FlowRoute: Inferring Forwarding Table Updates Using Passive Flow-level Measurements

Amogh Dhamdhere
CAIDA
University of California, San Diego
La Jolla, CA
amogh@caida.org

Lee Breslau, Nick Duffield, Cheng Ee,
Alexandre Gerber, Carsten Lund,
Subhabrata Sen
AT&T Labs-Research
Florham Park, NJ
{breslau, duffield, ctee, gerber, lund,
sen}@research.att.com

ABSTRACT

The reconvergence of routing protocols in response to changes in network topology can impact application performance. While improvements in protocol specification and implementation have significantly reduced reconvergence times, increasingly performance-sensitive applications continue to raise the bar for these protocols. As such, monitoring the performance of routing protocols remains a critical activity for network operators. We design FlowRoute, a tool based on passive data plane measurements that we use in conjunction with control plane monitors for offline debugging and analysis of forwarding table dynamics. We discuss practical constraints that affect FlowRoute, and show how they can be addressed in real deployment scenarios. As an application of FlowRoute, we study forwarding table updates by backbone routers at a tier-1 ISP. We detect interesting behavior such as delayed forwarding table updates and routing loops due to buggy routers – confirmed by network operators – that are not detectable using traditional control plane monitors.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations, Network Management

General Terms

Design, Measurement, Performance

Keywords

Measurement, Netflow, Routing Update

1. INTRODUCTION

The behavior of networks during and after routing changes – when packets may be subject to looping, loss, and delay variation – is an important determinant of the performance that end users perceive. Historically, inter-domain routing convergence could take minutes [12], while intra-domain protocols (e.g., OSPF) could take tens of seconds. Subsequent improvements in protocol specification and implementation reduced the reconvergence times of these protocols. However, the recent rise of applications with stringent performance requirements means that routing reconvergence is still a subject of intense interest among service providers.

To study routing behavior, network operators often use specialized control plane monitors [19–21] that peer with and receive routing updates from one or more routers. Control plane monitors can indicate when a router received a routing update, i.e., *when the router should have updated its forwarding table, and what it should have updated to.* They cannot detect, however, *when the router actually implemented the change in its forwarding table, and what change it made.* The actual change may be delayed, or may differ from what is expected due to hardware or protocol implementation bugs in routers. Further, monitoring the control plane at one or a few routers may not indicate how close in time different routers update their forwarding tables. If these times diverge, we could see transient forwarding loops and poor end-to-end performance. One could study forwarding table updates using logs collected directly from routers. In today's networks, however, we cannot instrument routers to dump their forwarding tables after every update. Even if this were feasible, it would require transporting and processing gigabytes of data after every update. SNMP polling would also have to deal with the same data volume, and under current operational practices, would give us a 5-minute resolution for inferring forwarding table changes – too coarse for our needs. One could use active probing (e.g., traceroutes) to monitor forwarding table updates. Active probing faces two drawbacks, however; provisioning monitors to achieve high spatial coverage can be costly, and the temporal resolution is limited by probing rates, which we cannot increase indefinitely without causing excessive overhead.

We describe the design and implementation of FlowRoute, a tool that works *in conjunction with existing control plane monitors* to analyze forwarding table dynamics. FlowRoute can measure forwarding table update times across routers and help operators identify slow forwarding table updates, transient forwarding loops, and large traffic shifts. FlowRoute works offline, using *passive flow-level measurements* (e.g., Cisco's Netflow [4]) that operators routinely collect in to-

IMC'10, November 1–3, 2010, Melbourne, Australia.
Copyright 2010 ACM 978-1-4503-0057-5/10/11 ...$10.00.

day's networks; as such, it does not impose additional overhead on routers. FlowRoute is agnostic to the specific routing protocols in use and can be applied to both intra-domain and inter-domain routing.

As an application of FlowRoute, we study forwarding table updates at backbone routers in a large tier-1 ISP. We find unexpected cases wherein forwarding table updates at a single router are delayed following a routing event. Delayed updates, in turn, cause forwarding loops lasting tens of seconds in some cases. Using FlowRoute, we attributed these effects to specific routers that were often late in updating their forwarding tables. Network operators confirmed that those routers did indeed have protocol implementation bugs causing performance issues.[1] Such unexpected behavior is not detectable using traditional control plane monitors.

We believe that operators can apply FlowRoute to a broad set of other debugging/analysis problems. For example, by measuring forwarding behavior across routers, FlowRoute can help network operators examine the extent to which forwarding table updates are synchronized across routers. FlowRoute can also quantify the network-wide effects of routing changes, such as traffic shifts and changes in link utilization after a routing change.

2. RELATED WORK

The OSPFmon [21] deployment at AT&T, the IP monitoring project at Sprint [10] and commercial products such as Route Explorer [16] provide route monitoring services to ISPs. However, none of these systems can study forwarding table performance at the timescales that FlowRoute is capable of. Feldmann et al. [9] describe an approach to periodically dump router configuration files in order to identify configuration errors. Their approach provides a static view of the routing state at each router. Such an approach would face significant scalability problems if extended to studying routing table dynamics.

In theory, one could use traceroute-like active measurement tools [3] to infer forwarding table changes. Active measurement provides a view of the routing state at routers on a path *at the time the probes are sent*. To achieve a sufficiently high temporal resolution for studying forwarding table updates, we would need a high probing rate and a large number of vantage points. Previous studies have used a combination of active probing, route monitoring and passive measurements to quantify the effects of routing events on end-to-end loss rates [23] and delays [18]. FlowRoute allows us to study forwarding table dynamics, which we can use to investigate the effects of routing events on end-to-end performance.

Operators use flow-level measurements from ISP networks for a variety of applications such as estimating intradomain traffic matrices [13] and flow size distributions [8]. Teixeira et al. [22] use NetFlow and routing data from a large Tier-1 ISP to quantify the effects of routing changes on the intradomain traffic matrix. Agarwal et al. use passive data to measure the effect of BGP route changes on the ingress-egress traffic matrix [1], and to study how traffic to neighboring ASes shifts due to changes in a local AS' IGP link metrics [2]. To the best of our knowledge, no previous work used flow-level data to study forwarding table dynamics.

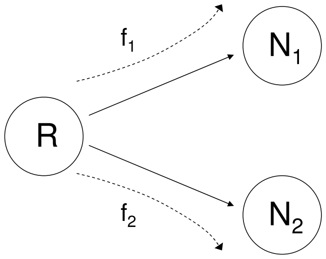

Figure 1: Basic model for routing change detection

3. FLOWROUTE DESIGN

We begin by describing, at a high level, the approach and algorithms embodied in FlowRoute. Figure 1 shows a single packet flow f_1 towards destination D. Suppose that router R forwards this flow towards next hop router N_1 at time t_1. The flow record at N_1 indicates that the previous hop was R, or, equivalently, that N_1 was R's next hop towards D at time t_1. If the next hop changes to N_2 at time $t_2 > t_1$, then a flow f_2 traversing R and destined to D will generate a corresponding flow record at N_2. By using these two flow records, we can infer when R started routing flows towards N_2. We next describe how we process the raw flow records from routers and infer routing changes. Note that FlowRoute processes flow records offline, on infrastructure that operators already use for network management, and does not impose additional overhead on routers.

3.1 Flow Records

We use flow records generated by Netflow [4], which is supported by multiple router vendors. Netflow records summarize flows, i.e., sets of packets which share common header fields (e.g., source and destination IP addresses). To infer routing changes, FlowRoute uses the following information that is present by default in Netflow records: the router R that observed the flow; the incoming and outgoing interfaces i and o at R; the times t_f and t_l of the first and last packets of the flow; and the destination D. We denote this flow record by the tuple (R, i, o, t_f, t_l, D).

While flow records report both incoming and outgoing interfaces, there is an important semantic difference between the two. The incoming interface is part of the flow key which defines a flow. A change in the incoming interface leads to the creation of a new flow record. Consequently, we know that each packet of a flow arrived on the input interface indicated by the flow tuple. In contrast, the outgoing interface o is not part of the flow key. Therefore, a change in the outgoing interface (next hop) while the flow is active will not generate a new flow record. Rather, the timestamp t_f in the flow record indicates the time when the flow record was created. Thus, we can only infer reliably that the reported outgoing interface was used for the first packet of the flow.

3.2 Inferring Routing Changes

We collect Netflow records and process them *offline* to obtain a stream of **Routing Flow Records** (RFRs). Each RFR is of the form (R_1, t_1, t_2, D, R_2), which states that during the interval $[t_1, t_2]$, router R_1 is forwarding packets to destination D via next hop R_2. In Figure 2, we describe how we construct two RFRs from each flow record

[1] These routers were subsequently retired from the network.

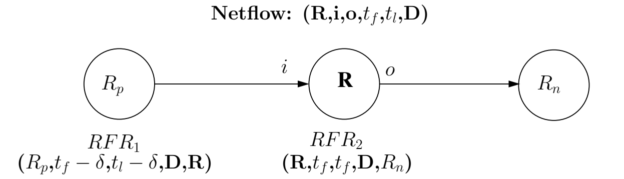
Netflow: $(\mathbf{R},\mathbf{i},\mathbf{o},t_f,t_l,\mathbf{D})$
R_p
i
$\mathbf{R}$
o
R_n
RFR_1
$(R_p,t_f-\delta,t_l-\delta,\mathbf{D},\mathbf{R})$
RFR_2
$(\mathbf{R},t_f,t_f,\mathbf{D},R_n)$

Figure 2: Construction of two Routing Flow Records from a flow record

(R, i, o, t_f, t_l, D) at router R. First, using the incoming interface i, we generate RFR_1=$(R_p, t_f - \delta, t_l - \delta, D, R)$, which indicates that during the interval $[t_f - \delta, t_l - \delta]$, router R_p used R as the next hop to reach destination D.[2] To obtain timestamps at R_p from the packet observation times t_f and t_l at R, we subtract the propagation delay δ of the link between R_p and R. We ignore queuing delays, as they cannot be estimated using topology and configuration information alone. Moreover, queuing delays are typically negligible at backbone routers [17]. We also ignore clock skew across routers, as routers use NTP [14] for clock synchronization. Next, using the outgoing interface o, we generate RFR_2=(R, t_f, t_f, D, R_n). Due to the semantic difference between incoming and outgoing interfaces, the two timestamps in RFR_2 are the same.

Consider again the routers R, N_1 and N_2 in Figure 1. Suppose we have RFRs $(R$,t_1,t_2,D,$N_1)$ and $(R$,t_3,t_4,D,$N_2)$ with $t_1 < t_4$. The first RFR indicates that R used N_1 as the next hop to D during the interval $[t_1, t_2]$, and the second RFR indicates that R used next hop N_2 during $[t_3, t_4]$. Two scenarios are possible: if $t_3 < t_2$, this may mean that Equal-Cost Multi-Path (ECMP) routing [15] is sending similarly-destined flows over two different downstream links (We address the issue of ECMP in the following section). If, on the other hand, $t_2 < t_3$, then R changed its forwarding path to D at some point in the range $[t_2, t_3]$. Our fundamental unit of route change measurement is thus a **Range** $(R, [t, t'], D)$, indicating that R changed its next hop towards destination D at some point during the interval $[t, t']$.

4. PRACTICAL CONSIDERATIONS

In this section, we discuss some issues that must be addressed before FlowRoute can be deployed in operational networks.

4.1 Destination Granularity

FlowRoute needs to observe consecutive flows to the same destination, D, to be able to infer forwarding changes towards D. While the destination IP address of a flow is an obvious choice, a router may not observe sufficient flow volume towards the same destination to obtain a sufficient temporal resolution for the time window of the routing change. Aggregating at the level of destination prefixes yields more flows to each destination but requires some additional information to map a flow to its associated prefix. A third option is to use the iBGP next hop field [6] in Netflow, in which case routers need to have complete iBGP routing tables.

Our motivation in developing FlowRoute was to understand the intra-AS routing dynamics of a large ISP. Such networks are often designed to have a *route free core*, meaning that routes consist of label-switched MPLS paths between ingress and egress routers. For routing within such a network, the address of the MPLS tunnel endpoint is the "destination" of a flow. In this work, we use FlowRoute to study intra-AS routing using MPLS tunnel endpoints as destinations. We emphasize, though, that *our method does not depend on the choice of MPLS tunnel endpoints as destinations.* An operator is free to choose the appropriate granularity of D.

4.2 Routing Change vs ECMP

We have assumed that consecutive RFRs with different next hops towards the same destination indicate a routing change. This may not hold when Equal-Cost Multi-Path (ECMP) [15] is enabled; a router may have two or more equal cost paths to the same destination and alternate between them on a per-flow basis. Consider a router R with two equal cost paths to destination D via next hops R_1 and R_2. Consecutive RFRs of the form $(R$,t_1,t_2,D,$R_1)$, $(R$,t_3,t_4,D,$R_2)$, and $(R$,t_5,t_6,D,$R_1)$ would appear as a change in R's next hop from R_1 to R_2 in the time interval $[t_2$,$t_3]$, and from R_2 to R_1 in time interval $[t_4$,$t_5]$. Next-hop changes due to ECMP *do not appear as routing changes to a control plane monitor.* Further, the mapping of a flow tuple to next hop routers can be dynamic, meaning that our offline analysis cannot leverage known properties of the mapping function to detect ECMP. We therefore design a method to distinguish between routing changes and ECMP using the sequence of RFRs.

We find empirically – using data from a day on which a production control plane monitor reported no routing events – that in more than 99% of cases, a router forwards fewer than 20 flows to a certain next hop, before routing a flow to a different equal cost next hop. Using this observation, we devise the following ECMP filtering algorithm. We keep track of the current next hop $n_c(R, D)$ for router R towards destination D, and the number of consecutive flows $f_c(R, D)$ that R has routed towards the current next hop $n_c(R, D)$. Let $n_h(R, D)$ be the next hop for router R towards destination D obtained from the latest RFR. If $n_h(R, D)$ is different from $n_c(R, D)$, and if $f_c(R, D) > t_{ecmp}$, then we count this as a routing change; otherwise we assume it is an instance of ECMP. For the analysis in this paper, we use t_{ecmp}=20.[3]

4.3 Traffic Rates and Sampling

FlowRoute produces a set of ranges, $(R, [t, t'], D)$, specifying the time window in which we infer a routing change to have occurred. The temporal resolution of our inference (the width of these ranges) depends on the frequency with which flows to destination D traverse router R, which itself depends on the popularity of D and the distance (in network hops) of R from D. Routers closer to destination D aggregate traffic from more sources on the sink tree towards D. Packet sampling in NetFlow also affects the temporal granularity of our inference. This effect is compounded by further sampling of flow records within the measurement collection infrastructure [7]. To quantify the effect of sampling, 1Gbps of backbone traffic would generate hundreds of Net-

[2] We can map the incoming and outgoing interfaces in Netflow records to previous and next hop routers, and obtain link propagation delays using slowly changing information available from SNMP and configuration data.

[3] We would need to estimate t_{ecmp} specifically for the network on which we deploy FlowRoute.

Flow records per second with 1 in 500 packet sampling [11], giving us a temporal resolution of a few milliseconds. 1Mbps of traffic would yield a temporal resolution of a few seconds, still sufficient for many applications. FlowRoute achieves high temporal resolution on routes that have the most traffic, which are exactly the routes that operators care most about monitoring. Also, FlowRoute is not limited to using flow records generated by existing traffic. We can augment existing traffic on low-traffic routes with active packet probes. A router equipped with Flexible NetFlow [5] can instantiate a dedicated packet filter to select all active probe packets based on a predetermined IP address/port signature. The resulting temporal resolution can be as small as the inter-probe time.

Though the temporal resolution of our inferences depends on the flow rate and sampling, we emphasize that these factors *do not affect the correctness of the ranges we infer*. For example, the range $(R, [t, t'], D)$ denotes that we last observed a flow at R towards the old next hop (say N_1) at time t, and first observed a flow at R towards the new next hop (say N_2) at time t'. Due to sampling, we could have missed a flow at R towards N_1 at time $t + \delta_1$, and a flow towards N_2 at time $t' - \delta_2$. A larger flow volume or less sampling would thus lead to narrower ranges. In the absence of short-lived route flaps, the bounds provided by our inferred range $(R, [t, t'], D)$ are correct – R was routing flows to N_1 *at least until* t, and R started routing flows to the new next hop N_2 *no later than* t'.

5. RESULTS

In this section, we use FlowRoute to study how routers update their forwarding tables in response to routing events reported by a control plane monitor. We first describe our datasets and verify the consistency of the route change information obtained across routers.

5.1 Approach and Data

Our study uses two types of data collected from a Tier-1 ISP network during July and August 2008. Our first dataset consists of routing events reported by OSPFMon[16] in the form of Link State Advertisements (LSAs) that indicate changes in link status (up/down) or link weights which could result in routing changes. OSPF events lend themselves to *clustering*, since events such as the cost-in or cost-out of links often result in multiple subsequent events relating to changes in path metrics. For each OSPF event cluster, a range $[t_s, t_e]$ indicates the start and end time of the cluster. We use a clustering threshold of 50 seconds, after which two events are assigned to separate clusters. Our second dataset consists of packet sampled Netflow records (which were also subject to flow-level sampling [7]) collected from several hundred backbone routers during the same two month period. We use FlowRoute to measure the times when routers updated their next hops in response to measured OSPF events (after filtering out occurrences of ECMP).

5.2 Validation of Inter-Router Consistency

Recall from Section 3 that each Netflow record generated at a router R creates two RFRs: one describing the routing state at R (towards its next hop) and one describing the routing state at the previous hop (towards R). We may observe consecutive RFRs at router R, or two RFRs at different routers downstream of R, indicating that R changed its next hop.[4] We validate the consistency of routing changes inferred from the two vantage points, providing assurance that our methodology is sound, and that NTP time synchronization across routers is sufficient for FlowRoute's purpose.

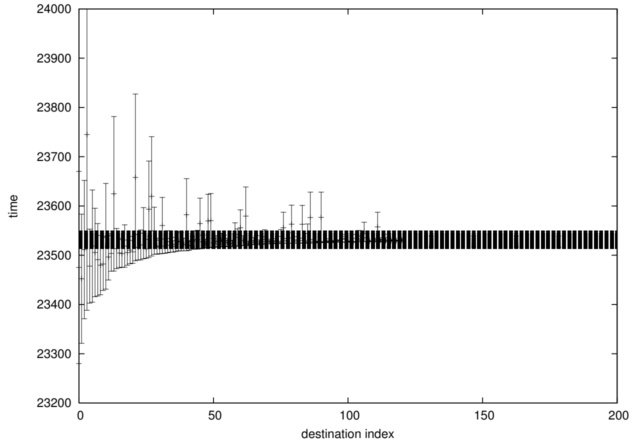

Figure 3: Router A: Updates of next hop to all destinations are consistent with OSPF events.

We separate the stream of RFRs into those obtained using input interfaces (I stream) and output interfaces (O stream). We apply the method of Section 3 to infer routing changes separately from the I and O streams. We compare cases where we infer a routing change for the same R, D pair using both I and O streams. For example, for a R, D pair, we infer the time windows $[t^i_1, t^i_2]$ and $[t^o_1, t^o_2]$ using the I stream and O stream, respectively. Using data from August 1, 2008, we find 2024 R, D pairs for which we inferred a routing change in both the I and O streams; *For each of these R, D pairs, the time windows are overlapping.* Thus, the I and O streams – which come from two independent sources – give consistent information about routing changes.

5.3 Delayed Forwarding Table Updates

We use FlowRoute to measure the times at which routers update forwarding tables in response to routing changes. Figure 3 shows the set of update ranges for a router A near the time of an OSPF event cluster on July 9, 2008. The OSPF event times are indicated by the horizontal band between 23,514 and 23,549 on the y-axis (denoting seconds since the start of the day.) The x-axis is the index of a destination D, and each vertical bar illustrates a range $(A, [t_1, t_2], D)$. The lower end of the bar (t_1) is the last time that we saw a flow routed towards to the old next hop, and the upper end of the bar (t_2) is the time we first saw a flow routed to the new next hop. We sort the destination indices in increasing order of t_1. For router A, the range for each destination overlaps the OSPF events; in this sense the FlowRoute view is consistent with the OPSFMon view. For some destinations, we obtain ranges that are narrower than one second, giving a fairly fine temporal resolution with which we can detect routing changes for those destinations.

Figure 4 shows the case of router B for an OSPF event

[4] While we may observe a routing change at both vantage points, often we only see it at one or the other location due to sampling.

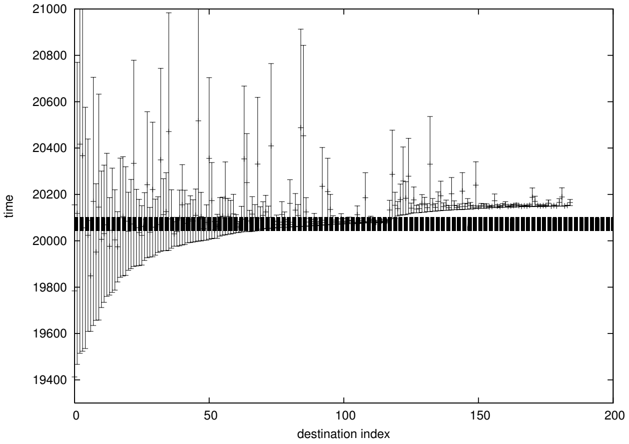

Figure 4: Router B: Updates of next hop to some destinations occur after last OSPF event.

cluster on July 4, 2008. The time range in which router B updated its route to some destinations (indices less than 100) overlaps the OSPF events; updates to these destinations could be consistent with the OSPF event cluster. For destinations with indices larger than 100, the *lower end of the range is up to 50 seconds after the last OSPF event*. The correctness of this observation is not affected by flow rates and sampling – the lower end of these ranges is *a lower bound* on when the router updated its forwarding table (see discussion in Section 4.3). We make two observations here. First, it appears that forwarding table updates for router B are spread out over time. Second, for destination indices greater than 100, router B was forwarding flows to the old next hop *at least until the lower end of the range* – more than 50 seconds after the OSPF event. This finding is significant because distributed routing protocols expect forwarding table updates to occur at about the same time across routers to ensure stability. Even a single router updating its forwarding table over an extended period (10s of seconds) could impact network-wide convergence. In Section 5.4, we describe cases where delayed forwarding table updates caused forwarding loops.

We measured the frequency of delayed forwarding table updates in the two month dataset from July and August 2008. Let $[t_1, t_2]$ be the window in which router R changed its next hop towards destination D in response to an OSPF event cluster from $[t_s, t_e]$. We say that a router R shows delayed forwarding table updates if $t_1 > t_e$ for at least N destinations. We use N=3 to avoid spurious conclusions due to a failure of our ECMP detection algorithm.[5] Across 2666 OSPF event clusters in the dataset, we found 97010 time ranges (one per R, D pair) that were consistent with the event cluster (i.e., like router A). We found 58 event clusters containing 117 time ranges where at least one router showed delayed forwarding table updates. Most routers showed this behavior only once over the two months; two routers, however, did so 14 times. We observed that most routers that showed delayed updates were of the same make/model. Discussions with network operators confirmed that those routers did indeed have protocol implementation bugs causing performance issues. The buggy routers have since been retired from the network.

5.4 Routing Loops

The delayed forwarding table updates described in the previous section can cause transient routing loops. We show how FlowRoute can detect such loops in practice. We use RFRs to identify routing loops using the following algorithm. Recall that an RFR (R_1, t_1, t_2, D, R_2) indicates that the next hop for router R_1 towards destination D was R_2 in the time interval $[t_1,t_2]$. To find routing loops, we search for RFRs of the form (R_1, t_1, t_2, D, R_2) and (R_2, t_3, t_4, D, R_1), where *the time windows* $[t_1, t_2]$ *and* $[t_3, t_4]$ *overlap*. This pair of RFRs indicates that a loop between adjacent routers R_1 and R_2 affected destination D during the intersection of time windows $[t_1, t_2]$ and $[t_3, t_4]$.[6] To study the duration of each such loop, we measure the number of consecutive seconds in which we saw overlapping RFRs for a pair of routers towards the same destination. We found 392 occurrences of one-hop loops during our two-month dataset. We found that more than 90% of these loops were short-lived, seen only in one or two consecutive seconds. We also found long-lived loops that lasted for 10s of seconds; the longest was seen in 67 consecutive seconds.

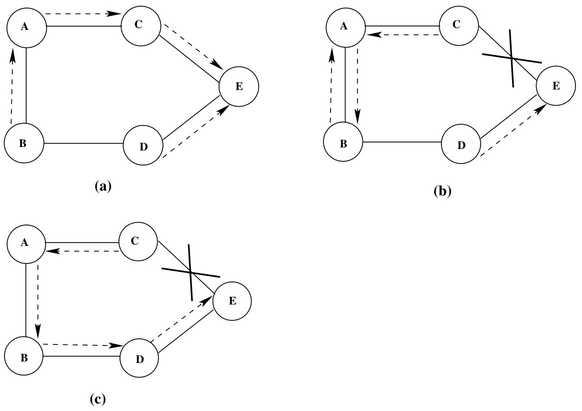

Figure 5: (a) Initial topology (b) A updates its next hop and forwards to B, but B is late to update, and continues to forward to A, resulting in a loop. (c) B updates its next hop and starts forwarding to D.

We next examine the relationship between the late forwarding table update phenomenon and transient routing loops. We found that of the 58 OSPF event clusters in which we observed late forwarding table updates, 33 were accompanied by transient one hop loops. Figure 5(a) shows the initial network topology for one such event. Figure 6 shows the route changes at routers A and B as they react to the OSPF event cluster which runs from approximately time 33280 to 33345 (seconds since the start of that day). This OSPF event cluster includes a cost-out event for link C-E, i.e., the link metric is increased such that it will not be part of any computed shortest path. The top graph in figure 6 shows that the forwarding table changes for router

[5] Different values of N yield qualitatively similar results.

[6] Our analysis focused on loops between adjacent routers; it can be easily extended to loops involving three or more routers.

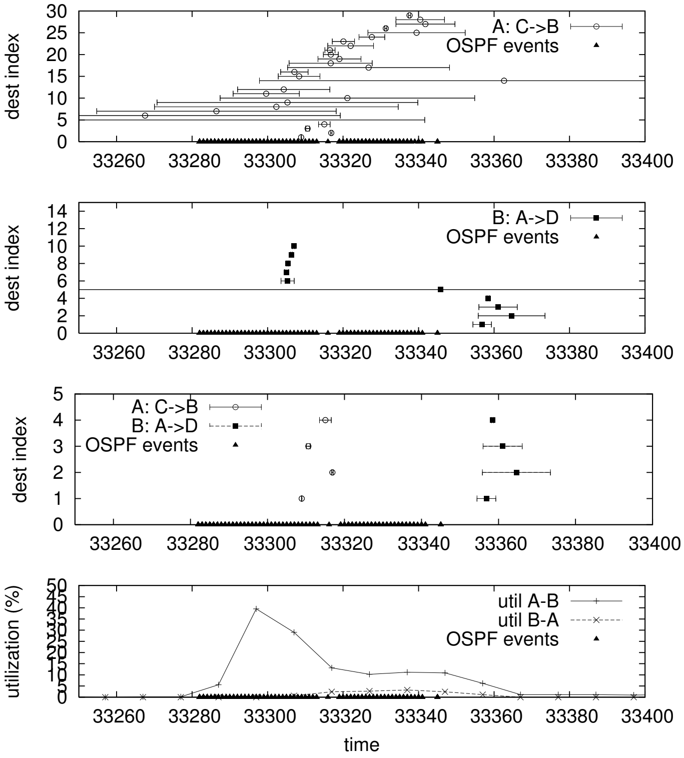

Figure 6: Updates by router A are consistent with OSPF events (top graph). Late updates by router B towards certain destinations (second graph). Update ranges for routers A and B for the same set of destinations (third graph). Spikes in the utilization of links A-B and B-A (bottom graph).

A are consistent with OSPF events (A changes its next hop from C to B). The second graph in Figure 6 shows that the forwarding table changes for router B towards certain destinations occur several seconds later (B changes its next hop from A to D). The third graph in Figure 6 shows the transition ranges for routers A and B towards the set of destinations for which B updated late. From time around 33320 to 33360, A is forwarding packets towards B (because it has already changed its next hop), and B is forwarding packets to A (because it has not yet changed its next hop). Figure 5(b) shows the situation from time around 33320 to 33360. Router A has updated its forwarding table to use B as its next hop. Router C has also updated its forwarding table to use A as its next hop. However, router B has not updated its forwarding table, causing a forwarding loop between A and B. Figure 5(c) shows the situation after time 33360, when router B has updated its forwarding table to use router D as its next hop; routing has stabilized.

The bottom graph in Figure 6 shows the utilization on links A-B and B-A, averaged over 10 second intervals during this event. Link utilization increases in both directions during the routing loop. Though route changes at other routers could impact utilization on these links (which could explain why the utilization in the direction A-B is higher than in the direction B-A), the fact that the spikes persist only during the loop suggest that they are primarily due to the routing loop.

6. DISCUSSION

We presented FlowRoute, a tool that uses passive flow-level measurements and works in conjunction with existing control plane monitors to study forwarding table dynamics. FlowRoute identified cases of delayed forwarding table updates and routing loops in a large tier-1 ISP network. These results are significant, as such anomalies are undetectable using control plane monitors alone. Before concluding, we mention ways in which the benefit of FlowRoute could be improved in the future.

6.1 Network-wide effects of routing changes

As FlowRoute uses data collected from hundreds of routers in the ISP's network, operators can use it to quantify the *network-wide* effects of routing changes. For example, we have used FlowRoute to measure how many routers in the network were "affected" by a routing change (in the sense that traffic traversing those routers was re-routed). Further, we have used FlowRoute to measure traffic shifts and changes in link utilization caused by routing events – at a finer time resolution than is possible with SNMP-based approaches. Due to space constraints, we could not present the results of measuring network-wide effects of routing changes.

6.2 Flexible Netflow

We expect that Flexible Netflow [5] will eventually make its way into production networks. Flexible Netflow allows operators to configure the fields that are part of the flow key. In particular, we could have the flow key consist of the input interface, output interface, and destination, where the destination refers to the iBGP next hop (see Section 4.1). Using this feature, the router can directly export a Routing Flow Record as defined in Section 3. These records, which would be generated directly by the router and which would include first and last packet timestamps, would allow us to get more detailed timing information about routing changes. Flexible Netflow can also help to overcome some of the problems of packet and flow sampling, which affect FlowRoute's temporal resolution. As mentioned in Section 4.3, we can augment existing traffic with additional active probes, and Flexible NetFlow can be configured to select all probe packets based on a pre-defined signature. We can then set the active probing rate to achieve a desired temporal resolution.

6.3 Online FlowRoute

FlowRoute, as described in this paper, uses an offline approach. The method relies on collecting Netflow records from different routers and creating Routing Flow Records using topology data collected by SNMP. Currently, FlowRoute requires on the order of a few hours to process a day's worth of Netflow data from several hundred routers. There is no reason *a priori* that this paradigm could not be extended to work in near real-time. Such a system would run a collector that aggregates and processes flow-level measurements from multiple routers and provides alerts about routing events. Tackling the systems and scalability issues involved in making FlowRoute near-online is a direction we plan to pursue in future work.

7. REFERENCES

[1] S. Agarwal, C.-N. Chuah, S. Bhattacharyya, and C. Diot. The Impact of BGP Dynamics on Intra-domain Traffic. *SIGMETRICS Performance Evaluation Review*, 32(1):319–330, 2004.

[2] S. Agarwal, A. Nucci, and S. Bhattacharyya. Measuring the Shared Fate of IGP Engineering and Interdomain Traffic. In *Proc. of the 13TH IEEE International Conference on Network Protocols (ICNP)*, pages 236–245, Sept. 2005.

[3] B. Augustin, X. Cuvellier, B. Orgogozo, F. Viger, T. Friedman, M. Latapy, C. Magnien, and R. Teixeira. Avoiding Traceroute Anomalies with Paris Traceroute. In *Proc. ACM SIGCOMM Internet Measurement Conference (IMC)*, Oct. 2006.

[4] Cisco. Cisco IOS Netflow. `http://www.cisco.com/en/US/products/ps6601/products_ios_protocol_group_home.html`.

[5] Cisco. IOS Flexible NetFlow Technology. `http://www.cisco.com/en/US/prod/collateral/iosswrel/ps6537/ps6555/ps6601/ps6965/prod_white_paper0900aecd804be1cc.html`.

[6] Cisco. NetFlow BGP Next Hop Support. `http://www.cisco.com/en/US/docs/ios/12_3/feature/guide/nfbgpnxt.html`.

[7] N. Duffield and C. Lund. Predicting Resource Usage and Estimation Accuracy in an IP Flow Measurement Collection Infrastructure. In *Proc. Internet Measurement Conference (IMC)*, Oct. 2003.

[8] N. Duffield, C. Lund, and M. Thorup. Estimating Flow Distributions from Sampled Flow Statistics. *IEEE/ACM Transactions on Networking*, 13(5):933–946, 2005.

[9] A. Feldmann and J. Rexford. IP Network Configuration for Intradomain Traffic Engineering. *IEEE Network Magazine*, 15:46–57, 2001.

[10] C. Fraleigh, C. Diot, B. Lyles, S. Moon, P. Owezarski, K. Papagiannaki, and F. Tobagi. Design and Deployment of a Passive Monitoring Infrastructure. In *Passive and Active Measurement (PAM) Workshop*, Amsterdam, Apr. 2001.

[11] Y. Gu, L. Breslau, N. Duffield, and S. Sen. On Passive One-Way Loss Measurements Using Sampled Flow Statistics. In *Proc. IEEE Infocom mini-conference*, Jul. 2009.

[12] C. Labovitz, A. Ahuja, A. Bose, and F. Jahanian. Delayed Internet Routing Convergence. *IEEE/ACM Transactions on Networking*, 9(3):293–306, 2001.

[13] A. Medina, N. Taft, K. Salamatian, S. Bhattacharyya, and C. Diot. Traffic Matrix Estimation: Existing Techniques and New Directions. *SIGCOMM Computer Communications Review*, 32(4):161–174, 2002.

[14] D. Mills. Network Time Protocol (Version 3) Specification, Implementation and Analysis. RFC 1305, 1992.

[15] J. Moy. OSPF Version 2. RFC 2328, April 1998.

[16] Packet Design. Route Explorer. `http://www.packetdesign.com/products/rex.htm`.

[17] K. Papagiannaki, S. Moon, C. Fraleigh, P.Thiran, and C. Diot. Measurement and Analysis of Single-Hop Delay on an IP Backbone Network. In *In IEEE Journal on Selected Areas in Communications, Special Issue on Internet and WWW Measurement, Mapping, and Modeling*, volume 21, 2003.

[18] H. Pucha, Y. Zhang, , Z. M. Mao, and Y. C. Hu. Understanding Network Delay Changes Caused by Routing Events. *SIGMETRICS Performance Evaluation Review*, 35(1):73–84, 2007.

[19] RIPE. RIPE Network Coordination Centre. `http://www.ripe.net`.

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

[21] A. Shaikh and A. Greenberg. OSPF Monitoring: Architecture, Design and Deployment Experience. In *Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, Mar. 2004.

[22] R. Teixeira, N. Duffield, J. Rexford, and M. Roughan. Traffic Matrix Reloaded: Impact of Routing Changes. In *in Proc. Passive and Active Measurement (PAM) Workshop*, Mar. 2005.

[23] F. Wang, Z. M. Mao, J. Wang, L. Gao, and R. Bush. A Measurement Study on the Impact of Routing Events on End-to-end Internet Path Performance. *SIGCOMM Computer Communications Review (CCR)*, 36(4):375–386, 2006.