of AS 7018. Using the configuration data, we are able to verify 18 out of 22 such cases; they are all customers of AS 7018. The replacement of another AS Y by AS 7018 is valid if AS Y is a sibling or customer of AS 7018. We can verify 7 out of 10 such replacements. The cases we were unable to validate could be due to errors in the inference, but are most likely due to incomplete or not up-to-date router configuration data.

Our algorithm could also produce new MOAS prefixes. These are commonly due to sibling AS's, customer AS's, or IXP's. In all the cases that correspond to customers of AS 7018, the prefix was specified in a static route associated with one or more access links to a customer. This means AS 7018 originated the route to this prefix on behalf of a customer; thus the prefix referred not to the equipment inside the backbone but rather to addresses in the customer's network. We found 16 newly identified MOAS prefixes involving AS 7018, 14 of them correspond to siblings or customers of AS 7018. There are also 6 modified MOAS prefixes, 2 of them are IXP's connected with AS 7018 and the rest are siblings or customers of AS 7018. Overall, we confirm 45 out of 54 changes (83%) to the IP-to-AS mappings. Though we are not able to validate all the cases, we did not find any evidence that contradicts the changes that the dynamic programming algorithm applied.

For the rest of the inferred MOAS prefixes that are not verified using local configuration data, we queried $whois$ using the AS number or prefix to see if the description contained the words "exchange point" or "Internet exchange". This check succeeded for 24 of our 1246 MOAS inferences. Then, we compared our results against a list of known IXP's that was derived from [8] and [98]. This confirmed 24 of the inferences, including well-known IXP's over the entire world such as Internet Exchange in Japan, Vienna, Paris, Hong Kong, New York, and San Jose. Comparing to the results in [73], the new approach identified 11 additional IXP's among the MOAS prefixes. Most of them are in Europe or of small size in the U.S., among them are IXP's in Vienna, Switzerland, France, London, San Diego, New York, Chicago, and Miami. In the previous approach, the fan-in and fan-out AS count for these prefixes is less than 2, causing them to be missed. Using both methods, we identify 38 of the inferred MOAS prefixes as IXP's. We also randomly sampled 10 out of the remaining 1208 MOAS prefixes. We found 4 that appear to have similar names and are likely siblings, 3 that appear to have a customer provider relationship since one of the AS's is a tier-1 ISP and the address has been divided into smaller blocks, and the last 3 cases are unclear based on registry name information.

With more complete data sources, a more thorough validation might be possible, but even the limited amount presented here, together with our mismatch statistics, supports the hypothesis that our dynamic-programming-based approach produces an IP-to-AS mapping of substantially improved quality and can be quite effective in the context of an AS-level traceroute tool.

6.11 Dynamic Programming Details

We present in this section the details of the dynamic programming algorithm used to improve IP-to-AS mapping.

Let A be a mapping with associated prefix set P_A, where A assigns a set $A(x)$ of AS's to each prefix $x \in P_A$. Suppose we are given a pair (p, q), where $p = (p_1, p_2, \ldots, p_n)$ is a sequence of IP-addresses in P_A and $q = (q_1, q_2, \ldots, q_m)$ is a sequence of AS's. Our goal is to find a matching a which minimizes $E_A(a, p, q)$, as defined in Section 6.7.

For any IP-address p_i and AS q_j, let $c(p_i, q_j)$ be the penalty for matching p_i to q_j. In our case $c(p_i, q_j)$ is 0 if $q_j \in A(P_A(p_i))$, where $P_A(p_i)$ is the longest prefix in P_A that matches p_i, and 1 otherwise. The following algorithm will however work for arbitrary choices of c. If a is a matching for (p, q) and $1 \leq h \leq n$, define

$$\begin{aligned} C(a,p,q,h) &= \sum_{i=1}^{h} c\Big(p_i, q_{a(i)}\Big) \\ &+ \Big(a(h) - \Big|\{a(i) : 1 \leq i \leq h\}\Big|\Big). \end{aligned}$$

This is the cost of the matching, restricted to the first h prefixes in p. Then, for $1 \leq i \leq n$ and $1 \leq j \leq m$, define $B(p, q, i, j)$ to be the minimum value of $C(a, p, q, i)$ over all matchings a with $a(i) = j$.

Note that $\min \{B(p, q, n, j) : 1 \leq j \leq m\}$ is the optimal cost of a matching for p and q, under the assumption that there is no penalty for failure to match AS's after $a(p_n)$. We can compute this minimum using straightforward dynamic programming. The initial conditions are

$$
\begin{aligned}
B(p,q,1,j) &= c(p_1,q_j)+j-1,\ \ 1 \le j \le m. \\
B(p,q,i,1) &= \sum_{h=1}^{i} c(p_h,q_1),\ \ 1 \le i \le n.
\end{aligned}
$$

For the recurrence relation, we have that for $2 \le j \le m$ and $2 \le i \le n$,

$$
\begin{aligned}
&B(p,q,i,j) = c(p_i,q_j) \\
&\quad + \min_{1 \le h \le j} \Big\{ B(p,q,i-1,h) + \max(0,j-1-h) \Big\}
\end{aligned}
$$

These can be computed in order of increasing j, and for each j in order of increasing i. In case of ties, we choose the maximum h.

The actual algorithm for computing the matching will of course have to keep a record of which option was the minimum for each i, j:

Let $M(p,q,i,j)$ be the index of the AS assigned to the $i-1$st prefix under the assignment determining $B(p,q,i,j)$. In other words, $M(p,q,i,j)$ is that h, $1 \le h \le j$, that minimizes $B(p,q,i-1,h) + \max(0,j-1-h)$.

The optimal assignment is then computed as follows:

$$
\begin{aligned}
a(n) &= \text{argmin}\Big\{ B(p,q,n,j) : 1 \le j \le m \Big\} \\
a(i) &= M\Big(p,q,i+1,a(i+1)\Big),\ \ 1 \le i < n
\end{aligned}
$$

Note that the running time for this algorithm is $O(nm^2)$. A more complicated algorithm can reduce this to $O(nm)$. Given its higher constant-factor overhead and the fact that m was so small in our instances, we did not implement this version, but present its details below as they may be of interest to future researchers.

We now use two variables in our recurrence. The first is simply the variable $B(p,q,i,j)$ from our first algorithm, which we will now call $B_1(p,q,i,j)$. The second, $B_2(p,q,i,j)$, is defined to be the minimum value of $C(p, q, a, i)$ over all matchings a for which $1 \leq a(i) < j$ and $a(i+1) \geq j$. The initial conditions for B_1 are the same as before, while those for B_2 are

$$\begin{aligned} B_2(p, q, 1, j) &= \min_{0 \leq h < j} \Big(c(p_1, q_h) + j - 1 - h \Big) \\ B_2(p, q, i, 1) &= i \end{aligned}$$

for $1 \leq j \leq m$ and $1 \leq i \leq n$. The recurrence relations for B_1 and B_2 now become

$$\begin{aligned} &B_1(p, q, i, j) = c(p_i, q_j) \\ &\quad + \min \Big(B_1(p, q, i-1, j), B_2(p, q, i-1, j) \Big) \end{aligned}$$

$$\begin{aligned} &B_2(p, q, i, j) = \\ &\quad \min \Big(B_1(p, q, i, j-1), B_2(p, q, i, j-1) + 1 \Big) \end{aligned}$$

To construct the matching we store new variables $R_1(p, q, i, j)$ and $R_2(p, q, i, j)$, where $R_h(p, q, i, j)$ is the index (1 or 2) of the function that provided the minimum in computing $B_h(p, q, i, j)$. We deduce the matching from these variables as follows:

As before, we let $a(n)$ be the j that minimizes $B_1(p, q, n, j)$. Given that $a(i) = j$ has been computed and $i > 1$, we compute $a(i-1)$ as follows.

1. If $R_1(p, q, i, j) = 1$, set $a(i-1) = j$.
2. Otherwise,
   (a) Set $h = j$.
   (b) While $R_2(p, q, i, h) = 2$, set $h = h - 1$.
   (c) Set $a(i) = h - 1$.

6.12 Summary

In this chapter, we have proposed techniques for improving how IP addresses of network infrastructure are mapped to the administering ASes. These techniques rely on a measurement methodology for (i) collecting both BGP and traceroute paths at multiple vantage points and (ii) using an initial IP-to-AS mapping derived from a large collection of BGP routing tables. We proposed simple heuristics for resolving traceroute paths with "*" and unmapped IP-level hops and describe how to verify the results using internal configuration data. Then, we presented heuristics that compare the BGP and traceroute AS paths to identify IXPs, sibling ASes, and other ASes that "share" address space, and evaluated the improved IP-to-AS mapping on traceroute paths collected from three vantage points. Compared to an initial IP-to-AS mapping constructed from the BGP tables, our heuristics reduced the fraction of incomplete paths from 18–22% to 6–8%; the ratio of matched to mismatched paths more than doubled, increasing from around 9–12 to 25–35. The adjustments to the IP-to-AS mapping are crucial for building an accurate AS-level traceroute tool for network operators and researchers. In addition, the improved mapping helps in highlighting the small number of important cases when the traceroute and BGP AS paths actually differ.

Our techniques capitalize on certain operational realities which arguably could change over time. For example, we were able to include more than 99% of the BGP AS paths in our analysis because most BGP routes are relatively stable and few BGP AS paths have private ASes or AS_SETs. We also exploited the fact that most ASes assign public, routable addresses to their equipment and often give meaningful domain names to the interfaces. Although quite a few traceroute hops did not return ICMP replies, most of the "*" hops occurred near the ends of paths or between other hops in the same AS. In addition, our techniques build on the assumption that the AS-level signaling and forwarding paths typically (though not always) match. This assumption would become less reasonable if route filtering were applied more aggressively in the core of the Internet, or if routing anomalies such as deflections were very common. Also, if the practice of "multi-homing without BGP" becomes more common, the notion of "origin AS" would become increasingly ambiguous. We plan to investigate the sensitivity of our results to these factors.

Converting an IP-level path to an AS-level path is extremely difficult, and additional measurement data would help. An accurate router-level graph [13, 31, 45, 99] would allow us to map interfaces to routers and, in turn, map routers to ASes. This would make our techniques less vulnerable to the interface numbering at AS boundaries (Section 6.6.2) and the source IP address in ICMP messages (Section 6.6.3). Although challenging in its own right, collecting the router-level topology does not require joint collection of BGP update messages, expanding the set of possible locations for launching the necessary traceroute probes. Our efforts would benefit from collecting both traceroute and BGP data at more locations, particularly in Europe and Asia. We are working on expanding the number and diversity of locations where we collect our data. Also, we are exploring the use of the public traceroute servers despite the many challenges they introduce. In particular, we are investigating ways to reduce the amount of measurement data needed from each vantage point to lower the load we would impose on the public servers.

Ultimately, developing an accurate AS traceroute tool depends on having a platform for collecting and managing information about the Internet infrastructure. Having a generic distributed platform, supported by service providers, for collecting and combining the traceroute and BGP data would be extremely valuable. Going one step further, computing the AS-level traceroute path would be much easier if ASes kept an up-to-date list of the address blocks used to number their equipment. This would simplify the interpretation of the source addresses in the ICMP messages. ASes could still protect access to their infrastructure from possible attack by filtering packets and routes that refer directly to their equipment. Alternatively, the ICMP specification could be extended to include an AS number or other identifying information in ICMP replies. In addition, the ICMP specification could be augmented to clarify whether the source address of the ICMP response messages refers to the incoming or outgoing interface at the router.

In this chapter we presented the third piece of our framework for improved understanding of BGP dynamics – AS-traceroute tool, the enabling tool for correlating the routing plane with the data plane. Integrating our tool with the BGP Beacons presented in Chapter 5, we can understand how the injected routing changes affect the data packets by sending a continuous stream of data packets towards the beacon prefixes from multiple vantage points. This tool also allows us to understand the impact of routing dynamics on the data plane. In the case of any mismatches between the two, the tool allows us to explore potential routing anomalies. The tool is also critical for evaluating routing performance in the context of how applications are affected. In the next chapter, we summarize our experience in building this infrastructure for improved understanding of BGP dynamics and suggestions for future work.

Chapter 7

Conclusions and Future Work

7.1 Thesis Summary and Discussion

In this chapter, we summarize the work presented in this dissertation and give some discussions on future work. The main contribution of this dissertation is to design a framework for providing more visibility in understanding complex BGP run-time behavior. Our framework consists of three components: analysis, measurement, and traffic correlation. We demonstrate the usefulness of this framework by illustrating how we discover, analyze, and correct an example of bad BGP dynamics: flap damping delays convergence of stable routes experiencing only a single originating route change. The problem is discovered and analyzed in detail through simulations, trace analysis of passive measurements, and a router testbed to understand actual implementation details. It is further validated through the use of an active measurement infrastructure – BGP Beacons. The solution proposed to remedy the interaction is also evaluated extensively using simulations. To evaluate BGP performance in the context of application behavior, we develop an AS-level traceroute tool that allows us to understand the discrepancies between the BGP routing paths and the packet forwarding paths. Such mismatches often indicate routing anomalies, *i.e.,* problems with BGP dynamics.

The framework developed in this work is a research methodology that allows researchers to analyze any large interdomain routing systems. It also provides insight in understanding the dynamics of large distributed systems spanning across multiple domains. We believe the principles of performing controlled measurement, correlating the control plane with the data plane, and test-bed based understanding of implementation variants are reusable methodologies.

In Sections 7.1.1, 7.1.2, and 7.1.3, we describe our experience and lessons learned from building the three components in the framework and analyzing the route flap damping problem. If we had a chance to build BGP from scratch, such measurement components should be *built into* the routing infrastructure to allow easier diagnosis of routing problems. We describe our vision for future work in Section 7.2.

7.1.1 Route Flap Damping: Fast Convergence vs. Stability

In the routing area, there has always been this tradeoff between good performance by reacting fast to changes and the stability to prevent oscillations and reduce overhead due to transient changes. The old saying of "propagating bad news fast, good news slowly" [63] applies very well here. The good news refers to the availability of a route with higher local preference. The bad news refers to the withdrawal of an existing route. In fact, we can augment that by saying that if the route is previously not available, the re-announcement of a route should be allowed to propagate faster than the case when a suboptimal route is being replaced by a more preferred route. In the former case, the reachability is initially not available for the route; thus, restoring the reachability is deemed to be more important. In the latter case, the reachability is already available even without the new update. However, if the bad news or good news are not stable, it would be best to ignore it.

Route flap damping attempts to estimate the stability of the routes by observing the past history of the route. Since we don't have the ability to tell the future, nor can we infer the cause of any update in BGP, we can only try to use the past to predict the future. This is assuming that an unstable route in the past will more likely to be unstable than a stable route with little updates in the past. Such a mechanism is only a heuristic and cannot deal with slowly flapping routes. Given the route flap damping algorithm using exponential decay, we can calculate the highest rate that a route can flap without being suppressed *i.e.,* the penalty never goes above the suppress threshold. This rate can be expressed as

$$(Suppress\_threshold - Smallest\_penalty\_increment) = Suppress\_threshold * e^{(ln2/H)*t}$$

Using Cisco's default parameter, the solution for t is about 6.22 minutes. Thus, a route flapping once every 6.22 minutes will not be punished based on the current scheme. In the above the equation, the type of flap with the smallest penalty increment is attribute change, *i.e.,* an announcement preceded by an announcement with different route attributes. The penalty for attribute change for both Cisco and Juniper is only 500. This shows that route flap damping is merely a heuristic, and cannot fix flaps caused by oscillations due to policy conflicts or other routing anomalies. It only alleviates the symptom by reducing the number of updates propagated to reduce router overhead.

One of the problems we address in this dissertation is understanding how such a heuristic may be triggered unexpectedly. Route flap damping treats all updates the same, without knowing the cause of the updates. Thus, we may conjecture that it may be triggered or a route may be suppressed unexpectedly due to transient convergence. BGP is well-known for its path exploration, where alternate routes are explored before a route is finally withdrawn. Thus, it is not surprising that flap damping or route suppression may be inadvertently triggered during the BGP convergence process. This can occur very frequently in certain topologies with multiple alternate paths due to the aggressive default settings of route flap damping in current router implementations. This is clearly a tradeoff between the ability to maintain stability on the Internet by reducing number of updates and the need for propagating changes quickly for fast failure recovery. Our work indicates that to achieve the right balance between the two, there needs to be some feedback-driven adaptation described below.

We have proposed an incrementally deployable modification to route flap damping that attempts to differentiate update sequences caused by transient path exploration. However, our change is not a panacea, it only reduces the likelihood but does not completely eliminate the occurrence of flap damping triggered during convergence. We should also follow a more liberal approach of setting the parameter setting, given the understanding of how many updates routers can handle today. Another way to achieve the better tradeoff or prevent flap damping being triggered during transient routing changes is to build in *feedback* into the system. There is no single magic setting of flap damping parameter that works for the entire Internet. One avenue of future work in this area is to use feedback to signal one peer for it to slow down the sending of updates. This is similar to flow control in TCP. This way, the updates can be sent as fast as possible given the rate of the peering router's ability to handle updates. And the rate of sending can be adapted dynamically over time.

7.1.2 BGP Beacons: an Active BGP Measurement Infrastructure

Because of the difficulty of interpreting BGP updates to calculate metrics such as convergence times, we built from scratch an active measurement infrastructure for controlled BGP experiments. If we know exactly when the routing change occurs and what the routing change is, it becomes much easier to correlate the observed updates with the given injected routing change. We were very fortunate to construct such an experiment infrastructure with support from many operational folks. During the course of the study, we realize that even with the controlled measurement infrastructure, ambiguity still exists in how to interpret the BGP updates. In this case, the interpretation means identifying updates associated only with our injected routing changes. External routing changes can occur, affecting the Beacon prefixes; thus, it is extremely important to ignore the Beacon events affected by external changes. Our technique of using Anchor prefix is useful; however, it may be treated differently from the Beacon prefix. As a result, we may not be absolutely certain to have eliminated the effect of external routing changes. However, statistically, these occurrences are rare. We are only beginning to make use of the Beacons infrastructure. There are many areas of future work. But, so far, it has shown to be useful in validating effects such as route flap damping on the Internet.

7.1.3 AS-traceroute Tool: How Happy are the Packets?

Initially, when we started out constructing a tool of AS-traceroute to identify AS-level forwarding paths, we never expected the difficulties and uncertainties in the task. Internet measurement data such as traceroute data can always contain a lot of "crud" due to diverse operational practices. It has been a useful exercise to understand how to deal with the operational reality and classify them into various categories for the purpose of inferring the owner ASes of infrastructure address spaces. However, validation is a serious problem, as with any inference work on the Internet. We did our best to validate using operational data, but it is not perfect. As future work, it may be more productive to follow a paradigm shift – build a knowledge plane like [34] infrastructure where such measurement data can be shared across networks. Each network itself knows accurately its IP-to-AS mapping, so sharing of such information across networks is feasible.

7.2 Future Work

In the near term, I plan to do happy packet measurement by making use of PlanetLab testbed [90] and use the AS-traceroute tool along with the Beacons infrastructure to understand packet forwarding behavior during routing changes. For the longer term, there are several areas of interest.

7.2.1 How to Debug the Routing System?

Problem

Today, network operators have very limited tools to debug routing problems. Only primitive tools such as traceroute and ping are available to identify existing routing behavior. There is very little visibility into the routing behavior of other ISPs' networks from a given ISP's perspective, making it even more difficult to identify the culprit of any routing anomalies. This also means that it is difficult to predict the impact of any routing policy change has on the global routing behavior. Oftentimes, routing problems are noticed only after a customer complains about reachability or severe degradation of performance. There is a lack of proactive, automated analysis of routing problems. As certain routing problems initially may not be very obvious, but subsequently may result in suboptimal, unintended routes. Diagnosing Internet routing problems often require analysis of data from multiple vantage points.

Proposed Solutions

1. Build routing assertions, so that nothing fails silently. When network operator configures a network, it is important to create a set of assertions, equivalent to integrity constraints in database or assertions in software programs. This generates the expected behavior of the routing protocols in terms of which routes are allowed, the resulting attributes of the routes, etc.

2. Cooperation among networks: Each network builds a measurement repository to collect data

from multiple locations. It builds a profile of the expected routing behavior to quickly identify any deviations using statistical techniques. Cooperation across networks is absolutely necessary to diagnose global Internet routing problems. It is a challenge to provide summaries of measurement data at sufficiently detailed level to be useful but without revealing sensitive information about internals of ISP's networks. A complementary approach is to allow special distributed queries of the detailed network data from multiple vantage points without direct access to the data.

3. Scalable distributed measurement interpretation and measurement calibrations: Routing measurement (e.g., BGP) can result in significant data volume and it may be infeasible to perform real-time or online interpretation of such measurement data by combining all the data from multiple locations in distinct networks at a centralized location. Distributed algorithms are useful to interpret measurement results locally and then aggregate them intelligently to identify routing anomalies. Interpreting measurement can be challenging as there is a lack of global knowledge of topologies and policies which can arbitrarily translate a given measurement input signal to observed output signals. We propose the use of calibration points to help identify expected or normal routing behavior and correlate the output with the input. Calibration points are well-controlled active measurement probes with known measurement input. The BGP Beacons work is one such example of attempting to understand the patterns of output for a known input routing change.

4. Internet-wide emulation for network configurations: The impact of a single routing configuration change caused by a policy change for example could be global; thus, it is important to emulate the behavior in advance to study its impact. It is useful to abstract the routing behavior in a single network at a higher level to study the perturbation on the global routing

system. Currently, the routing configuration is done at a device level. Higher-level programming support is needed to provide semantically more meaningful configuration of networks. Predicting the output of a routing configuration implicitly assumes that routing is deterministic. However, nondeterministic routing may be more stable. Such tradeoffs are important to study.

5. Understanding the interaction of multiple routing protocols and implementation variants: Internet routing consists of multiple protocols, e.g., interdomain, intradomain routing protocols, and MPLS label distribution protocol. All these protocols interact to achieve end-to-end routing behavior from an application's point of view. It is critical to understand their dependency on each other. For instance, in BGP/MPLS IP VPNs, the label distribution protocol is needed to set up label switched paths across the network and if that is unsuccessful, BGP cannot find a route. There is similar dependence of BGP on OSPF or IS-IS. Implementation variants among router vendors determine routing dynamics which is poorly understood. The interaction among the variants may result in unexpected behavior and needs to be studied.

6. Understanding routing ”politics”: When a customer complains about routing problems either in terms of reachability or poor performance, it typically is in the context of some applications. Network operators install route filters in the routers to determine which routes to accept in calculating the best path to forward traffic. Packet filters at the routers are much more flexible in the sense that they determine which packets are accepted for forwarding based on attributes of the packets, e.g., port numbers, protocol types. Given a route in one's routing table received by one's upstream provider, there is no guarantee that all application traffic can reach the destination due to the presence of packet filters. Some networks, for instance, perform port-based filtering to protect against known worm traffic. When debugging routing problems, we

need to view from application's perspective to understand which type of application traffic is correctly forwarded.

7.2.2 How to Improve the Application Performance?

Problem

Today, the Internet has no performance guarantees for real-time or delay-sensitive applications, such as VoIP, gaming, especially if traffic goes across multiple networks. To obtain flexible routing in terms of control over cost and performance of network paths, end users resort to either multihoming to multiple networks or overlay routing. However, a recent study **??** has shown that there may be potential adverse interaction between application routing and traffic engineering at the IP layer. Multihoming, similarly, is not a perfect solution as it does not directly translate to paths with performance guarantees, has little impact on how incoming traffic reaches the customers, and may further amplify the amount of routing traffic during convergence.

Proposed Solution

The Application is king: correlate routing with forwarding plane, evaluate and improve using application performance metrics: delay, loss rate, jitter. When studying routing protocol performance, researchers often use convergence delay as a universal metric. However it does not translate directly to metrics applications care about, e.g., delay, loss rate, and jitter. Understanding the stability of such measurements as a function of the network locations provides a way for overlay routing algorithms to intelligently route around network problems. Application performance measurements also expose the detailed interaction between the dynamics of forwarding plane and control plane.

7.2.3 How to Protect the Routing System?

Problem

There has been relatively few studies on protecting the Internet routing infrastructure against attacks. Vulnerabilities in router architectures are relatively unknown and have not been widely exploited. The routing system can also be indirectly affected due to enormous traffic volume. Recently, there has been a large number of worms exploiting end host OS vulnerability. Significant attack traffic volume causes router sessions to time out and session resets result in exchange of entire routing tables and disruption of routing. Cascaded failures can occur if the session reset traffic subsequently cause router overload and other peering sessions to be affected.

Proposed Solution

1. Understanding vendor implementation of routing protocols: Through detailed black-box testing and support from vendors, we can better understand the obscure, undocumented ”features” of routers that are not documented in RFCs and their implication on router security.

2. Understanding vulnerability points on the Internet: Network topology and policy information are more widely known through various Internet mapping effort. It is important to discover vulnerability points by analyzing failure scenarios.

3. Higher priority for routing traffic: The delay and loss of routing traffic, especially keepalive HELLO messages, can cause sessions to reset. This can occur when there is significant data traffic. Increasing the queuing and processing priority of routing packets in the routers is one possibility to reduce the impact of bandwidth attacks on the routing system.

4. Automated dynamic installation of packet and route filters The attack against windowsu-

pate.com was prevented just in time by invalidating the relevant DNS entry in the DNS system, which takes at least 24 hours to propagate any change globally. To react to any attacks in real time, there needs to be a faster and automated way. One possibility is dynamically install relevant packet and route filters across a selected set of networks to eliminate/reduce the impact of the attacks. Routers have limited memory for such filters and the order of the filters determine the actual routes or packets permitted. We need to study efficient algorithms to compute such filters on the fly.

Bibliography

[1] Benchmarking Methodology (bmwg). http://www.ietf.org/html.charters/bmwg-charter.html. IETF Working Group.

[2] From private email exchanges with Randy Bush.

[3] Global Routing Operations (grow) Charter. http://www.ietf.org/html.charters/grow-charter.html. IETF Working Group.

[4] GNU Zebra – Routing Software. http://www.zebra.org/.

[5] Inter-Domain Routing (idr) Charter. http://www.ietf.org/html.charters/idr-charter.html. IETF Working Group.

[6] Nanog Mailing List. http://www.merit.edu/mail.archives/nanog/.

[7] Nanog traceroute. ftp://ftp.login.com/pub/software/traceroute/.

[8] Packet Clearing House. http://www.pch.net/resources/data/exchange-points/.

[9] Prtraceroute. http://www.isi.edu/ra/RAToolSet/prtraceroute.html.

[10] RIPE BGP Beacons. http://www.ripe.net/ris/beacon.html.

[11] Ripe NCC. `http://www.ripe.net/ripencc/pub-services/np/ris/`.

[12] Singaren. `http://noc.singaren.net.sg/netstats/routes/`.

[13] Skitter. `http://www.caida.org/tools/measurement/skitter`.

[14] The SSFnet Project. `http://www.ssfnet.org`.

[15] University of Oregon Route Views Archive Project. `www.routeviews.org`.

[16] University of Oregon Route Views Project. `http://www.routeviews.org/`.

[17] Visualizing Internet topology at a macroscopic scale. `http://www.caida.org/analysis/topology/as_core_network/`.

[18] Abha Ahuja. Talk on BGP Routing Dynamics. Presented at AT&T Labs-Research, Summer 2001.

[19] Lisa Amini, Anees Shaikh, and Henning Schulzrinne. Issues with inferring Internet topological attributes. In *Proceedings of SPIE*, volume 4865, July 2002.

[20] D. Andersen, N. Feamster, S. Bauer, and H. Balakrishnan. Topology Inference from BGP Routing Dynamics. In *Proc. Internet Measurement Workshop*, November 2002.

[21] Paul Barford, Azer Bestavros, John Byers, and Mark Crovella. On the marginal utility of network topology measurements. In *Proc. Internet Measurement Workshop*, November 2001.

[22] Paul Barford, Azer Bestavros, John Byers, and Mark Crovella. On the Marginal Utility of Network Topology Measurements. In *Proc. Internet Measurement Workshop*, November 2001.

[23] Paul Barford and Winfred Byrd. Interdomain routing dynamics. Unpublished report, June 2001.

[24] G. Battista, M. Patrignani, and M. Pizzonia. Computing the Types of the Relationships Between Autonomous System s. In *Proc. IEEE INFOCOM*, March 2003.

[25] H. Berkowitz, E. Davies, S. Hares, P. Krishnaswamy, and M. Lepp. Terminology for Benchmarking BGP Device Convergence in the Control Plane . `http://www.ietf.org/internet-drafts/draft-ietf-bmwg-conterm-05.txt`, June 2003. IETF Working Group Draft.

[26] Randy Bush. Happy Packets. Position Statement Presentation at WIRED Workshop, OCT 2003.

[27] H. Yu C. Alaettinoglu, V. Jacobson. Towards Milli-Second IGP Convergence. IETF Internet Draft: draft-alaettinoglu-ISIS-convergence-00, November 2000.

[28] Don Caldwell, Anna Gilbert, Joel Gottlieb, Albert Greenberg, Gisli Hjalmtysson, and Jennifer Rexford. The Cutting EDGE of IP Router Configuration. In *Proc. Workshop on Hot Topics in Networks (HotNets)*, November 2003.

[29] D Chang, R. Govindan, and J. Heidemann. An Empirical Study of Router Response to Large BGP Routing Table Load. In *Proc. Internet Measurement Workshop*, 2002.

[30] D Chang, R. Govindan, and J. Heidemann. The Temporal and Topological Characteristics of BGP Path Changes. In *Proc. of ICNP*, 2003.

[31] Hyunseok Chang, Sugih Jamin, and Walter Willinger. Inferring AS-level internet topology

from router-level path traces. In *Proc. Workshop on Scalability and Traffic Control in IP Networks, SPIE ITCOM Conference*, August 2001.

[32] E. Chen and S. Sangli. Avoid BGP Best Path Transition from One External to Another. IETF Draft: draft-chen-bgp-avoid-transition-00.txt, 2000.

[33] E. Chen and J. Stewart. A Framework for Inter-Domain Route Aggregation. Request for Comments 2519, February 1999.

[34] D. Clark, J Ramming, and J. Wroclawski. A Knowledge Plane for the Internet. In *Proc. ACM SIGCOMM*, August 2003.

[35] J. Cowie, A. T. Ogielski, BJ Premore, E. Smith, and T. Underwood. Impact of the 2003 Blackouts on Internet Communications. available at `http://www.renesys.com/news/2003-11-21/Renesys_BlackoutReport.pdf`, 2003.

[36] J. Cowie, A. T. Ogielski, BJ Premore, and Y. Yuan. Internet Worms and Global Routing Instabilities. In *Proc. of SPIE*, 2002.

[37] K. Kompella et. al. Virtual Private LAN Service. draft-ietf-l2vpn-vpls-bgp-00.txt, Work in Progress.

[38] N. Feamster. Practical Verification Techniques for Wide-Area Routing. In *Proc. Workshop on Hot Topics in Networks (HotNets)*, November 2003.

[39] N. Feamster, D. Andersen, H. Balakrishnan, and M. F. Kaashoek. Measuring the Effects of Internet Path Faults on Reactive Routing. In *Proc. ACM SIGMETRICS*, June 2003.

[40] N. Feamster and H. Balakrishnan. Towards a Logic for Wide-Area Internet Routing. In *Proc. ACM SIGCOMM Workshop on Future Directions in Network Architecture*, August 2003.

[41] V. Fuller, T. Li, J. Yu, and K. Varadhan. Classless Inter- Domain Routing (CIDR): an Address Assignment and Aggregation Strategy. Request for Comments 1519, September 1993.

[42] L. Gao. On inferring autonomous system relationships in the Internet. *IEEE/ACM Trans. Networking*, December 2001.

[43] L. Gao, T. Griffin, and J. Rexford. Inherently Safe Backup Routing with BGP. In *Proc. IEEE INFOCOM*, March 2001.

[44] Lixin Gao and Jennifer Rexford. Stable Internet Routing Without Global Coordination. In *Proceedings of ACM SIGMETRICS 2000*.

[45] Ramesh Govindan and Hongsuda Tangmunaraunkit. Heuristics for Internet map discovery. In *Proc. IEEE INFOCOM*, 2000.

[46] T. Griffin, A Jaggard, and V. Ramachandran. Design Principles of Policy Languages for Path Vector Protocols. In *Proc. ACM SIGCOMM*, August 2003.

[47] T. Griffin and G. Wilfong. A safe path vector protocol. In *Proc. IEEE INFOCOM*, March 2000.

[48] T. G. Griffin, F. B. Shepherd, and G. Wilfong. Policy disputes in path-vector protocols. In *Proc. International Conference on Network Protocols*, November 1999.

[49] Tim Griffin. What is the sound of one route flapping? Network Modeling and Simulation Summer Workshop at Dartmouth, July 2002.

[50] Timothy G. Griffin and Brian J. Premore. An Experimental Analysis of BGP Convergence Time. In *Proceedings of ICNP 2001*, 2001.

[51] Timothy G. Griffin and Gordon Wilfong. An Analysis of BGP Convergence Properties. In *Proceedings of ACM SIGCOMM 1999*, 1999.

[52] Timothy G. Griffin and Gordon Wilfong. An Analysis of the MED Oscillation Problem in BGP. In *Proc. International Conference on Network Protocols*, 2002.

[53] Timothy G. Griffin and Gordon Wilfong. On the Correctness of iBGP Configuration. In *Proc. ACM SIGCOMM*, August 2002.

[54] S. Halabi and D. McPherson. *Internet Routing Architectures*. Cisco Press, Indianapolis, Indiana, second edition, 2000.

[55] Geoff Huston. BGP Reports. `http://bgp.potaroo.net/`.

[56] Geoff Huston. Analyzing the Internet BGP Routing Table. *The Internet Protocol Journal*, March 2001.

[57] Young Hyum, Andre Broido, and k claffy. Traceroute and BGP AS Path incongruities. 2003. `http://www.caida.org/outreach/papers/2003/ASP/`.

[58] IETF Working Group. IP Performance Metrics (ippm).

[59] John W. Stewart III. *BGP4 Inter-Domain Routing in the Internet*. Addison-Wesley, 1999.

[60] Van Jacobson. Traceroute. `ftp://ftp.ee.lbl.gov/traceroute.tar.gz`.

[61] Atul Khanna and John Zinky. The Revised ARPANET Routing Metric. In *Proceedings of ACM SIGCOMM 1989*, 1989.

[62] C. Labovitz, A. Ahuja, and M. Bailey. Shining Light on Dark Address Space. Technical report, Arbor Networks, Meri Networks Inc., 2001.

[63] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. Delayed Internet Routing Convergence. In *Proc. ACM SIGCOMM*, 2000.

[64] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. Delayed Internet routing convergence. *IEEE/ACM Trans. Networking*, 9(3):293–306, June 2001.

[65] C. Labovitz, A. Ahuja, and F. Jahanian. Experimental Study of Internet Stability and Wide-Area Network Failures. In *Proceedings of FTCS*, 1999.

[66] C. Labovitz, R. Malan, and F. Jahanian. Internet Routing Stability. In *Proc. ACM SIGCOMM*, 1997.

[67] C. Labovitz, R. Malan, and F. Jahanian. Origins of Internet Routing Instability. In *Proc. IEEE INFOCOM*, 1999.

[68] C. Labovitz, R. Wattenhofer, S. Venkatachary, and A. Ahuja. The Impact of Internet Policy and Topology on Delayed Routing Convergence. In *Proceedings of INFOCOM 2001*, 2001.

[69] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. Inferring Link Weights using End-to-End Measurements. In *Proc. Internet Measurement Workshop*, November 2002.

[70] Ratul Mahajan, David Wetherall, and Tom Anderson. Understanding BGP Misconfigurations. In *Proc. ACM SIGCOMM*, August 2002.

[71] G. Malkin. Request for Comments: 2453, December 1998.

[72] Z. Mao, R. Bush, T. Griffin, and M. Roughan. BGP Beacons. In *Proc. Internet Measurement Workshop*, October 2003.

[73] Z. Mao, J. Rexford, J. Wang, and R. Katz. Towards an Accurate AS-level Traceroute Tool. In *Proc. ACM SIGCOMM*, August 2003.

[74] Z. Morley Mao, Ramesh Govindan, George Varghese, and Randy Katz. Route flap damping exacerbates internet routing convergence. Technical Report UCB//CSD-02-1184, U.C. Berkeley, June 2002.

[75] Zhuoqing Morley Mao, Ramesh Govindan, George Varghese, and Randy H. Katz. Route flap damping exacerbates internet routing convergence. In *Proc. ACM SIGCOMM*, 2002.

[76] Pedro Roque Marques). Bgp route advertisement interval. Talk at RIPE 45 Meetings.

[77] J. Moy. *OSPF: Anatomy of an Internet Routing Protocol*. Addison-Wesley, 1998.

[78] J. Moy. OSPF Version 2. Request for Comments: 2328, April 1998.

[79] Ripe NCC. Routing Information Service Raw Data.

[80] W. B. Norton. The Art of Peering: The Peering Playbook, December 2001.

[81] D. Oran. OSI IS-IS Intra-domain Routing Protocol. Request for Comments: 1142, February 1990.

[82] H. Ould-Brahim, E. Rosen, and Y. Rekhter. Using BGP as an Auto-Discovery Mechanism for Provider-provisioned VPNs. draft-ietf-l3vpn-bgpvpn-auto-00.txt, Work in Progress, July 2003.

[83] Christian Panigl, Joachim Schmitz, Philip Smith, and Cristina Vistoli. RIPE Routing-WG Recommendations for Coordinated Route-flap Damping Parameters. available at `http://www.ripe.net/ripe/docs/ripe-229.html`, October 2001. Document ID: ripe-229.

[84] V. Paxson. End-to-End Routing Behavior in the Internet. *IEEE/ACM Trans. Networking*, pages 601–615, October 1997.

[85] V. Paxson. End-to-End Routing Behavior in the Internet. *IEEE/ACM Trans. Networking*, 5(5):601–615, October 1997.

[86] Vern Paxson. *Measurements and Analysis of End-to-End Internet Dynamics*. PhD thesis, U.C. Berkeley, 1997.

[87] D. Pei, X. Zhao, L. Wang, D. Massey, A. Mankin, S. F. Wu, , and L. Zhang. Improving BGP Convergence Through Consistency Assertions. In *Infocom 2002*, 2002. available at `http://fniisc.nge.isi.edu/papers/infocom2002.ps`.

[88] Dan Pei, Matt Azuma, Nam Nguyen, Jiwei Chen, Dan Massey, , and Lixia Zhang. BGP-RCN: Improving BGP Convergence through Root Cause Notification. Technical report, UCLA, 2003.

[89] Dan Pei, Lan Wang, Dan Massey, S. Felix Wu, and Lixia Zhang. A Study of Packet Delivery Performance during Routing Convergence. In *Proc. of IEEE DSN*, 2003.

[90] PlanetLab. `http://www.planet-lab.org`.

[91] J. Postel. Internet Control Message Protocol. RFC 792, September 1981.

[92] Y. Rekhter and T. Li. A Border Gateway Protocol. RFC 1771, March 1995.

[93] Y. Rekhter and T. Li. A Border gateway protocol 4 (BGP-4). IETF Draft: draft-ietf-idr-bgp4-20.txt, April 2003.

[94] J. Rexford, J. Wang, Z. Xiao, and Y. Zhang. BGP routing stability of popular destinations. In *Proc. Internet Measurement Workshop*, November 2002. `http://www.research.att.com/~yzhang/papers/bgp-imw02.pdf`.

[95] Thomas Rodeheffer and Michael D. Schroeder. Automatic Reconfiguration in Autonet. In *Proceedings of 13th ACM Symposium on Operating System Principles*.

[96] Stefan Savage, Andy Collins, Eric Hoffman, John Snell, and Tom Anderson. The end-to-end effects of Internet path selection. In *Proc. ACM SIGCOMM*, September 1999.

[97] Aman Shaikh, Lampros Kalampoukas, Rohit Dube, and Anujan Varma. Routing Stability in Congested Networks: Experimentation and Analysis. In *Proc. ACM SIGCOMM*, pages 163–174, August 2000.

[98] Elena Silenok. Study of Internet eXchange Points.

[99] Neil Spring, Ratul Mahajan, and David Wetherall. Measuring ISP topologies with Rocketfuel. In *Proc. ACM SIGCOMM*, August 2002.

[100] J. W. Stewart. *BGP4: Inter-Domain Routing in the Internet*. Addison-Wesley, 1998.

[101] Lakshminarayanan Subramanian, Sharad Agarwal, Jennifer Rexford, and Randy H. Katz. Characterizing the Internet hierarchy from multiple vantage points. In *Proc. IEEE INFOCOM*, June 2002.

[102] Hongsuda Tangmunarunkit, Ramesh Govindan, Scott Shenker, and Deborah Estrin. The impact of policy on Internet paths. In *Proc. IEEE INFOCOM*, 2001.

[103] Agilent Technologies. Agilent RouterTester Solution. `http://advanced.comms.agilent.com/routertester/`.

[104] Henk Uijterwaal. Routing Beacons. `http://www.potaroo.net/iepg/november2002/`, November 2002.

[105] K. Varadhan, R. Govindan, and D. Estrin. Persistent Route Oscillations in Inter-Domain Routing. *Computer Networks*, March 2000.

[106] C. Villamizar, R. Chandra, and R. Govindan. BGP Route Flap Damping. RFC 2439, 1998.

[107] C. Villamizar, R. Chandra, and R. Govindan. BGP Route Flap Damping. RFC 2439, 1998.

[108] Lan Wang, Xiaoliang Zhao, Dan Pei, Randy Bush, Daniel Massey, Allison Mankin, S. Felix Wu, and Lixia Zhang. Observation and Analysis of BGP Behavior under Stress. In *Proc. Internet Measurement Workshop*, 2002.

[109] X.Zhao, M. Lad, D. Pei, L.Wang, D. Massey, S. F. Wu, , and L. Zhang. Understanding BGP Behavior Through A Study of DoD Prefixes. In *Proc. of DISCEX III*, April 2003.

[110] Xiaoliang Zhao, Dan Pei, Lan Wang, Dan Massey, Allison Mankin, S. Felix Wu, and Lixia Zhang. An Analysis of BGP Multiple Origin AS (MOAS) Conflicts. In *Proc. Internet Measurement Workshop*, November 2001.