On the Feasibility of Bandwidth Detouring

Thom Haddow[1], Sing Wang Ho[1], Jonathan Ledlie[2],
Cristian Lumezanu[3], Moez Draief[1], and Peter Pietzuch[1]

[1] Imperial College London, United Kingdom
[2] Nokia Research Center, Cambridge, MA, USA
[3] Georgia Institute of Technology, Atlanta, GA, USA

Abstract. Internet applications that route data over default Internet paths can often increase performance by sending their traffic over alternative "detour" paths. Previous work has shown that applications can use detour routing to improve end-to-end metrics such as latency and path availability. However, the potential of detour routing has yet to be applied where it may be most important: improving TCP throughput.

In this paper, we study the feasibility of bandwidth detouring on the Internet. We find that bandwidth detours are prevalent: between 152 Planetlab nodes, 74.8% of the paths can benefit from detours with at least 1 Mbps and 20% improvement. To understand how to exploit bandwidth detours in practice, we explore the trade-offs between network- and transport-level mechanisms for detouring. We show, both analytically and experimentally, that direct, TCP-based detour routing improves TCP throughput more than encapsulated, IP-based tunneling, although the latter provides a more natural interface.

1 Introduction

The Internet was designed for best-effort data communication. It is limited to a basic role—to provide connectivity—and does not guarantee good path performance between hosts in terms of latency, bandwidth or loss. Not surprisingly, direct end-to-end routing paths may be more congested, longer, or have lower bandwidth than necessary. To overcome these inefficiencies and improve network performance, distributed applications can use *detour routing* [17]. Detour routing constructs custom paths by concatenating multiple network-level routes using an overlay network.

Existing proposals use detour routing to improve latency [13] and availability [1,3]. However, an important potential benefit of detour routing—improving end-to-end bandwidth—is still unrealised. Bandwidth is critical for many Internet applications. For example, emerging data-intensive applications, such as HD video streaming and content-on-demand systems, require consistently high bandwidth in order to operate effectively. Further, as enterprises begin to store their data in "cloud" data centres, access to high throughput paths is critical.

Discovering and exploiting bandwidth detours is challenging. Unlike latency or path availability, bandwidth is more expensive to measure. Bandwidth measurement tools generally require many probes of differing sizes sent over long periods

N. Spring and G. Riley (Eds.): PAM 2011, LNCS 6579, pp. 81–91, 2011.

of time [5,18]. Available bandwidth also varies with the volume of cross-traffic on the path: measurements must be done not just once, but continuously.

In this paper, we study the feasibility of bandwidth detouring and lay the groundwork for a general Internet detouring platform for bandwidth. We explore the variability of bandwidth measurements and the properties of detour paths. Our measurements on the PlanetLab testbed show that 74.8% of the paths can benefit from at least 20% and 1 Mbps bandwidth increase. Bandwidth detours are often symmetric, benefiting both forward and reverse paths at the same time, and last for more than 90 minutes.

To understand how to build a bandwidth detouring platform, we investigate the trade-off between network- and transport-level mechanisms for detour routing and the relationship between detours for different path metrics. We provide evidence, both analytically and experimentally, that TCP-based detouring, rather than IP detouring, achieves better performance. In addition, we show that employing cheaper latency probes to find bandwidth detours is not effective.

The rest of the paper is organised as follows. In §2 we review related work. We consider Internet bandwidth measurement and analyse properties of bandwidth detour paths in §3. In §4 we propose how detour paths can be exploited. We conclude in §5.

2 Related Work

Routing overlay networks exploit detours to improve the performance and robustness of packet delivery [1,13,3,15]. They delegate the task of selecting paths to applications, which can choose paths that are more reliable, less loaded, shorter, or have higher bandwidth than those selected by the network. Gummadi et al. [3] found that path failures occur frequently, but can be circumvented through random detours. *iPlane* [15] uses measurements from PlanetLab nodes to build a structural map of the Internet that predicts path performance properties, such as latency, bandwidth and loss. While this previous work focused on path availability and end-to-end latency, our focus is on bandwidth.

Prior research has studied bandwidth-aware overlay routing. Lee et al. [11] describe *BARON*, a method for switching to an overlay path with higher available bandwidth. It relies on periodic all-to-all network capacity measurements, which are less transient than available bandwidth measurements. When searching for possible alternative paths, BARON uses high capacity to infer potential for high available bandwidth on a path. Since evaluation results are simulated, it is unclear how a deployment would perform. In contrast, we evaluate the discrepancy between predicted and measured bandwidth on a live system.

Zhu et al. [19] propose an overlay-based approach for selecting a path with high available bandwidth; because their focus is on fairly small networks, they re-measure bandwidth to a large fraction of the network with each path adjustment, which is not scalable. Jain et al. [6] are able to implicitly learn available bandwidth through a video streaming application; they disseminate this information through a link-state protocol with limited scalability.

Split-TCP [7] improves end-to-end throughput by establishing a relay between the two endpoints of a TCP connection. Its benefits have been thoroughly studied in many domains, especially for mobile devices [8]. While our approach for TCP bandwidth detouring benefits from splitting TCP connections, the bulk of improvements result from carefully choosing the right detour nodes (cf. §4).

3 Detour Properties

In this section, we use measurements to demonstrate the existence of bandwidth detours. We show that most measured paths could benefit from detours with higher bandwidth. We also investigate how bandwidth detours change over time and how they compare with latency detours.

PlanetLab. We use PlanetLab to demonstrate the feasibility of bandwidth detouring. Nodes are selected from independent sites to maximise path diversity and avoid known bandwidth restrictions. We created a list of 256 nodes with a bandwidth cap higher than 10 Mbps on May 3rd, 2010. Some experiments used fewer nodes due to node failures or bandwidth limits on PlanetLab. In these cases, we state the actual number of used nodes in the text.

UkairoLab. To circumvent the above limitations and validate measurement results, we also use our own *UkairoLab* testbed hosted on corporate and university machines. It consists of 10 geographically-dispersed nodes located in the US, India, Kenya, UK and France. Their network connectivity is provided by commercial hosting companies, which results in a lower median bandwidth: 4.87 MBps on UkairoLab versus 6.54 MBps on PlanetLab. Machines are virtualised but are dedicated with full kernel access.

3.1 Bandwidth Measurement

To discover detour paths, we must measure a particular bandwidth metric. Since our focus is on the TCP protocol, we consider *bulk transfer capacity (BTC)*, which is the steady-state throughput (in terms of successfully transmitted data bits) of a TCP connection[1]. We measure BTC using the standard *Iperf* tool[2], which observes the throughput of an elastic TCP transfer. We deploy Iperf on 256 PlanetLab nodes and collect all-pairs measurements with a 5 second timeout. We ensure that each node makes only one inbound and one outbound measurement at any point in time. On average, each Iperf measurement takes 8 seconds and consumes 10.8 MBytes.

To understand the variability of bandwidth measurements, we perform repeated measurements at 30 sec, 5 min, 30 min, and 1.5 hour intervals. To stay

[1] We use the terms BTC, throughput and bandwidth interchangeably in this paper.

[2] We explored the use of available bandwidth predictions tools such as Pathload [5] for estimating BTC with lower measurement overhead. However, on average, Pathload took 50 seconds to measure a path, which is too slow for a large deployment.

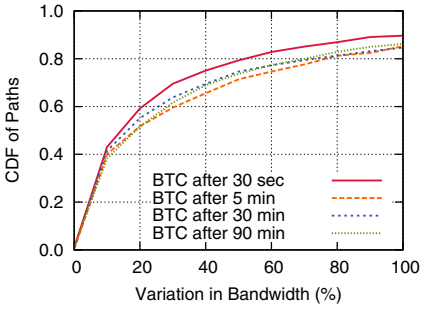

Fig. 1. Bandwidth measurements vary significantly over time.

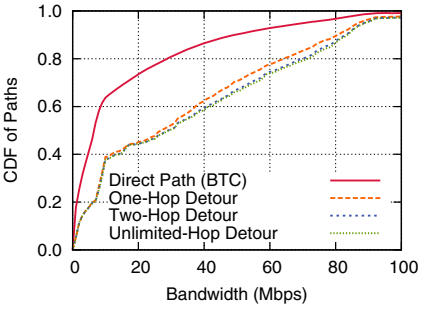

Fig. 2. Detouring via a node increases bandwidth but more hops have little effect.

within Planetlab's 10 GB daily limit, 48 Planetlab nodes measure to 20 randomly-chosen nodes within those 48 nodes. This is repeated three times at different times, measuring 920 paths. As Figure 1 shows, bandwidth can vary significantly, even when measured in quick succession, as confirmed by others [12]. Approximately half of the paths have a 20% variation in bandwidth, regardless of when remeasured. This means that good bandwidth detours have to be significantly better to compensate for this variation.

3.2 Bandwidth Detouring

We want to understand how often traffic between two Internet hosts can benefit from a detour path with higher bandwidth than the direct path. Of the 20 323 successful BTC measurements between 152 PlanetLab nodes, we examined whether detour paths via another node have higher bandwidth. We consider the bandwidth of a detour path as the minimum bandwidth of the paths between the source and the detour node and the detour node and the destination.

Figure 2 shows the cumulative distribution of path bandwidth. We find that 96.6% of all pairs of nodes have a detour path with higher bandwidth. The median increase in path performance is 18.6 Mbps (i.e. a factor of 2.24). We also noted 74.8% of the paths can improve by at least 20% and 1 Mbps. Because detouring via one node can significantly increase bandwidth, we also investigate if additional detour nodes yield similar gains. As the figure shows and confirmed by Lee et al. [11], this is not the case and it provides only minimal additional benefits. We also observe that 40% of paths cannot benefit from detours with more than 10 Mbps bandwidth. This is likely because many PlanetLab paths have 10 Mbps network capacity.

In Figure 3, we compare the relative improvement from bandwidth detouring to latency and loss detouring, discovered by brute-force search. Bandwidth detouring has a significantly larger gain: half the paths can double in bandwidth, while only 13.5% of paths are half the average path latency. We measure loss by

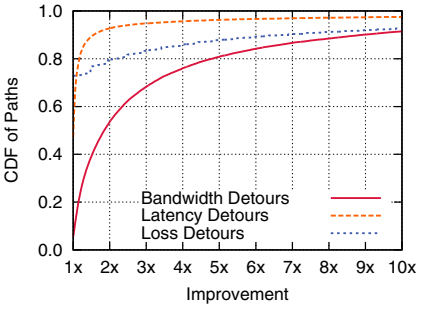

Fig. 3. Bandwidth can be improved significantly more than latency using detouring.

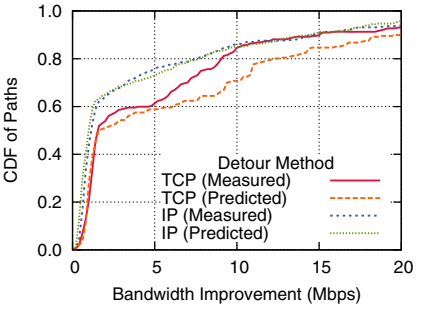

Fig. 4. TCP detouring improves actual bandwidth between nodes significantly more than IP detouring on UkairoLab.

sending 1200 UDP probes with a payload of 1472 bytes and an interval of 100 ms, which is similar to the rate of VoIP connections [15]. Only 27.2% of paths benefit from detouring for loss because most paths suffer no loss at this low rate.

Why are there such a large number of good detour paths with higher bandwidth? Previous studies show that latency detours are due to ISP routing policies [14], which we believe also cause bandwidth detours. We have preliminary evidence that good detours can be found by avoiding one or more autonomous systems (AS) in the default path: for 32% of the pairs of PlanetLab nodes, for which we have complete AS paths, at least one AS in the direct path is avoided more than half the time by the detour. For 29% of the pairs of nodes, the detour paths traverse all the ASes on the direct path. These detours may be due to Internet congestion or differences in intra-domain routing policies.

We expect that "similar" paths in terms of their AS-links would benefit from the same detour nodes. This idea has been exploited in latency detouring [4]—we aim at exploring analogous mechanisms for bandwidth detouring. We leave further investigation of this to future work.

3.3 Bandwidth Detour Properties

Symmetry. We define a detour to be symmetric if the same detour node benefits both the forward and reverse direction of the direct path. Since congestion in the forward path rarely affects the reverse path, we expect bandwidth to be different for each direction. However, our results show that 89% of the 18 036 paths for which we have measurements in both directions, have at least one symmetric detour. We believe this happens because the quality of a detour path is dominated by the properties of the detour node (such as download and upload speed), which are the same in both directions, rather than by congestion on the path. Symmetric detours are better than average: they improve the median path performance by 39% compared to 16% for the asymmetric detours.

Skewness. Detour nodes that have lower latency to the source or destination are more likely to provide higher throughput for TCP transfers. We define the *skewness* of a detour path as the ratio of the absolute difference between the latencies from the detour node to the source and destination to the maximum of the two latencies. As their skewness decreases towards 0, detours are more likely to improve the bandwidth of the direct path: the median skewness value for good detours is 0.43 compared to the median (0.54) of all detours. The reverse case is also true: as skewness increases towards 1, detours are less beneficial for the direct path. In our measurements, the detours that do not benefit the direct path at all have a median skewness of 0.58. These results suggest that low skewness values may be associated with detours that have high-capacity links, and which in turn have a higher probability of being good detours.

Persistence in Time. Given the variability of bandwidth measurements, we investigate the longevity of detour paths: for a detouring platform, short-lived detours would be less useful. Our measurements show that approximately two-thirds of all bandwidth detours persist for more than 90 minutes. This suggests that a platform can make long-term decisions about detour paths.

4 Exploiting Detours

Applications must be able to discover and exploit good bandwidth detours. Here we examine the challenges in implementing a detour routing platform when it consists of cooperative edge or near-edge nodes. In particular, we find that low-level kernel access is not required for good detouring performance.

4.1 Detouring Mechanisms

Two options exist for routing between a pair of Internet hosts via a tertiary detour node: (a) network-level *IP detouring* or (b) transport-level *TCP detouring*.

IP detouring works by encapsulating every IP packet on egress from the source node and sending it to the appropriate detour node, which in turn forwards it to the destination node. From an application standpoint, IP detouring is the more natural approach: (1) it can be deployed transparently because it only operates at the IP layer; (2) it supports both TCP and UDP traffic; and (3) the same detouring mechanism can be used for other metrics such as latency. However, it also has a major disadvantage: the detour path is composed of two complete end-to-end Internet paths. This increases the network-level hop count compared to the direct path. The associated increase in loss probability and latency adversely affects TCP throughput [10].

The alternative to IP detouring is to break the TCP connection at the detour node and use TCP detouring, which is analogous to *split-TCP* [7]. By splitting a long TCP connection into two separate connections terminated mid-path, the feedback-based control loop of TCP becomes more responsive due to reduced path latency. Although this comes at the cost of increased state within the network, this may be acceptable when TCP connections are split by end hosts,

instead of network routers [9]. For TCP detouring, we deploy SOCKS proxies at potential detour nodes and use application-level "socksifying" software to redirect connections via the appropriate detour proxy. This retains the benefit of being transparent to destination nodes and preserves path symmetry.

To compare IP and TCP detouring, we deploy both detouring mechanisms on UkairoLab[3]. We then perform an all-to-all-via-all measurement: for each pair of nodes, we predict and measure the throughput achievable via each of the potential detour nodes using both IP and TCP detouring. For TCP detouring, we predict the throughput of the detoured connection to be the minimum of the throughput of the two paths, i.e. the narrow link [2]. For IP detouring, we also predict the throughput analytically as described in §4.2.

Figure 4 shows the predicted and measured detouring improvement for each method. The results match the intuition that the long TCP paths created by IP detouring adversely affect performance. In contrast, splitting the TCP connection significantly boosted most pairs; for example, 40% of paths improved by at least 5 Mbps. However, the discrepancy between measured and predicted TCP detouring performance is larger for paths which are predicted to benefit more from detouring, suggesting there can exist a bottleneck in forwarding throughput at the detour node.

Although TCP detouring benefits from the effect of a split TCP connection, most improvement comes from choosing a good detour node with respect to the throughput it offers, rather than its latency to the endpoints. For example, 77% of all detours provide at least 10% and 1 Mbps bandwidth improvement; of the detours where the intermediate leg latencies are lower than the direct path latency (which stand to benefit most from a split TCP connection), only 28% provide similar improvements. While the features of IP detouring, such as transparency and UDP support, outweigh those of TCP detouring, we conclude that the performance gains of TCP detouring make it the better choice.

4.2 Analysis of IP and TCP Detouring

Using a stylised stochastic model of TCP's congestion control mechanism [16], the following square-root formula relates the steady-state throughput of a path's BTC to its packet loss probability p and its average round trip delay RTT:

$$\mathrm{BTC} = \frac{\Phi}{\mathrm{RTT}\sqrt{p}}\,. \tag{1}$$

This formula is valid for both the case where loss is independent of the rate, in which case $\Phi = 2$, and the rate dependent case where the loss depends (linearly) on the rate, in which case $\Phi \approx 1.31$. We use this formula to perform a back-of-the-envelope calculation to derive the IP detouring bandwidth.

IP Detouring. Let us denote by BTC_1, p_1, RTT_1, and BTC_2, p_2, RTT_2 the average throughput, the loss and the round trip delay of the constituent paths

[3] We found that, on PlanetLab, the long delay between timeslices due to heavy load severely damaged performance of userspace IP processing.

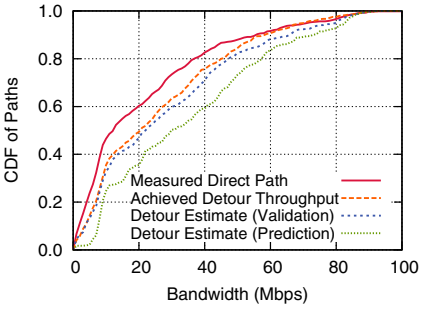

Fig. 5. Detour paths substantially increase throughput. However, bandwidth variations over time can lead to overestimating a given detour's potential improvement.

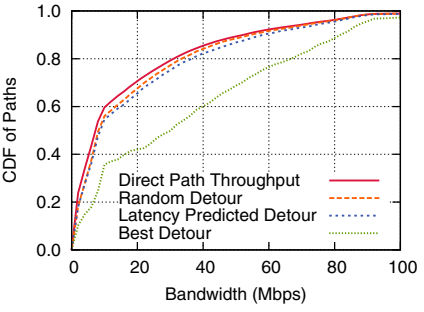

Fig. 6. Latency performs no better than random selection for discovering bandwidth detour paths.

that we will refer to as the first and second leg, respectively. The following approximates the resulting throughput:

$$\mathrm{BTC_{IP}} \approx \frac{\mathrm{RTT_1}}{\mathrm{RTT_1}+\mathrm{RTT_2}}\sqrt{\frac{(\mathrm{RTT_2BTC_2})^2}{(\mathrm{RTT_1BTC_1})^2+(\mathrm{RTT_2BTC_2})^2}}\mathrm{BTC_1} \qquad (2)$$

where we drop the p_1p_2 term in the corresponding square-root expression since the loss probabilities p_1 and p_2 are in general small; in the second equality, we replace p_1 and p_2 using Eq. (1). It is easy to see that the predicted throughput is always strictly smaller than the minimum of $\mathrm{BTC_1}$ and $\mathrm{BTC_2}$, i.e. the respective throughputs of the two legs taken in isolation.

TCP Detouring. Baccelli et al. [2] describe two coupled stochastic differential equations that govern the dynamics of the throughput of the two legs of a detour path. The coupling is dictated by the buffer at the detour node. The key feature of this model is that the TCP throughput of the composed path is, in general, the minimum bandwidth of the two constituent paths given that the buffer at the detour node is sufficiently large. In our system, we ensure this holds.

The above analysis confirms what we observed in practice in Figure 4: IP detouring provides worse performance compared to TCP detouring as predicted by the minimum of the throughputs of the two legs.

4.3 Detouring Overlay Performance

We describe our experience in deploying a TCP detouring platform on 50 PlanetLab nodes. The experiment is divided in two phases: *prediction* and *validation*. First, we measure BTC between all pairs of nodes to predict good detours, consuming on average 571 MBytes per node. We stop after 90 minutes and find that 1845 out of 2019 paths are detourable. We estimate detour bandwidth by taking the minimum bandwidth of the two intermediate legs. In the second validation

phase, we use TCP detouring to validate the best detour for each path. Since we avoid concurrent measurements, the second phase takes substantially longer: after 11 hours, we obtain 689 detourable paths. In Figure 5, we plot the distributions of (a) measured direct path bandwidth; (b) estimated detour bandwidth in the prediction phase and (c) in the validation phase; and (d) achieved detour throughput measured in the validation phase.

We make several observations. First, the median bandwidth improves significantly, from 12 Mbps to 21 Mbps, using TCP detouring. Detours improve the bandwidth on direct paths in 69% of the cases (not shown in the plot). The large increase in bandwidth of detours can justify the fixed measurement overhead per node, assuming at least a modest usage of detoured paths after their discovery to amortise measurement costs. Second, we observe the 10 Mbps egress bandwidth limit present on some PlanetLab nodes. Finally, the benefits of detouring are largely lost at around 50 Mbps, suggesting a throughput bottleneck due to limitations on node performance.

The substantial difference between the detour bandwidths at the time of prediction and the estimated bandwidth at the time of validation may be caused by the variability of bandwidth measurements (cf. Figure 1). Since detour bandwidth is constrained by the minimum bandwidth of the two legs, we see a consistent decrease of around 25% upon validating detour bandwidth a few hours later. Although the *best* detours for any given path may be constantly changing, we can still see temporal consistency in detour path performance.

4.4 Detour Transferability

To discover if good latency detours can also be effective for finding good bandwidth detours, we compare the estimated bandwidth via the best latency detour for each direct path. We measure latency and BTC on 10 265 paths between 136 PlanetLab nodes and compute the best bandwidth and latency detours between each pair of nodes for which we have measurements.

Figure 6 shows the distribution of estimated bandwidth for the best bandwidth and latency detours found through brute-force search, and the estimated bandwidth through detours chosen randomly. As discussed earlier, the best possible detour results in significant improvements over the direct path, although these are likely unachievable due to bandwidth flux. Employing the best latency detour for bandwidth detouring results in performance equal to a random detour. This implies that discovery methods for finding good bandwidth detour based on latency detours are not effective.

5 Conclusions

To understand how to exploit bandwidth detouring on the Internet, we addressed several key questions in this paper. We illustrated the preponderance and longevity of potential bandwidth detour routes: 74.8% of paths had a detour that improved bandwidth by at least 20% and 1 Mbps; and most detours lasted

for more than 90 minutes. Contrary to our initial goals of providing transparent IP-level detouring, we gave evidence that significantly better performance can be achieved through the use of TCP-level detouring. Interestingly, this also means that kernel access is not required for overlay participation, perhaps broadening adoption of a general detouring platform. More research is needed to explore practical and scalable methods for detour discovery and how wide-spread bandwidth detouring would interact with traffic engineering policies by ISPs.

Acknowledgements. We thank Nokia Research IT and David Eyers for hosting UkairoLab machines. Thom Haddow is supported by a Doctoral Training Grant from the UK Engineering and Physical Sciences Research Council (EPSRC).

References

1. Andersen, D.G., Balakrishnan, H., Kaashoek, M.F., Morris, R.: Resilient Overlay Networks. In: SOSP, Chateau Lake Louise, Banff, Canada (2001)
2. Baccelli, F., Carofiglio, G., Foss, S.: Proxy Caching in Split TCP: Dynamics, Stability and Tail Asymptotics. In: INFOCOM 2008, pp. 131–135 (2008)
3. Gummadi, K.P., Madhyastha, H., Gribble, S.D., et al.: Improving the reliability of internet paths with one-hop source routing. In: OSDI (2004)
4. Ho, S.W., Haddow, T., Ledlie, J., Draief, M., Pietzuch, P.: Deconstructing Internet Paths: An Approach for AS-Level Detour Route Discovery. In: IPTPS (2009)
5. Jain, M., Dovrolis, C.: Pathload: A Measurement Tool for End-to-End Available Bandwidth. In: PAM, Fort Collins, CO (2002)
6. Jain, M., Dovrolis, C.: Path Selection using Available Bandwidth Estimation in Overlay-based Video Streaming. Com. Networks 52(12), 2411–2418 (2008)
7. Karbhari, P., Ammar, M.H., Zegura, E.W.: Optimizing End-to-End Throughput for Data Transfers on an Overlay-TCP Path. In: Boutaba, R., Almeroth, K.C., Puigjaner, R., Shen, S., Black, J.P. (eds.) NETWORKING 2005. LNCS, vol. 3462, pp. 943–955. Springer, Heidelberg (2005)
8. Kopparty, S., Krishnamurthy, S.V., Faloutsos, M., Tripathi, S.K.: Split TCP for Mobile Ad Hoc Networks. In: GLOBECOM (2002)
9. Ladiwala, S., Ramaswamy, R., Wolf, T.: Transparent TCP Acceleration. Com. Communications 32(4), 691–702 (2009)
10. Lakshman, T., et al.: Performance of TCP/IP for Networks with High Bandwidth-delay Products and Random Loss. IEEE/ACM Trans. Netw. 5(3), 336–350 (1997)
11. Lee, S.J., Banerjee, S., Sharma, P., Yalagandula, P., Basu, S.: Bandwidth-Aware Routing in Overlay Networks. In: INFOCOM, Phoenix, AZ (2008)
12. Lee, S.-J., Sharma, P., Banerjee, S., Basu, S., Fonseca, R.: Measuring Bandwidth Between PlanetLab Nodes. In: Dovrolis, C. (ed.) PAM 2005. LNCS, vol. 3431, pp. 292–305. Springer, Heidelberg (2005)
13. Lumezanu, C., Baden, R., Levin, D., Bhattacharjee, B., Spring, N.: Symbiotic Relationships in Internet Routing Overlays. In: NSDI (2009)
14. Lumezanu, C., Baden, R., Spring, N., Bhattacharjee, B.: Triangle Inequality and Routing Policy Violations in the Internet. In: Moon, S.B., Teixeira, R., Uhlig, S. (eds.) PAM 2009. LNCS, vol. 5448, pp. 45–54. Springer, Heidelberg (2009)
15. Madhyastha, H.V., Isdal, T., Piatek, M., Dixon, C., et al.: iPlane: An Information Plane for Distributed Services. In: OSDI, Seattle, WA (2006)

16. Padhye, J., Firoiu, V., Towsley, D.F., Kurose, J.F.: Modeling TCP Throughput: A Simple Model and Its Empirical Validation. In: SIGCOMM (1998)
17. Savage, S., Anderson, T., Aggarwal, A., Becker, D., Cardwell, N., Collins, A., Hoffman, E., Snell, J., Vahdat, A., Voelker, G., Zahorjan, J.: Detour: Informed Internet Routing and Transport. IEEE Micro. 19(1), 50–59 (1999)
18. Strauss, J., Katabi, D., Kaashoek, M.F.: A Measurement Study of Available Bandwidth Estimation Tools. In: IMC (2003)
19. Zhu, Y., Dovrolis, C., Ammar, M.H.: Dynamic Overlay Routing based on Available Bandwidth Estimation: A Simulation Study. Com. Networks 50(6), 742–762 (2006)