Efficient Remapping of Internet Routing Events

Elverton Fazzion[1], Ítalo Cunha[1], Dorgival Guedes[1], Wagner Meira Jr.[1]
Renata Teixeira[2], Darryl Veitch[3], Christophe Diot[4]

[1]UFMG [2]Inria [3]University of Technology Sydney [4]Safran

ABSTRACT

Routing events impact multiple paths in the Internet, but current active topology mapping techniques monitor paths independently. Detecting a routing event on one Internet path does not trigger any measurements on other possibly-impacted paths. This approach leads to outdated and inconsistent routing information. We characterize routing events in the Internet and investigate probing strategies to efficiently identify paths impacted by a routing event. Our results indicate that targeted probing can help us quickly remap routing events and maintain more up-to-date and consistent topology maps.

CCS Concepts

•Networks → Network measurement; Network monitoring;

Keywords

Topology mapping, traceroute, routing events.

1. INTRODUCTION

Path changes are caused by *routing events* such as router reconfiguration, link failures, software errors, and scheduled maintenance. Routing events impact multiple paths in the Internet. Current monitoring techniques monitor paths independently: detecting a routing event on one Internet path does not trigger any action on other possibly-impacted paths. This approach leads to (i) outdated routing information, as we delay remapping other paths that have changed due to the routing event, and (ii) prevents us from observing the extent of a routing event, as another routing event might happen before we remap all paths impacted by the first one.

We investigate how to use partial information about a detected routing event to efficiently identify which paths it impacted and to quickly remap changes. Whenever a measurement (e.g., traceroute) detects and remaps a change on the path to a destination d, we remeasure other paths that *intersect* (traverse) the hops on the path to d impacted by the change. We characterize intersecting paths, and their changes, to guide the design of new probing strategies to efficiently detect changes.

We find that targeted probing allow for efficient detection of paths impacted by a routing event. In particular, after detecting and remapping a change on the path to destination d, we can often identify whether an intersecting path to destination d' has also changed using a single measurement probe. Also, other paths that have identical intersections with impacted hops on the path to d often share fate, i.e., paths with identical intersections either all change or none change. If we send a probe to identify whether an intersecting path changed, we can apply the same result to other paths with identical intersections. Finally, the probability that an intersecting path changes increases with the size of the intersection. When operating on a budget, we should probe paths by decreasing intersection size.

SIGCOMM '16, August 22 - 26, 2016, Florianopolis , Brazil

ISBN 978-1-4503-4193-6/16/08…$15.00

DOI: http://dx.doi.org/10.1145/2934872.2959052

2. DEFINITIONS AND DATASET

We define a local change zone, denoted LCZ, comparing two consecutive measurements of a path p. An LCZ is a sequence of contiguous hops removed from the previous measurement plus the immediately surrounding divergence (h_d) and convergence hops (h_c) present on both previous and current measurements. Each LCZ is computed minimizing the edit distance between consecutive measurements and comparing the set of interfaces at each hop [1]. The example in Fig. 1 shows two consecutive measurements of the path from source s do destination d (dotted and solid lines). The two measurements remap a path change, with an LCZ containing hops $\{h_1, h_2, h_3, h_5\}$, $h_d = h_1$, and $h_c = h_5$.

We used our topology mapping tool DTRACK [2] to collect a dataset to study routing events. Upon detecting a change on the path to a monitored destination d, DTRACK immediately queues the path to be remapped. We extended DTRACK to collect additional data in order to study strategies for identifying which paths have changed due to a detected routing event. After the remapping of the path to d is complete, the extended DTRACK computes LCZ and proceeds to remap other paths that could be impacted by the routing event. More precisely, DTRACK enqueues all (other) *inter-*

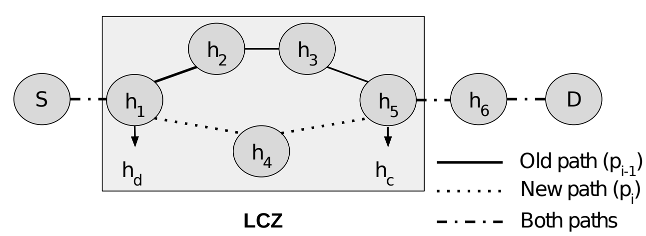

Figure 1: Example path change and definitions.

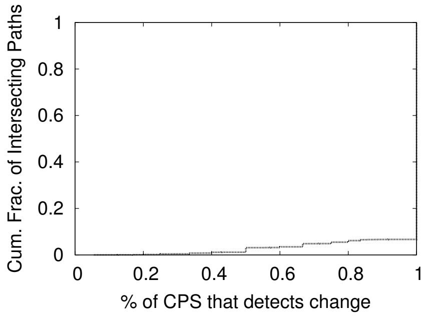

Figure 2: Fraction of hops in CPS(q) that detects change.

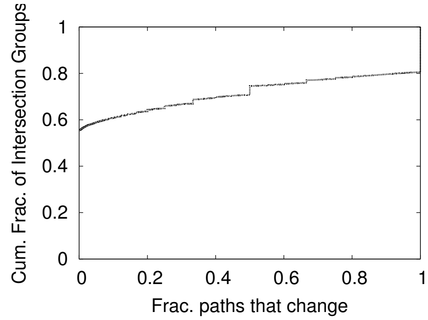

Figure 3: Fraction of paths that change in intersection groups.

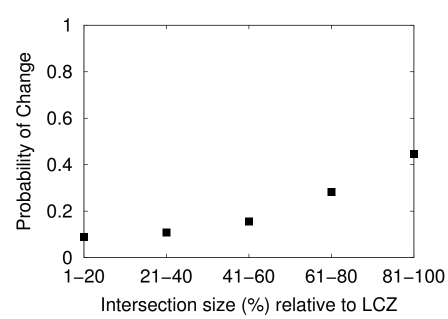

Figure 4: Probability of change as a function of intersection size.

secting paths that traverse hops in LCZ for remapping (if not already queued). Although this approach does not guarantee DTRACK remaps all paths impacted by the routing event, e.g., impacted paths that are not monitored or that do not overlap LCZ, it still provides a more accurate snapshot of a routing event than previous datasets. Note that DTRACK, and its extension described here, works with branched routes resulting from flow based load balancing, which is common in the internet. This complicates path/route mapping, tracking, and intersection estimation. All the paths and intersections described here are to be understood in this sense.

We deployed the extended DTRACK on 79 PlanetLab nodes. We generated a list of 12763 destinations including Alexa's TOP100 Websites (resolved from different PlanetLab nodes around the world), RIPE Atlas probes chosen at random, and random reachable /24 prefixes. The list of destinations covers 5715 ASes (computed using IP-to-AS mapping from iPlane). On each PlanetLab node, we selected 1000 random IP addresses to be monitored. The data used in this analysis was collected between Jan. 27th and Mar. 7th, 2016.

3. FINDINGS AND DISCUSSION

We now discuss our three main findings and possible guidelines for strategic probing when remapping routing events.

Detecting changes. We want to find an algorithm to identify whether an intersecting path has changed or remained stable using few probes. We consider sending probes to intersecting paths at specific hops and checking for changes [2]. For each intersecting path p, we compute a *candidate probing set*, CPS(p), containing all hops between the first (closest to the source) and last (furthest from the source) hops in the intersection between p and LCZ. We include all intermediate hops in CPS(p) to detect changes that impact hops that are not in LCZ, e.g., due to per-destination load balancing [1].

We find that probing the divergence hop h_d on the overlapping path rarely detects a path change. As a result, we do not consider probing h_d to detect a change and remove it from CPS(q). If h_d is not in CPS(q), then CPS(q) is unchanged. We also find that probing the convergence hop h_c on the overlapping path also rarely detects a change, *except* for routing events that change the convergence hops's distance from the source by adding or removing hops to the path. As a result, we only consider probing h_c for routing events that change the path length and remove it from CPS(q) otherwise. Fig. 2 shows, over all intersecting paths that changed (20% of the total), the distribution of the fraction of hops in CPS(q) that can detect the change. The distribution shows that, for 94% of the intersecting paths that changed, probing *any* hop in CPS(q) will detect the change.

Finding which intersecting paths have changed. The results above show we can identify, with high probability, whether an intersecting path has changed or not using a single probe. However, they only apply to the 20% of overlapping paths that do change. We now look at mechanisms to more efficiently find the 20% of routes that change. We find that most LCZs have multiple intersecting paths (5% of monitored paths on average). We group intersecting paths that have identical intersections with LCZ in an *intersection group*. Fig. 3 shows the distribution, over all intersection groups, of the fraction of intersecting paths that change in each intersection group. We note that 75% of all intersection groups with more than one intersecting path have shared fate, i.e., either all or none of their paths change. This result indicates that sending a probe to identify whether an intersecting path has changed allows us to adjust the expected utility of probing other paths in the same intersection group.

Maximizing probe utility. We now study which intersecting paths to probe first, i.e., paths that have a higher probability to change. Fig. 4 shows, over all routing events, the fraction of intersecting paths that changed. We group intersecting paths on the horizontal axis by the size of the intersection relative to the number of hops in LCZ. We note that the larger the intersection (i.e., when intersecting paths have more in common with the change), the higher the probability that the path will change. This suggests probing intersection groups by decreasing intersection size.

4. ACKNOWLEDGMENTS

This work was partially funded by NIC.BR, Fapemig, CNPq, Capes, InWeb, MasWeb and EuBra-BigSea.

5. REFERENCES

[1] B. Augustin, T. Friedman, and R. Teixeira. Measuring Multipath Routing in the Internet. *IEEE/ACM Trans. Netw.*, 19(3):830–840, 2011.

[2] I. Cunha, R. Teixeira, D. Veitch, and C. Diot. DTRACK: A System to Predict and Track Internet Path Changes. *IEEE/ACM Trans. Netw.*, 22(4):1025–1038, 2014.