The Impact of Internet Policy and Topology on Delayed Routing Convergence

Craig Labovitz, Roger Wattenhofer, Srinivasan Venkatachary
Microsoft Research
{labovit, rogerwa, cheenu}@microsoft.com

Abha Ahuja
Merit Network, Inc.
ahuja@merit.edu

July 2000
Technical Report
MSR-TR-2000-74

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
http://www.research.microsoft.com

0.1 Introduction

The Internet's sustained exponential growth and the continued emergence of new and varied network applications provides testament to the scalability of the backbone infrastructure and protocols. The original TCP/IP decision to place network intelligence and state almost exclusively on end-nodes has enabled a diverse progeny of applications ranging from MP3 file exchange to collaborative learning. This scalability, however, comes at a price. Since its commercial inception in 1995, the Internet has lagged behind the public switched telephone network (PSTN) in availability, reliability and quality of service (QoS). This relative lack of reliability stems in part from the absence of intermediate backbone state and synchronization between routers. Despite the remarkable tolerance demonstrated by today's end-users for failures and delays in email and web services, the relative lack of Internet backbone reliability poses a significant challenge for emerging transaction-oriented and interactive applications like Internet telephony, online business and collaboratories.

Although recent advances in the IETF's Differentiated Services working group promise to improve the performance of application-level services within some networks, across the wide-area Internet these QoS algorithms are usually predicated on the existence of a stable underlying forwarding infrastructure. In recent work, we showed that the Internet lacks effective inter-domain path fail-over [1]. Specifically, we found that multi-homed Internet sites may experience periods of degraded performance as well as complete loss of connectivity persisting fifteen minutes or more after a single fault.

We showed that most of the latency in Internet fail-over stems from *delayed convergence*, or the temporary routing table oscillations formed during the operation of the path selection process on Internet backbone routers after a fault. Unlike switches in the public telephony network which exhibit failover on the order of milliseconds, our analysis found that inter-domain routers in the packet switched Internet may take several minutes to reach a consistent view of the network topology after a fault.

The current Internet inter-domain routing protocol, BGP, evolved from earlier distance vector routing algorithms. These protocols, including RIP [2], suffer from a number of well-documented problems, including slow convergence times [3]. Distance vector routing requires that each node maintain the distance from itself to each possible destination and the vector, or neighbor, to use to reach that destination. Whenever this connectivity information changes, the router transmits its new distance vector to each of its neighbors, allowing each to recalculate its routing table. The count-to-infinity problem [2] provides the canonical example used to illustrate the slow convergence in distance vector routing.

The adoption of the path vector in BGP is widely and incorrectly believed to have "solved" the routing table oscillation problems exhibited by RIP. Instead, we showed in [1] that the adoption of the path vector exponentially exacerbates the number of potential routing table oscillations. Specifically, we found that a default configuration (i.e. one without additional administratively added policies or filters) of n BGP autonomous systems connected in a complete graph may potentially explore $n!$ routes, or all possible paths of all possible lengths between each AS after a fault. This upper theoretic bound on BGP convergence compares poorly with earlier routing protocols, such as RIP which have been shown to have $O(n^3)$ computational complexity [4].

We based our earlier analysis on a simplified, abstract model of BGP interconnectivity. This model neglected the impact of routing policies, more realistic timing assumptions and inter-AS connectivity on the process of delayed convergence. Although our initial model provides a useful theoretic upper bound on BGP distributed computation, we note that this bound is unlikely to occur in practice. In this work, we expand on our earlier effort by exploring the measured convergence behaviors of "real" topologies, including more than 20 unique BGP route advertisements between more than 200 pairs of Internet service providers (ISPs). We also provide analysis of BGP behavior in general network topologies and under other more realistic assumptions. Our major results include:

- The time complexity for Internet fail-over convergence is $30 * \Theta(n)$ seconds, where n is the length of the longest backup path between the source and any destination autonomous system for a route.
- On average, routes from customers of larger ISPs exhibit faster convergence than routes announced by customers of smaller Internet providers.
- Errant paths are frequently explored during delayed convergence. These "vagabond" paths likely stem from misconfiguration or software bugs.
- Most Internet routes exhibit multiple backup paths which transit several times the number of Internet providers as steady state paths.

The remainder of this paper is organized as follows: In Section 0.2, we provide some background and related work. Section 0.3 discusses our experimental data collection infrastructure. In Section 0.4, we present survey results on ISPs policy mechanisms and discuss the impact of these policies on the flow of routing information. In Section 0.5, we present both empirical observations as well as quantitative analysis of the relationship between specific Internet topological configurations and the rate of convergence. We demonstrate a relationship between the convergence delay of a route announced between two providers and the longest path allowed by the topology and policy between both providers. In Section 0.6, we present formal proof of this relationship. Finally, we conclude in Section 0.7 with a discussion of modifications to BGP which, if deployed, would significantly improve inter-domain routing convergence.

0.2 Background

In this Section, we provide a brief review of the more salient aspects of BGP inter-domain routing related to the discussion in this paper. We assume that the reader is familiar with Internet routing concepts and terminology discussed in [5, 6, 1].

As a path vector protocol, BGP updates include an *ASPath*, or a sequence of intermediate autonomous systems between source and destination routers that form the directed path for the route. BGP uses the ASPath for both loop detection and policy decisions. Upon receipt of a BGP update, each router evaluates the path vector and invalidates any route which includes the router's own AS number in the path.

Although not specified in the BGP standard, most vendor implementations ultimately default to best path selection based on ASPath length. The number of ASes in the path is used in a manner similar to the metric count attribute in the RIP. While BGP allows for path selection based on policy attributes, including local preference and multi-exit descriminator values, the majority of ISP policies ultimately default to the selection of the route with the shortest path. In the remainder of this paper, we base our analysis on such constrained shortest path first policies.

The BGP standard also includes a minimum route advertisement interval timer, abbreviated in this paper as MinRouteAdver, which specifies a minimum amount of time that must elapse between advertisements of routes for a particular destination from a given BGP peer. This timer provides both a rate-limiter on BGP updates as well as a window in which BGP updates with common attributes may be bundled into a single update for greater protocol efficiency. The standard recommends thirty seconds as the MinRouteAdver interval plus/minus some additional random jitter.

A number of recent studies, including Varadhan et al. [7] and Griffin and Wilfong [8] have explored BGP routing *divergence*. BGP allows the administrator of an autonomous system to specify arbitrarily complex policies. In BGP divergence, Griffin and Wilfong show that it is possible for autonomous systems to implement "unsafe," or mutually unsatisfiable policies, which will result in persistent route oscillations. In [9], Gao et al. prove that adherence to specific common ISP policies, including provider and customer relationships, will guarantee convergence.

The authors of all the above papers note that BGP divergence remains a theoretical finding and has not been observed in practice. Our work explores a complimentary facet of BGP routing – the convergence behavior of inter-domain routers under the default BGP path selection policies. In this paper and [1], we show that even under constrained policies, the BGP Internet routing exhibits an order of magnitude longer convergence latencies than previously believed.

An increasing number of Internet customers today choose to *multi-home*, or provision external connectivity through multiple ISPs. This provider redundancy is designed to secure against single link, router or even ISP failures. In [1], we showed that the convergence delay associated with route failure is equivalent to the delay of multi-homed failover. In the remainder of this paper, we focus our analysis on T_{down} convergence for clarity of presentation.

A number of studies, including [10, 11] have explored the inter-domain topology and diameter of the Internet. These studies typically build topological maps based on periodic snapshots of network routing tables or active traceroutes. Our work focuses on a complimentary aspect of Internet topology – the set of all *possible* paths between source and destination autonomous systems. Although in steady-state, Internet routers will normally select the shortest path to a given destination, we show in Section 0.6 that BGP routers will explore all available longer paths to the destination following a failure.

0.3 Methodology

Our study builds on the experimental infrastructure originally developed in [1]. Our measurement and fault injection apparatus consists of Unix-based probe machines maintaining geographically and topologically diverse BGP peering sessions with more than 20 ISPs. While in [1], we observed the impact of faults injected into only two Internet providers, in this work we expand our instrumentation to inject BGP route transitions (i.e. announcements and withdraws) into more than 10 geographically and topologically diverse providers.

Software from the MRT and IPMA projects [12, 13] running on both FreeBSD PCs and Sun Microsystems workstations was used to generate BGP routing update messages at random intervals of roughly a two-hour periodicity. The faults simulated route failures and repairs. In [1], we showed that the convergence behavior of route failures is equivalent to multi-homed failover.

We generated faults over a six month period to provide statistical guarantees that our analysis was based on deliberately injected faults rather than normally occurring exogenous Internet failures, which the authors in [14] found occur on the average of once a month. Each cooperating provider agreed to both accept our fault-injection announcements and treat the address space as a customer address block with respect to policy and filtering. As we only injected routing information for addresses assigned to our research effort, these faults did not impact routing for commodity ISP traffic with the exception of the addition of some minimal level of extra routing control traffic.

While one set of probe machines actively injected faults, we observed the impact of these faults through passive instrumentation of an additional twenty ISP default-free routing tables. Again using software from the MRT and IPMA projects, we logged both periodic routing table snapshots and all BGP routing updates received by our "Routeviews" probe machines from the 20 peers to disk. We then correlated the data between our NTP synchronized fault injection and measurement probe machines. These correlations provided data on the conver-

Transit relationships -- Inbound	
Prefix filters to receive customer routes only	100%
Peer relationships -- Outbound	
Communities	73%
Prefix filters and ASPaths	13%
Prefix filters only	13%
Peer relationships -- Inbound	
Bogon Filtering Only	80%
No Filtering	20%

Figure 1: Survey results of Internet provider filtering mechanisms.

gence delays between multiple source and destination ISP peers. We inferred the steady-state and convergence topologies between all probed ISP pairs using the ASPath information included in BGP update messages advertised to the passive Routeviews probe machine.

In addition to our experimental measurements, we surveyed a broad spectrum of Internet backbone providers about the details of their routing and peering policies. Responses from 15 backbone providers of varied network size and topologies provide the framework in which we discuss the impact of specific filtering and policy implementation mechanisms on the process of delayed convergence.

At the request of the providers participating in our study, we anonymize the AS numbers, IP addresses and names of all providers in our examples and accompanying discussions. We use the anonymized Internet provider names and AS numbers consistently throughout the paper.

0.4 Policy

In this Section, we present the results of our survey on backbone provider routing policy mechanisms. We first provide a taxonomy of Internet hierarchical peering relationships and then illustrate the role these policies and specific implementation mechanisms play in the flow of routing information.

The Internet retains a significant physical interconnection hierarchy with several "tiers" of service providers. We illustrate this hierarchy with an example topology in Figure 2. In [9], Gao et al. describe the provider use of filters to implement several types of commercial peering relationships. Based on this discussion and the results of our survey of Internet provider policies, we describe three categories of inter-provider relationships. Unlike the taxonomy used by Gao et al, we define these relationships based on both import and export policies. From our discussions with ISPs, these classifications closely reflect what providers themselves use to describe the relationships.

Peer – Service providers bilaterally exchange their customer and backbone routing information. These peers do not exchange routes learned from other peer or upstream providers.

Customer/Transit – A customer announces its backbone and downstream customer routes to an upstream provider. The customer receives default-free routing information from the upstream provider. The upstream provides *transit* service to the customer.

Backup transit – A type of peer relationship in which a provider only provides transit after detection of a fault. In steady-state, both providers are peers. After a failure, the backup transit peer begins advertising its now downstream peer's backbone and customer routes.

Providers typically enter into multiple combinations of the above commercial and engineering relationships with other ISPs. For example, a tier-2 provider may have *peer* relationships with other tier-2 providers while also purchasing *transit* from a tier-1 provider. We will show later in this Section that both the type of peering relationship as well as the implementation details of that relationship impact the number of possible paths for a route between two providers.

Figure 1 presents the results of our survey of Internet provider route filtering mechanisms. Each row lists the percentage of surveyed ISPs that implement a specific filtering mechanism on their border routers. We separate filtering mechanisms into three broad categories based on the type of inter-provider relationship. For each grouping, an ISP typically will implement only one mechanism in that set. We illustrate the impact of these policy mechanisms through several examples using the topology shown in Figure 2.

We first see in Figure 1 that all surveyed providers use prefix filters to limit acceptance of customer announcements to only "legitimate" address space assigned to that customer. For example, ISP D filters the peering session with ISP G to only accept ISP G's backbone and customer routes. Following the hierarchy upwards in Figure 2, ISP A similarly filters the peering with D to only accept backbone and customer routes from ISP D. Since ISP G provides transit to ISP D, ISP A also accepts ISP G's routes from ISP D.

The outbound peer relationship category in Figure 1 shows that 73 percent of surveyed ISPs advertise routes to peers based on community attributes. Communities provide a mechanism for providers to tag, or color, routes. The providers then use these tags to distinguish between which routes are advertised to customers and peers. For example, ISP A in Figure 2 may use communities to tag routes received from ISP D as customer routes. Based on this tag, ISP A would then announce these routes to peers ISP B and ISP C.

We also see that 26 percent of ISPs control their outbound advertisements to peers through some combination of prefix and ASPath filters. Providers use lists of these prefixes and/or ASPath regular expressions to explicitly permit the routes which should be advertised to peers. Thirteen percent implement both ASPath and prefix filter mechanisms and another 13 percent only prefix filter.

For example, in Figure 2 we assume ISP A implements both prefix and ASPath filters on its outbound route announcements to peers. For routes learned from customer D, ISP A's filters ensure that both A's advertisements to peers match D's address space and that all advertised routes come directly from ISP D (i.e. ISP D is the first AS in the ASPath). Based on these filters, we observe that ISP A may advertise routes with the paths "D G" and "D", but not "C D G".

The combination of ASPath and prefix filters prevents the unintentional creation of back-up transit paths. If ISP A only implements prefix filters, then after a failure between ISP A and ISP D, ISP A might learn ISP D's routes from ISP C with an ASPath of "C D" and "C D G". Lacking ASPath filters, ISP A will advertise these "C D G" routes to peer ISP B and, thus, provide transit to ISP B for ISP C.

As another example, we consider a tier-2 provider, ISP D, multi-homed to two upstream providers, ISP A and ISP C. ISP D also maintains peer relationships with tier-2 providers, ISP E and ISP F, and provides transit to ISP G. We list the set of possible paths for routes originating in ISP D below the example. We observe that both ISP C and A do not reannounce any of the routes learned from peers (non-transit/customer routes) to other peers. Thus, the paths "D A C B" and "D C A B" are not valid. However, both the "D A C B" and "D C A B" would be valid after a failure if ISP A and ISP C provide back-up transit for each other.

In general, our survey data shows that smaller tier providers tend to possess a higher degree of peer and transit intereconnectivity than larger providers. For example, by definition tier-1 providers do not purchase transit or generally maintain backup transit relationships with other providers. We will show in the next Section that these difference between tier-1 and tier-2 policies often lead to more numerous and longer backup paths for customers of tier-2 ISPs.

Finally, Figure 1 illustrates an important aspect of current filtering practices – the relative lack of filters applied to routes received from peers. Eighty percent of providers filter only "bogon" peer advertisements, or prefixes which represent private address space, default, unallocated address space, etc. Due to the technical and contractual difficulties of maintaining filter lists for the large number of routes advertised by peers, most providers resort to trusting their peers to send only valid information. We note that in a few well-publicized incidents, this trust has proven catastrophic for Internet routing [15].

0.5 Experimental Results

In this section, we present both empirical observations and analysis of the data collected by our fault injection experiments. We first provide several examples illustrating the process of delayed convergence. In the second subsection, we explore the impact of specific topological factors on convergence latencies.

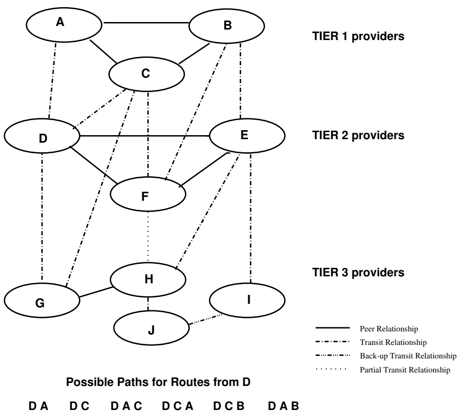


Figure 2: Example illustrating impact of provider policies on the propagation of routing information.

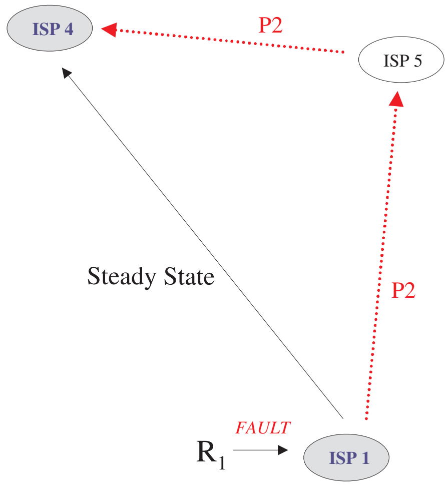
0.5.1 Convergence Topologies

During the six months of our study, we analyzed the routing topologies between more than 200 pairs of Internet providers. We graph only three representative topologies in Figure 3 for clarity. We note that all of the other monitored topologies in our study exhibited related behaviors.

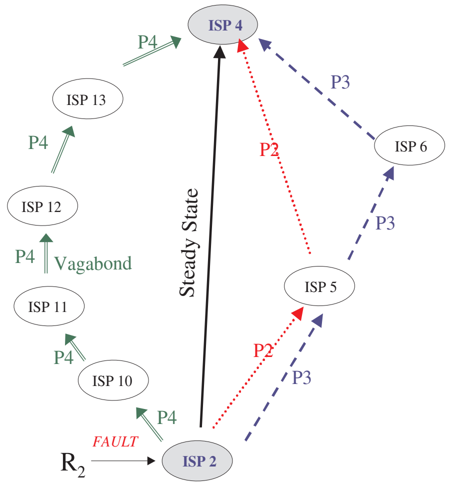
The three subfigures in Figure 3 show a subset of the primary and backup paths announced to our Routeviews machine by a single Japanese Internet provider, ISP4, after we withdrew routes $R1$, $R2$, and $R3$ from our Mae-West exchange point BGP peering sessions with providers IS1, ISP2 and ISP3. For convenience, we will refer to these three ISPs at Mae-West as our *immediate providers*. The arrows represent the flow of routing information as inferred from the BGP ASPath update information announced by ISP4.

In each diagram, we label the *steady-state path*, or the path normally selected by ISP4 in the absence of a fault. The steady-state paths include IS1-ISP4 in Figure 3(a), ISP2-ISP4 in (b) and ISP3-ISP4 in (c). Similarly, we label backup paths chosen by ISP4 in each diagram with the letter P followed by integers denoting the frequency with which we observed that backup path (i.e. P1 to P6 in Figure 3(c)). For clarity, we graph only the most common backup paths observed during our study. In addition to the paths illustrated, ISP4 announced an additional 11 unique paths for $R2$ and 7 additional paths for $R3$ after 23 and 27 percent of the faults, respectively. We note that ISP4 only announced a single backup path throughout the course of our study for the topology in Figure 3(a).

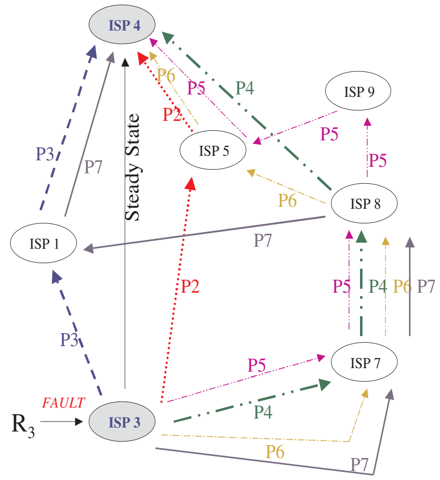
In steady state, we first observe from Figure 3 that ISP4 maintains a direct BGP peering session with all three of our immediate providers. Active ICMP ping and traceroute measurements show these three steady-state paths exhibit similar loss and latency characteristics. Although the steady-state paths are



(a) ISP1



(b) ISP2



(c) ISP3

Figure 3: Subset of backup paths explored during the process of delayed convergence for routes from a Japanese provider to three different ISP routers at the California Mae-West exchange point.

Delayed Convergence of R1	Delayed Convergence of R2	Delayed Convergence of R3
96% Average: 92 (min/max 63/340) seconds Announce AS4 AS5 AS1 (44 seconds) Withdraw (92 seconds) 4% Average: 32 (min/max 27/38) seconds Withdraw (32 seconds)	63% Average: 79 (min/max 44/208) seconds AS4 AS5 AS2 (35 seconds) Withdraw (79 seconds) 7% Average: 88 (min/max 80/94) seconds Announce AS4 AS5 AS2 (33 seconds) Announce AS4 AS6 AS5 AS2 (61 seconds) Withdraw (88 seconds) 7% Average: 54 (min/max 29/9) seconds Withdraw (54 seconds) 23% **Other**	36% Average: 110 (min/max 78/135) seconds Announce AS4 AS5 AS3 (52 seconds) Withdraw (110 seconds) 35% Average: 107 (min/max 91/133) seconds Announce AS4 AS1 AS3 (39 seconds) Announce AS4 AS5 AS3 (68 seconds) Withdraw (107 seconds) 2% Average:140.00 (min/max 120/142) Announce AS3 AS8 AS7 AS3 (27 seconds) Announce AS3 AS5 AS8 AS7 AS3 (86 seconds) Withdraw (140 seconds) 27% **Other**

Figure 4: Ordered list of the most common ASPath sets announced by ISP4 during the process of delayed convergence following the withdrawal of routes $R1$, $R2$ and $R3$ from three Internet providers at the Mae-West exchange point.

similar, we observe significant variation in both convergence latencies and the topologies explored by ISP4 for each of the three routes following the injection of a fault.

Figure 4(a) provides a numerical breakdown of the alternate path data graphically represented in Figure 3. The first column shows the distribution of paths announced by ISP4 after $R1$ is withdrawn from ISP1. The next two columns show similar results for routes $R2$ and $R3$ withdrawn from ISP2 and ISP3, respectively. Each entry provides a partial list of ASPath sets ordered by their measured frequency, and an associated average, minimum and maximum convergence delay. For example, the first entry for ISP1 indicates that after 96 percent of $R1$ failures, ISP4 announces a backup path "AS4 ASA5 AS1" (in an average of 44 seconds) followed by a withdrawal an average of 92 seconds after the fault. This backup path announcement corresponds to the primary backup path, P2, labeled in Figure 3(a). The second entry for $R1$ in Figure 4 denotes that after four percent of failures, ISP4 withdrew $R1$ without an intervening announcement of any backup path. We will provide probable explanations for these behaviors later in this Section.

In Figure 3(b) and the second column of Figure 4, we provide a slightly more complex example of the process of delayed convergence. After 63 percent of $R2$ failures, we again see that ISP4 first fails-over to the primary backup path, P2, followed by a withdrawal. After less than 7 percent of faults, however, we observe that ISP4 fails-over to backup path P2 followed by P3 before a final withdraw. For another 7 percent of faults, ISP4 immediately withdraws the route. Finally, after the remaining 23 percent of faults, we observe more than 45 sequences of "other" ASPath set announcements, each with a frequency of well less than one percent. These other ASPath sequences include an additional 11 unique paths as well as inter-mixed withdrawals from ISP4. Analysis of this "other" category and discussions with Internet providers suggests that the majority of these rare backup routes represent transient paths due to router misconfiguration errors.

Throughout the six months of our study, we observed frequent examples of these misconfigured, or *vagabond* paths between the majority of the 200 pairs of Internet providers we monitored. We define a vagabond path as a backup route which persists for a brief, fixed period of time (usually only several days) and does not conform to any intended or published policies. For example, over the course of two weeks we observed the backup path P3 in Figure 3(b) after less than five percent of faults for $R2$. We note that P3 both begins and ends with two ISP routers co-located at the Mae-West exchange point. Between these two co-located routers, however, P3 traverses an additional four ISPs and transits one small Mediterranean country. ISP2 responded to our inquiries about this unusual path and pointed the blame at a single border router access-list configuration error which was subsequently resolved. We were able to partially automate detection of this and other vagabond paths as many of these erroneous routes transited the same misconfigured router, or Internet provider. This example of a vagabond path emphasizes an important aspect of Internet routing – the disproportionate impact a single ISP misconfiguration error may have on global Internet routing. As we described in Section 0.4, most large ISPs only filter customers and do not check the validity of announcements from peers.

Finally, we observe a yet more complex example of delayed convergence in Figure 3(c). In this diagram, we illustrate ISP4's exploration of six backup paths ranging between lengths of 3 and 5 after a fault for route $R3$ from ISP3. After the majority of faults, ISP4 fails-over to path P2 followed by a withdraw. At a slightly lower frequency, ISP4 fails over first to P3, then P2, and finally withdraws the route. During convergence after the remaining 29 percent of faults, we measured ISP4's exploration of an additional 145 ASPath set combinations which include an additional 14 unique ASPaths. Analysis of these remaining paths and discussions with providers indicate that approximately 65 percent represent vagabond paths, while 35 percent are "legitimate" backup paths. We next provide probable explanations for these different fail-over behaviors.

Analysis of our data shows that both the probability of selection and order of backup paths chosen during delayed convergence depends primarily on the interaction of the MinRouteAdver timer on routers along each path. As described in [1], the most widely deployed commercial router software today implements MinRouteAdver on a per peer basis. As a result, the initial MinRouteAdver timer value applied to the propagation of a new route failure is dependent on earlier routing instability propagated across each peering session. For example, for the majority of failures in Figure 3(a), the withdrawal for $R1$ propagates along the steady-state path *before* ISP4 learns the backup P2 is also invalid. After four percent of faults, however, ISP1's initial MinRouteAdver associated with ISP4 delays the withdrawal to ISP4 longer than the time required for a withdrawal to propagate along path P2. In this latter case, ISP4 will first invalidate the backup path P2. As the primary steady-state path still exists, ISP4 will not send out any update until it finally receives the ISP1 withdrawal and invalidates the primary path after an average of 32 seconds. This latter sequence of events is much less probable than an initial fail-over to P2 as it requires ISP1's MinRouteAdver timer to ISP4 to be longer than the combined MinRouteAdver timers of ISP1 to ISP5, and ISP5 to ISP4.

In general, we expect initial routing information to propagate more slowly via longer paths than shorter as each router along the path adds between 0 and 30 seconds of MinRouteAdver delay. We observe in Figure 3 that backup path selection also may depend on the interaction, or *MinRouteAdver interference*, of multiple paths in a topology. For example, Figure 3(c) shows that ISP5 maintains a steady-state active path through ISP3 and secondary paths through both its neighbors, ISP8 and ISP9. After a fault for $R3$, ISP5 on average will first receive a withdrawal from ISP3. In this scenario, ISP5 will then fail-over deterministically to one of its backup paths, announce the new active path and re-initialize its MinRouteAdver timer to each peer. We observe that this initial fail-over and reset of ISP5's MinRouteAdver timers will delay the propagation of subsequent fail-over announcements. Specifically, after receipt of a withdrawal from ISP9 or ISP8, ISP5's MinRouteAdver timer will suppress the announcement of fail-over to either P4 or P5 for an additional 30 seconds. We note that the MinRouteAdver timer value between each peer initially follows a non-uniform probability distribution between 0 and 30 seconds. After propagation of the initial update, however, all subsequent MinRouteAdver timer values will delay updates for at *least* 30 seconds.

In most instances, the interaction of MinRouteAdver timers provides significant benefit in maintaining scalability of the distributed operation of the BGP protocol. As we observed in [1], MinRouteAdver timers add a level of synchronization to the system and thereby limits the number of BGP update messages and computational states. In earlier simulations, we found that the synchronization due to MinRouteAdver reduces computational complexity of BGP convergence from $O(n!)$ to $O(n)$. Although MinRouteAdver reduces the computation complexity, the timers also introduce significant additional latency during delayed convergence. In the next Subsection, we explore the dominant relationships between convergence latency and network topology.

0.5.2 Topology Impact on Convergence

In the previous Subsection, we showed that both the order and selection of backup paths explored during delayed convergence depends on a number of factors, including the initial setting of the MinRouteAdver timers between peers and, to a lesser degree, link and router processing delay. In Figure 5, we present a cumulative distribution graph of the convergence latencies for routes in the three topologies shown in Figure 3. The horizontal axis represents the number

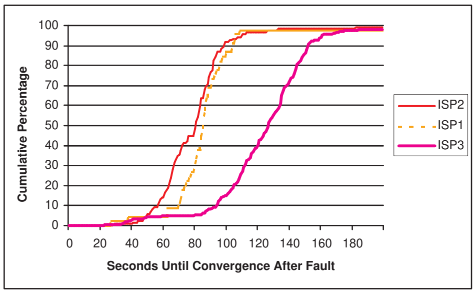


Figure 5: Convergence latency of route from ISP1, ISP2 and ISP3 after a cumulative percentage of faults.

of seconds from injection of the fault until each ISPs' BGP routing tables reach steady state for that prefix; the vertical axis shows the cumulative percentage of all such events. For clarity we limit the horizontal axis to 190 seconds. All ISPs exhibit a long-tailed distribution of convergence latencies extending up to fifteen minutes for a small, but tangible percentage of events. Analysis of these long-tailed events finds that most correspond to the exploration of vagabond paths.

In Figure 5, we first observe that the distribution of convergence latencies of ISP2 and ISP1 appear similar, while ISP3 exhibits significantly slower convergence times. We note that 80 percent of ISP1 and ISP2 failures converged in 100 seconds, while only 20 percent converged from ISP3 in the same time period. If we neglect the impact of vagabond paths, at a high level these convergence latencies correspond with the relative complexity of the secondary topologies explored in Figure 3. Specifically, we observe that in Figure 3(a) ISP4 explored a single backup path, P1, of length two. In Figure 3(b), ISP4 explored backup paths P1 and P2 of lengths 2 and 3, respectively. In contrast, ISP4 explored significantly longer and more complex topologies, including paths of length 5, during the delayed convergence of $R3$. This relatively higher degree of topological complexity corresponds to the longer convergence latencies shown for ISP3 in Figure 5.

Analysis of the convergence latencies for more than 200 (source, destination) inter-domain paths shows that the average convergence latency for a route failure corresponds to the length of the longest possible backup path allowed by policy and topology between two providers. We quantitatively illustrate this relationship between path length and convergence delay with a scatter plot in Figure 6. The vertical axis provides the average convergence delay for each (source, destination) pair observed during our study. The horizontal axis provides the longest, non-vagabond ASPath length we observed announced during the process of delayed convergence. We include a trend line in the graph to better illustrate data inter-relationships.

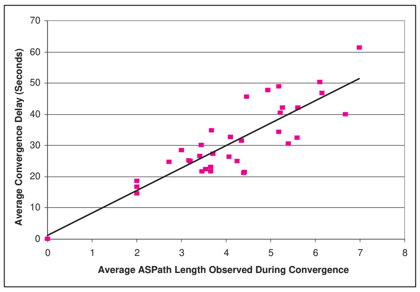


Figure 6: Scatter plot of the average convergence latency and longest ASPath length explored during the process of delayed convergence between more than 200 pairs of Internet providers.

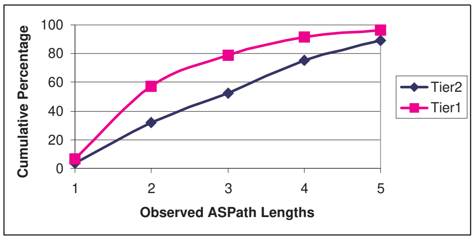


Figure 7: Cumulative percentage distribution plot of ASPath lengths seen by customers of tier-1 and tier-2 ISPs during convergence.

Although the data in Figure 6 contains significant variability, we observe a linear relationship between the longest ASPath length for a route between two ISPs and the average failure convergence delay for that route. We present a formal proof of this relationship in the next Section. A probable explanation for the variability in Figure 6 and differences between convergence latencies for ISP2 and ISP1 in Figure 5 involve differences in initial the MinRouteAdver settings due to previous routing instability, intra-domain routing protocol convergence, and the processing and link delays on intermediate backbone routers.

We now explore the relationship between backup path length and Internet provider policies. Analysis of the more than 200 paths observed during our study shows a relationship between the tier size of an ISP and both the convergence delay and the length of ASPaths exhibited by routes from customers of that ISP. For example, in Figure 3 ISP1 represents one of the largest tier-1 Internet backbone providers; ISP2 represents a moderate sized US-based tier-2 provider; and ISP3 represents a regional, tier-3 network.

We illustrate this relationship between the tier of a provider and convergence behaviors in Figure 7. This graph shows the cumulative percentage of backup ASPath lengths observed during the six months of our study for routes originating from customers of two categories of providers: tier-1 and tier-2. We group providers into tier-1 and tier-2 categories based on published network topologies, published peer policies and provider self-designation. In Figure 7, we observe upstream routers on average explore shorter ASPath for routes originating in tier-1 than tier two providers. While the longest non-vagabond path explored during convergence for tier-1 customer routes is 9 ASes, tier-2 customer routes grew as long as 12 ASes. As we discussed earlier, a probable explanation for these differences in the backup paths from different tiers of providers is that smaller ISPs typically purchase transit from multiple upstream providers. These upstream transit providers may also, in turn, purchase connectivity from additional tier-1 and tier-2 providers. Smaller ISPs also implement policies unnecessary in larger providers, such as backup transit.

0.6 Proof

In this Section, we present a formal proof of the relationship between the longest backup inter-domain path for a route between two Internet providers and the convergence latency of that route. We first present a model of network topology and BGP communication.

0.6.1 Model of BGP Convergence

We model the Internet as a directed graph G, with a set V of nodes and a set E of ordered pairs (u, v) which are also called arcs, where $u, v \in V$. The nodes V represent the set of autonomous systems in the Internet. We fix a destination X. The arcs E are connections; an arc $e = (u, v)$ exists if and only if node u will inform node v (i.e. send an update) about its best route to destination X (but not vice versa).

Definition 0.6.1 We define the *out-neighbors* $N^-(u) = \{v \in V | \exists e = (u,v) \in E\}$, and similarly the *in-neighbors* $N^+(v) = \{u \in V | \exists e = (u,v) \in E\}$.

As discussed in Section 0.2, the BGP protocol decision process selects the best path to a destination X based on ASPath length (minimum number of hops). A node u informs all nodes in $N^-(u)$ about its best path to destination X. In addition to its primary path to destination X, a node u stores also alternative paths; at most one per neighbor in $N^+(u)$.

If the best path of a node u to destination X changes, node u will announce its new best path to all nodes in $N^-(u)$ (it will send an update message with complete path information, e.g. "$u \to a \to b \to c \to X$"); if node u loses a connection to destination node X, u will send a withdrawal message "withdraw(X)". Our model also implements other BGP behaviors, including loop detection and implicit withdrawal of routes.

We are given a node X (a client) that is not connected to the rest of the network, except a single arc to node A (X's AS). We let this single connection go down, thus destroying all possible paths from any node in the internet to X. We are interested in the time it takes until the network is stable again (every node knows that there is no path to X). Let's call this the T_{down} *convergence time*. In addition, we are also interested in the time it takes from establishment of a connection from X to A until the BGP routing tables at all nodes are filled accordingly. We call this the T_{up} *convergence time*.

Due to the operation of the MinRouteAdver timer, a node u does not send two update messages over a link e within time MinRouteAdver. If the last update message was sent more than time MinRouteAdver ago, node u sends an update message over a link e after waiting time t_{wait}. The value t_{wait} is between 0 and MinRouteAdver. As noted in Section 0.2, the initial MinRouteAdver value depends on the previous routing traffic. Sending a message over an arc e costs time t_{send}.

We will see that the T_{up} convergence time is significantly smaller than the T_{down} convergence time. Informally speaking, T_{up} convergence latency is related to the shortest path, while T_{down} convergence is related to the longest path in a network.

0.6.2 T_{up} Convergence

We first prove the relationship between T_{up} convergence latency and the shortest path between a source and destination node for a route.

Definition 0.6.2 The *distance* from node u to node v (denoted as $d(u,v)$) is the minimum number of hops from u to v. More formally, $d(u,u) = 0$, and $d(u,v) = \min\{d(u,w)|w \in N^+(v)\} + 1$. Let $d^{max}(u)$ be the *maximum distance* from u to any node, that is $d^{max}(u) = \max\{d(u,v)|v \in V\}$.

We connect X to A at time 0.

Theorem 0.6.3 *A node v with distance d from node A learns its best route to X at time $d(t_{wait} + t_{send})$, where t_{wait} and t_{send} respectively are the average experienced waiting time and transmitting time on the nodes and arcs along the best route.*

Proof: By simple induction. Node A learns the shortest route to X at time 0. Let u be an in-neighbor of node v with minimal distance, that is $d(A,v) = d(A,u) + 1$. From the induction hypothesis we know that node u received the update with its best route to node X at time $d(A,u)(t_{wait} + t_{send})$. Node u waits for time t_{wait} and sends an update to node v, which is received t_{send} time later, thus at time $d(A,u)(t_{wait} + t_{send}) + t_{wait} + t_{send} = d(A,v)(t_{wait} + t_{send})$. ■

Corollary 0.6.4 *The T_{up} convergence time is at least $d^{max}(A)(t_{wait} + t_{send})$, and at most $(d^{max}(A) + 1)(t_{wait} + t_{send})$, where t_{wait} and t_{send} respectively are the average experienced waiting time and transmitting time on the nodes and arcs along the best routes.*

Proof: The lower bound is established in Theorem 0.6.3. The upper bound follows because, in the BGP protocol, nodes also store alternative paths; a node with maximum distance d^{max} will therefore forward its best path at time $d^{max}(t_{wait} + t_{send}) + t_{wait}$, and they will be received at its out-neighbors at $(d^{max} + 1)(t_{wait} + t_{send})$ time. ■

In order to simplify and strengthen our results we will make several assumptions about the above parameters. In practice, the transmission delay of an update message is significantly less than the average MinRouteAdver delay. The probability that a route to *any* destination has changed in the last 30 seconds is high. Thus, the waiting time t_{wait} is often in the same order of magnitude as MinRouteAdver, or $\Theta(w)$ for short. To shorten and brace our results we therefore assume that

$$\Theta(1) = t_{send} \ll t_{wait} = \Theta(\text{MinRouteAdver}) = \Theta(w).$$

Corollary 0.6.5 *The short form of the T_{up} convergence time is $\Theta(dw)$, where d is the length of the maximum shortest path in the network.*

The T_{up} convergence time is better when paths with less hops deliver the update messages faster than paths with more hops. If a message on a longer path is delivered slightly before a message on a shortest (best) path, then the update for the best path might experience MinRouteAdver interference before it can be forwarded to the out-neighbors.

In order to optimize the T_{up} convergence time we need to both minimize MinRouteAdver delay as well as reduce the diameter (all shortest paths) of a network.

0.6.3 T_{down} Convergence Time

We now show that the T_{down} convergence time for a network is dependent on the longest path in that topology.

Definition 0.6.6 A *simple path* p is an ordered sequence of k nodes $(u_1, \ldots, u_k)$ such that for any pair i, j with $i \neq j$ we have $u_i \neq u_j$, and for any i ($i = 1, \ldots, k-1$) there is an arc e such that $e = (u_i, u_{i+1}) \in E$.

Note that a simple path between two nodes can be considerably longer than the distance (shortest path).

We disconnect X and A at time 0, and henceforth assume that $N^-(X) = \{\}$.

Starting with time 0, various update messages are sent along the arcs. We specify a *message* m with a record of three elements: the arc $e(m)$ it was sent over, the time $s(m)$ it was sent, and the time $r(m)$ it was received, that is $m = (e(m), s(m), r(m))$. Note that for all m we have $s(m) \geq 0$.

Definition 0.6.7 Let the set M be the set of all update messages. We say that a simple path $p = (u_1, \ldots, u_{k+1})$ from node u_1 to node u_{k+1} is *covered at time* T if there is a subset $M_k = \{m_1, \ldots, m_k\}$ of messages in M such that:

- $e(m_i) = (u_i, u_{i+1})$, for $i = 1, \ldots, k$.
- $r(m_i) \leq s(m_{i+1})$, for $i = 1, \ldots, k-1$.
- $r(m_k) \leq T$.

Lemma 0.6.8 *Node u has a route to X at time T if and only if there is a simple path from X to u that is not covered at time T.*

Proof: First we show "$\Rightarrow$": At time T node u has a best route $u = u_k \to u_{k-1} \to ... \to u_1 \to u_0 = A \to X$ (node u_k forwards a message for destination X to node u_{k-1} which in turn forwards the message to node u_{k-2}, and so on). The according simple path for u's best route is $p = (u_0, \ldots, u_k)$. Assume for the sake of contradiction that the subset $M_k = \{m_1, \ldots, m_k\}$ covers p at time T. For all $m' \in M$, let m_k be the update messages such that $e(m_k) = e(m') = (u_{k-1}, u)$ and $s(m') \leq s(m_k)$, and $r(m_k) \leq T$. In other words, m_k is the last update message sent by node u_{k-1} to node u and received by node u at time $r(m_k) \leq T$. If there is no update message m_k we are already done because there cannot be a cover M_k – we have a contradiction. Since at time T, node u has route p, the message m_k was announcing this route (another update message m' node would have another route). This update message m_k was sent by u_{k-1} at time $s(m_k)$, therefore at time $s(m_k)$ the best route for u_{k-1} was $u_{k-1} \to \ldots \to u_1 \to u_0 = A \to X$. By induction, we conclude that at time $s(m_1)$ the best route for $u_0 = A$ was $A \to X$. This is a contradiction since at time $s(m_1) \geq 0$ ($s(m) \geq 0$ for all $s \in M$), node A already knew that the link to X is down.

Now we show "$\Leftarrow$": Specifically we show that if node u has no routes to X at time T, then all simple paths from X to u are covered at time T. Assume for the sake of contradiction that there is a simple path $p = (A = u_0, \ldots, u_k = u)$ that is not covered at time T. Since node u has no routes to X at time T, node u has received a message $m_k = \text{withdraw}(X)$ from all its in-neighbors (specifically node u_{k-1}) on time $r(m_k) \leq T$. Given $s(m) \leq r(m)$ we know that node u_{k-1} had no route at time $s(m)$. Therefore it has received a withdrawal message m_{k-1} from u_{k-2} at time $r(m_{k-1}) \leq s(m_k)$. We inductively construct a subset of messages $M_k = \{m_1, \ldots, m_k\}$ which obeys the rules of a covering for path p at time T. ∎

The Lemma 0.6.8 puts us in a very comfortable position to argue about the T_{down} convergence time. We only have to worry about the minimum time it takes to cover all the simple paths from A to a node u. For a given simple path $p = (u_0, \ldots, u_k)$ of length k we know (in analogy to Theorem 0.6.3) that covering simple path p takes time $k(t_{wait}+t_{send})$. Therefore we get immediately:

Definition 0.6.9 Let $D^{max}(u,v)$ be the *length of the longest simple path* from node u to node v, that is $D^{max}(u,v) = \max_{\forall p} \text{length}(p)$, where p is a path from node u to node v.

Theorem 0.6.10 *The T_{down} convergence time is $D^{max}(A,u)(t_{wait} + t_{send})$, where t_{wait} and t_{send} respectively are the average experienced waiting time and transmitting time on the nodes and arcs along the longest simple path.*

As for Corollary 0.6.5, we use a short form for T_{down} of the same type as used for T_{up}.

Corollary 0.6.11 *The short form of the T_{down} convergence time is $\Theta(Dw)$, where D is the length of the longest path in the network.*

Computing the longest simple path for a given graph is known to be NP-complete. Even more problematic, we note that there is provably no good heuristic to approximate the longest simple path unless P = NP (the longest path problem is in APX [16]).

In order to optimize T_{down} convergence time, we need to minimize the longest paths in a network. However, we encounter a trade-off between the length of possible paths and the degree of connectivity in a network. Specifically, in a highly connected network, the longest path is linear in the number of nodes of the network. We will elaborate on this trade-off in future work.

0.7 Conclusions and Future Work

In this paper and previous work, we have shown that the Internet currently lacks the level of reliability and fault-tolerance required for the successful deployment of many emerging mission-critical network services. We argue that current Internet convergence latencies of up to fifteen minutes after a single multi-homed fault will prove untenable for new interactive and transaction-oriented network applications like Internet telephony.

This paper demonstrated that the time complexity for multi-homed Internet path fail-over scales linearly with the length of the longest possible backup path for that route. We showed that the length of inter-domain backup paths depends on a number of inter-provider contractual and policy implementation details. Our results show that customers sensitive to fail-over latency should multi-home to larger providers, and that smaller providers should limit their number of transit and backup transit interconnections. Finally, the large number of erroneous vagabond paths we observed during our study suggests a significant need for better route validation and authentication mechanisms.

In ongoing work, we are exploring possible solutions to the problems of delayed convergence and path authentication. Solutions under exploration include the use of adaptive MinRouteAdver timers and the association of additional information with BGP withdrawal messages. Our hope is to identify mechanisms which ameliorate the delayed convergence problem while preserving the scalability and flexibility of the Internet routing protocols.

Bibliography

[1] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian, "Delayed Internet Routing Convergence," *Proc. of the ACM SIGCOMM*, Aug. 2000.

[2] R. Perlman, *Interconnections, Second Edition*, Addison-Wesley, Reading Massachusetts, 1999.

[3] J. Garcia-Luna-Aceves, "Loop-free Routing Using Diffusing Computations," *IEEE/ACM Transactions on Networking*, Feb. 1993.

[4] K. Bhargavan, D. Obradovic, and C. Gunter, "Formal Verification of Distance Vector Routing Protocols," *International Conference on Theorem Proving and Higher Order Logics*, Aug. 2000.

[5] Y. Rekhter and T. Li, "A Border Gateway Protocol 4 (BGP4)," Sept. 1999, draft-ietf-idr-bgp4-09.txt.

[6] B. Halabi, *Internet Routing Architecture*, Cisco Press, 1997.

[7] Kannan Varadhan, Ramesh Govindan, and Deborah Estrin, "Persistent Route Oscillations in Inter-Domain Routing," Tech. Rep. USC CS TR 96-631, Department of Computer Science, University of Southern California, Feb. 1996.

[8] T. Griffin and G. Wilfong, "An Analysis of BGP Convergence Properties," *Proceedings of the ACM SIGCOMM*, Aug. 1999.

[9] Lixin Gao and J. Rexford, "Stable Internet Routing Without Global Coordination," *Proc. of ACM SIGMETRICS*, June 2000.

[10] Bill Cheswick and Hal Burch, "Internet Mapping Project," http://www.cs.bell-labs.com/who/ches/map/.

[11] Ramesh Govindan and Hongsuda Tangmunarunkit, "Heuristics for Internet Map Discovery," *Proc. of IEEE INFOCOM*, 2000.

[12] "Internet Peformance Measurement and Analysis Project (IPMA)," http://www.merit.edu/ipma.

[13] "Multithreaded Routing Toolkit (MRT) Project," http://www.mrtd.net.

[14] C. Labovitz, A. Ahuja, and F. Jahanian, "Experimental Study of Internet Stability and Wide-area Network Failures," *Proc. International Symposium on Fault-Tolerant Computing*, June 1999.

[15] R. Barrett, S. Haar, and R. Whitestone, "Routing snafu causes internet outage," *Interactive Week*, 1997.

[16] David R. Karger, Rajeev Motwani, and G. D. S. Ramkumar, "On approximating the longest path in a graph," *Algorithmica*, vol. 18, no. 1, pp. 82–98, May 1997.