CRONets: Cloud-Routed Overlay Networks

Chris X. Cai (UIUC) Franck Le (IBM Research) Xin Sun (FIU) Geoffrey G. Xie (NPS)
Hani Jamjoom (IBM Research) Roy H. Campbell (UIUC)

***Abstract*—Overlay networking and ISP-assisted tunneling are effective solutions to overcome problematic BGP routes and bypass troublesome autonomous systems. Despite their demonstrated effectiveness, overlay support is not broadly available. In this paper, we propose *Cloud-Routed Overlay Networks (CRONets)*, whereby users can readily build their own overlays using nodes from global and well-provisioned cloud providers like IBM Softlayer or Amazon EC2. While previous studies have demonstrated the benefits of overlay networks with the high-speed experimental Internet2 backbone, we are the first to evaluate the improvements in a realistic—cloud—setting. We conduct a large-scale experiment where we observe 6,600 Internet paths. The results show that CRONets improve the throughput for 78% of the default Internet paths with a median improvement factor of 3.26 times, at a tenth of the cost of leasing private lines of comparable performance. We also performed a longitidunal measurement, and demonstrate that the performance gains are consistent over time with only a small number of overlay nodes needed to be deployed. However, given the size and dynamic nature of the Internet routing system (e.g., due to congestion and failures), selecting the proper path is still a challenging problem. To address it, we propose a novel solution based on the newly-introduced MPTCP extensions. Our experiments show that MPTCP can achieve the maximum observed throughput across the different overlay paths.**

I. Introduction

In the current interdomain routing system, autonomous systems (ASes) select paths mainly based on their business agreements with other service providers, without taking into account specific performance metrics (e.g., delay, loss, jitter, available bandwidth). This behavior often results in poor performance for the end users. To overcome these problems, researchers have proposed overlay networks, where traffic can be tunneled through intermediate nodes deployed at key access and interchange points to bypass ASes experiencing congestions, failures, or attack [27], [3], [26]. However, despite these efforts, few commercial overlay networks are available, except for costly solutions [31] that are offered by content delivery network (CDN) providers to large web-centric companies. As a result, users (including enterprises of all sizes) cannot readily take advantage of the overlay network solutions to improve their network performance. In particular, we describe two motivating scenarios where such performance boosts are important.

Connectivity between office branches: It is common that enterprises and government agencies consist of multiple branches across different geographic locations. They often lease private network lines between the branch offices to guarantee certain network performance. At the same time, each line typically costs thousands of dollars per month. Given the considerably lower prices of Internet access, overlay networks could be an attractive alternative at a hundredth of the cost [15], [29].

Remote users: VPNs let remote users securely connect to a corporate or home network and maintain access to private applications and data. The quality of such access directly impacts how productive a remote user can be. Therefore, by improving network performance, overlay networks can boost the productivity of the mobile workforce. As a growing number of companies authorize employees to work from home, overlay networks could also help improve the job performance of telecommuters.

In this paper, we propose cloud-routed overlay networks (CRONets), a new approach for end users to reap the benefits of overlay networking in a proactive fashion, *without explicit support from ISPs*. With CRONets, users (e.g., startups) build their own overlay networks by renting virtual nodes from cloud service providers. CRONets leverage four key trends about cloud providers (e.g., IBM Softlayer and Amazon EC2). First, cloud providers continue to expand their global footprint. IBM Softlayer, for example, has more than 40 geographic locations [8]. Second, the provider's cloud data centers are connected by a well-provisioned private network. Third, each cloud data center is aggressively peered with a diverse set of ISPs at Internet exchange points (IXPs). This leads to path diversity from and to most endpoints. Finally, cloud providers allow users to rent and control virtual servers with 100 Mbps virtual NIC from any of those locations starting at about $20 per month.

To the best of our knowledge, cloud-routed overlay networks have never been studied before; little known is about their performance characteristics. Although previous studies (e.g., [3]) have demonstrated the benefits of overlay networks, the studies were conducted more than a decade ago, and most of the overlay nodes were on high-speed experimental academic backbones (e.g., Abilene, Geant). The studies showed that these high-speed experimental academic networks present opportunities for path improvement. However, existing studies have not established available opportunities in the public Internet. On one hand, recent technologies such as Multi-Path TCP (MPTCP) [11] focus on the last mile, suggesting that network bottlenecks occur on the edge of the Internet. If that is the case, overlay networks would not be able to provide significant network improvements. On the other hand, studies on Internet bottlenecks suggest that congestions occur in the Internet core, and overlay solutions should therefore help: for example, in 2003, Akella *et al.* [1] conducted an empirical evaluation of Wide-Area Internet bottlenecks, and

found that although performance limitations in the current Internet are often thought to lie at the edges of the network, half of the bottlenecks were actually in the Internet core. The authors further argue that as access links of stub ASes were upgraded, the ratio of bottlenecks happening in the Internet core were likely to increase. Much more recently, in 2014, Kang and Gligor [19] revealed that the Internet is extremely vulnerable to routing bottlenecks (because of Inter-domain routing policy, hot-potato routing policy, and Internal AS router-level topology) and showed that most bottlenecks links are within or connecting Tier-1 ASes.

This paper therefore focuses on two particular questions. First, *can CRONets provide similar improvements to previous experimental studies, but in a realistic—cloud—setting (e.g., in the face of congestion, failures, etc.)?* Second, *how to choose and place overlays (at scale) or can emerging technologies like MPTCP simplify the selection and overlay placement problems?*

Towards answering these questions, we have built a CRONet and have conducted a measurement study in the last twelve months, using seven data centers from a global cloud provider and hundreds of PlanetLab nodes, with network paths across five continents (North America, Europe, Asia, South America, and Australia). The main results are summarized as follows.

• We conducted a *prevalence measurement*, consisting of real-life Internet servers and Planetlab-hosted clients. In a previous study [5], Banerjee *et al.* stressed that when the source and destination of an end to end measurement belongs to an academic network (e.g., Geant, Abilene), the properties may not be representative of the paths in the Global Internet. However, for "mixed" measurements where the paths traverse some commercial ASes, the results were similar to those where all nodes are all off academic networks. In order to have a large coverage, we use PlanetLab nodes as the clients. However, the servers are in the public commercial Internet, and traceroute traces confirm that the traffic traverses commercial ASes. As such, although we rely on PlanetLab nodes, the measurements do not suffer from the bias of previous studies where source, destination, and overlay nodes were all in academic networks.

We thus observed more than 6,600 paths, and analyzed whether tunneling to virtual overlay nodes deployed on a cloud service provider can improve their performance. This is the first study of this scale: the number of observed paths is one order of magnitude larger than previous studies. Our results show that 78% of the default Internet paths have a lower throughput than at least one overlay path between the same two endpoints, and focusing on those paths with throughput gain, the average and median improvement factors are 4.03 and 2.24 times, respectively. A cloud-based overlay network can also reduce packet loss and the average RTT for more than half of the observed paths. Moreover, default Internet paths with lower performance tend to get greater improvement through the overlay, making the overlay solution particularly useful. Further analysis indicates that a cloud routed overlay path has a high likelihood of increasing the throughput as long as it reduces the packet loss rate and RTT of the default paths simultaneously, by only 12.1% and 10.5% respectively. This result is encouraging because such low reduction thresholds are likely to be met for most scenarios given the path diversity on the Internet. Indeed, our analysis show that cloud routed overlay is capable of creating alternative paths that are substantially different from the default paths: half of the overlay paths created in our experiment contain 57% or less of the routers from the corresponding default paths, and 20% contain only 44% or less.

• We performed a *longitidunal* measurement, focusing on, and continuously observing a subset of 30 Internet paths over an entire week period. The results show that 90% of the 30 observed Internet paths get significant improvements across the measurement period, with an average improvement ratio of 8.39, and a median improvement ratio of 7.58. In summary, the performance gains are consistent over time. In addition, the study revealed that only a small number of overlay nodes needs to be deployed, with one to two overlay nodes being able to provide most of the benefits.

• We propose a new automated path selection algorithm—based on the recent Multi-Path TCP (MPTCP) technology [11]—for the sender to decide which path(s) to send traffic. Taking advantage of the MPTCP congestion control algorithm design objectives [32], the proposed solution guarantees that the obtained throughput equals that of a single-path TCP connection on the best available path. We conducted experiments both in a controlled testbed, and in the Internet to validate it. The results show that MPTCP can achieve the maximum observed throughput across the overlay paths.

The rest of the paper is organized as follows: Section 2 describes our methodology. Sections 3 and 4 quantify the performance gains of the sampled overlay paths and analyze how the gains persisted over time. A variety of analyses aiming to identify the key factors for the performance gains are presented in Section 5, followed by a study of using MPTCP to auto-extract the performance gains from a set of overlay paths in Section 6. Section 7 discusses the related works. related work. Finally, we offer concluding remarks in Section 8.

II. Methodology

Conceptually, CRONets are overlays built from virtual compute nodes provisioned from global cloud providers. As mentioned earlier, CRONets rely on the connectivity provided by cloud data centers to bypass problematic autonomous systems. Realistically, we know relatively little about the network infrastructures of cloud service providers (i.e., how their data centers are connected to each other and to the external world). Therefore, we have chosen a predominantly blackbox in measuring the potential performance gains from using cloud-routed overlay paths. Our goal is to improve the performance of TCP applications across three metrics: (i) throughput, (ii) packet loss, and (iii) packet delay.

We selected a major commercial cloud provider and rented virtual Linux servers as overlay nodes in several of its 40 data-

centers. Each server runs Ubuntu 12.04, and is provisioned with a single core (2.0 GHz), a 100 Mbps network, and 4GB RAM. To act as an overlay node, a virtual server establishes a tunnel (GRE or IPsec) with one endpoint and runs a NAT through the Linux IP Masquerade feature. The NAT allows the return traffic from the other endpoint to also traverse the overlay node, without having to establish any tunnel with that other endpoint.

The measurements were carried out across two stages. First, we performed a series of large-scale file download experiments leveraging PlanetLab nodes (PlanetLab is a global research network with 1343 nodes at 649 sites), and downloading a large file from real-life web servers. Second, we performed a similar large scale experiment, however this time the file was downloaded from servers that we control. For both stages of measurements, we found that a 100 Mbps network capacity is high enough to not become a bottleneck for overlay paths. For this work, we focus on one-hop overlay paths which means each overlay path traverses through exactly one overlay server.

A. Real-Life Web Server Experiment Setup

To measure the performance gain for users with a large range of geographical location distribution, we performed measurement involving over 100 PlanetLab nodes (48 in Europe, 45 in America, 14 in Asia, and 3 in Australia). Each PlanetLab node can be representative of a remote user, or a branch office. Each PlanetLab node downloads a 100 MB file from an "Eclipse" mirror server [12]. As an effort to provide high-speed downloading to users all over the world, Eclipse hosts mirror servers in many different locations. On the Eclipse download webpage, one server is automatically recommended based on the user's location, and the other servers can be manually selected to download from as well. We intentionally select 10 servers from the list to cover a wide geographical range, with locations in Canada, USA, Germany, Switzerland, Japan, Korea and China. To build the overlay network, we rented virtual compute nodes at five data center locations: Washington DC, San Jose, Dallas, Amsterdam, and Tokyo. Thus, we sample a total of 6,600 Internet paths (including one direct, and 5 overlay paths for each pair of Internet endpoints) in this experiment.

For each pair of sender and receiver (A, B), we measure the TCP throughput of the file download for the following four different path types:

Direct: We first measure the characteristics of the direct path $A \rightarrow B$. This measurement represents the performance of the default path as selected by the current Internet routing system. The other five paths are all through an overlay node (denoted by O) that we have set up in the cloud.

Discrete overlay: The discrete measurement analyzes each segment $A \rightarrow O$, and $O \rightarrow B$, separately. This measurement does not take into account the overhead of the tunnels, and the processing overhead at the overlay node. The minimum of the two segments' throughputs can serve as an upper bound of the achievable throughput through the overlay path.

Overlay: We measure the characteristics of the path $A \rightarrow O \rightarrow B$, where O decapsulates/encapsulates the packets, modifies the source/destination IP addresses as a NAT, and forwards the packets. This *tunnel overlay* measurement represents the actual overlay performance over the path $A \rightarrow O \rightarrow B$.

Split-Overlay: Some of the paths are transcontinental paths, where the round trip time (RTT) is likely to be large and potentially limit the TCP performance. We thus hypothesize that running a split-overlay proxy [2], [4] at the overlay node may further improve the performance. Our insight is based on the following observation of TCP congestion control algorithm: Mathis *et al.* developed a simple model for the average bandwidth delivered by TCP for sufficiently long network flows [22]:

$$BW \approx \frac{MSS}{RTT}\frac{1}{\sqrt{p}}, \quad (1)$$

where p is the probability that a packet is dropped. The equation highlights that the throughput over an overlay path $A \rightarrow O \rightarrow B$ is not the minimum of the throughputs over the segments $A \rightarrow O$, and $O \rightarrow B$. For example, when the two segments have similar RTT, the RTT of the overlay path $A \rightarrow O \rightarrow B$ doubles, and the throughput consequently gets halved. To test this hypothesis, we measure the characteristics of the path $A \rightarrow O \rightarrow B$, where the overlay node acts as a split-Overlay, and breaks the TCP connection into two segments. This mode is applicable only when the end points do not enforce IPsec, but the TCP headers are in clear text.

B. Controlled Servers Experiment Setup

To understand the reasons behind the improvements, we then host the TCP sender on one of the virtual servers[1]. This allows us to capture packets, and run different tools (e.g., Traceroute) on the TCP sender. As such, we repeated the above experiments with 50 PlanetLab nodes (26 in North and South America, 18 in Europe, 5 in Asia, and 1 in Australia), and for each PlanetLab node (e.g., planetlab-3.cmcl.cs.cmu.edu), we have five candidate TCP senders, i.e., the virtual servers we have rented at Washington DC, San Jose, Dallas, Amsterdam and Tokyo. When one virtual server acts as a TCP sender (e.g., San Jose), the other four virtual servers (e.g., Washington DC, Dallas, Amsterdam and Tokyo) act as overlay nodes. Thus, we sample a total of 1250 pairs of Internet end points (including one direct, and 4 overlay paths) in this experiment.

By controlling both sender and receiver of each path, we are able to collect much richer data, that include not only throughput result, but also packet loss rate and round trip time (RTT). The latter two metrics can be as important as throughput for many applications such as video conferencing, and online gaming.

As such, for each pair of sender and receiver (A, B), we measure the TCP throughput of a 30-second data transfer, the TCP retransmission rate, and the RTT. The throughput

[1]PlanetLab nodes are not suitable to be TCP senders for our purpose because PlanetLab nodes have a daily outbound traffic limit. After a node sends more network traffic than its limit, its outbound throughput is capped, which affects the measurement results.

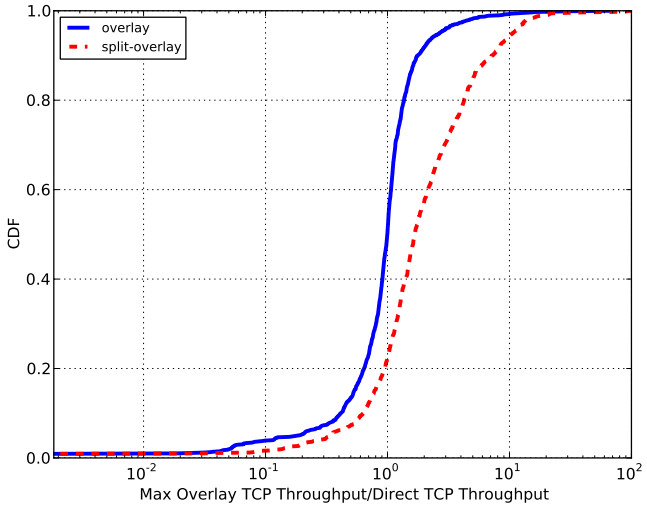

Fig. 1: CDF on the throughput improvement ratios of plain overlay and overlay with split TCP over the corresponding direct paths.

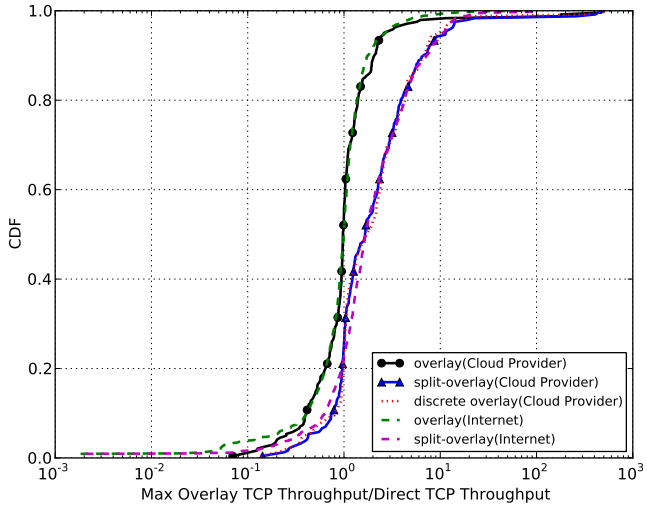

Fig. 2: CDF on the throughput improvement ratios of plain overlay, split-overlay and discrete overlay, over the corresponding direct paths.

is measured through *iperf*, and the retransmission rate and average RTT are derived using *tstat* [23]. Additionally, we collect traceroute data for all the paths.

We have also examined the possibility of collecting the path capacity and available bandwidth estimates using several of the existing tools [30], [18], [13], [14], [10]. However, we found from lab experiments that the capacity and bandwidth estimates are not reliable for paths with high bandwidth links and large RTTs as reported by prior work [14], [10]. An additional difficulty stems from the fact that the cloud nodes are virtual machines subject to software-based rate limiting, which may also significantly impact the accuracy of the estimation made by the tools.

III. Performance Gains

This section presents the results of the PlanetLab and controlled servers measurements.

A. Real-Life Web Servers Measurements

In this subsection, we present the measurement results from our latest real-life web server experiments, collected during January 20-27, 2015 and May 5-12, 2015.

The measurement results are depicted in Figure 1. We first quantify the throughput gain of using an overlay path. For each pair of source and destination nodes, we compare the maximum throughput achieved by the five overlay paths with the throughput of the direct path. The solid blue curve in Figure 1 shows the cumulative distribution function (CDF) of the ratio of the two throughputs. A point (x, y) indicates that for $y\%$ of the observed source and destination pairs, the overlay path offers a throughput improvement ratio no bigger than x. Note that the X-axis represents the improvement ratio in logarithmic scale. We see that for 49% of the source and destination pairs, the improvement ratio is greater than 1, meaning that the throughput between those node pairs is improved by using the overlay network. Focusing on those node pairs with throughput gain, the average and median improvement factors are 1.94 and 1.26 respectively.

We next repeat the experiment but with a split-overlay proxy running at the overlay node. We again consider the ratio of the maximum throughput achieved by the five overlay paths to the throughput of the direct path, and the result is shown in Figure 1 by the dashed red line. We observe that, with split-overlay the overlay tunnel improves the throughput between 78% of the source and destination pairs compared to the direct path. Focusing on those node pairs with throughput gain, the average and median improvement factors are 4.03 and 2.24 respectively. 67% of the source and destination pairs have at least 25% throughput improvement. Compared to the improvement achieved by plain tunnels, the additional gain from using split-overlay is substantial for a large fraction of Internet paths. These results highlight the effectiveness of split-overlay in reducing the perceived RTT of the end-to-end path and subsequently leading to better TCP congestion control behavior.

B. Controlled Servers Measurement

In order to better understand the performance gain via cloud-based overlay network, we repeat the measurement experiments using one of the overlay servers as the TCP sender. We present the measurement results in Figure 2. The black solid curve with circle markers represents the CDF for plain overlay paths. The blue solid curve with triangular markers represents the CDF for split-overlay paths. The curves are labeled with "Cloud Provider" to emphasize that the TCP sender is hosted on a virtual machine in the cloud provider.

We see that for 45% of the source and destination pairs, the improvement ratio is greater than 1, meaning that the throughput between those node pairs is improved by using the overlay network. Focusing on those node pairs with throughput gain, the average and median improvement ratios are 13.51 and 1.33 respectively. Interestingly, we observe that some paths get

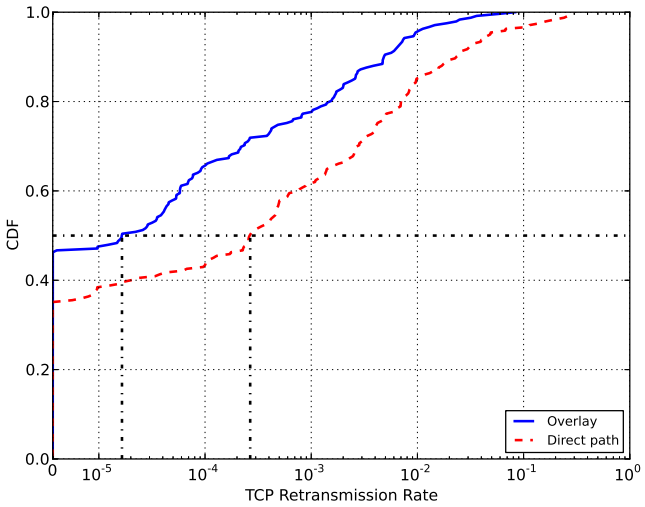

Fig. 3: CDF on the TCP retransmission rates of the direct paths and of the overlay paths. The overlay paths reduce the median TCP retransmission rates by an order of magnitude.

as high as over 400 times improvement. We discuss those paths in details in Section IV.

We observe that split-overlay improves the throughput between 74% of the source and destination pairs compared to the direct path. Focusing on those node pairs with throughput gain, the average and median improvement factors are 12.18, and 2.38, respectively. 59% of the source and destination pairs have at least 25% throughput improvement.

The measurement results from Section III-A are also plotted in Figure 2 by dashed-lines for comparison purpose. The curves are labeled with "Internet" to emphasize that for those cases, the TCP sender was a public web server on the Internet. We notice that both plain overlay and split-overlay paths have similar performance in the two sets of measurements. This result suggested that using cloud provider as source does not introduce significant bias, and we can use the captured packets and traceroute to analyze the reasons behind the results.

Finally, we seek to understand the optimality of the split-overlay results. While we have demonstrated that split-overlay can effectively cut the perceived RTT, a potential concern is that split-overlay may introduce additional processing delay due to the proxy behavior. We thus compare the split TCP-throughput to that of the discrete overlay. Recall that the throughput of the discrete overlay may be considered as the upper bound of the end-to-end tunnel throughput (Section II). The result of the discrete overlay measurement is shown in Figure 2 by the dotted red line. We see that, for 76% of the source and destination pairs, the use of overlay can potentially improve the throughput between them, as modeled by the discrete measurement result. For those node pairs, the theoretical average and median improvement factors are 10.36, and 2.41, respectively. Overall, the results are very close to those of the split-overlay tunnels. Thus we can conclude that using split-overlay effectively achieves the optimal throughput that an overlay can offer.

1) Packet Loss Reduction: To infer the packet loss rates, we use *tstat* [23] and extract the ratio of number of retransmitted bytes, over the total number of bytes sent in the TCP payloads. Although TCP retransmissions may not exactly correspond to packet losses, TCP retransmissions are the main cause for TCP congestion window size reduction, and poor TCP performance. Figure 3 represents the CDF on the number of retransmissions experienced by the TCP transfers over the direct paths (red dotted line) and over the tunnel overlay paths (solid blue line). Recall that for each source and destination node pair, four overlay nodes may be used to set up tunnels. The numbers reported here represent the lowest TCP retransmission rates across the four tunnels for each node pair. A point (x, y) indicates that y% of the TCP transfers experience a TCP retransmission rate no bigger than x. The X-axis is in logarithmic scale, and a value of 0.01 means that 1 out of 100 TCP segments is retransmitted. We observe that the median TCP retransmission rate experienced over the direct Internet paths is 2.69×10^{-2}%, whereas that over the tunnel overlay paths is 1.66×10^{-3}%. The overlay network reduces the median TCP retransmission rates by an order of magnitude.

2) Packet Round Trip Time Reduction: We also analyzed the potential reduction in network delay that an cloud-based overlay network can offer. We observe that for 52% of the source and destination pairs can actually reduce the average RTT between them, and that the overlay is more likely to reduce RTT for direct paths with higher RTTs. For example, the overlay network reduces the average RTT for 68% of the direct paths with 100ms or higher RTTs, and for 90% of direct paths with 150ms or higher RTTs. Details of the measurements and results are in [6].

IV. Persistency of Gains

Section III demonstrated that a cloud-based overlay network can bring significant performance gains. In this section, we aim to answer two questions: First, will the performance gains be consistent over time, and attainable for a large number of end users? Second, how many overlay servers are needed? To answer these questions, we conduct a longitudinal study where we focus on the 30 direct Internet paths with the highest throughput improvements with a split-TCP at the overlay nodes, as measured in Section III-B. We sampled the direct paths' throughput and corresponding split-overlay overlay throughputs 50 times, at an interval of 3-hour over a 7-day period.

Figure 4 depicts the results of the measurement. The X-axis represents the path index: path index 1 corresponds to the Internet path with the largest improvement as measured in Section III-B; path index 2 corresponds to the Internet path with the second largest improvement, etc. For each path, the left bar represents the average throughput of the direct Internet path over the new measurement period. For the right bar, we measure the maximum observed throughput across the four overlay paths and plot the average of the maximum observed throughputs over the same period. The error bar represents the standard deviation. First, with the exception of path indexes

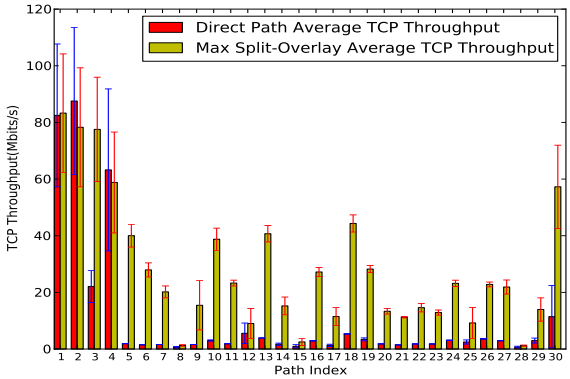

Fig. 4: Comparison of the direct paths' throughput and overlay paths' throughputs for a subset of 30 Internet paths over a one-week period.

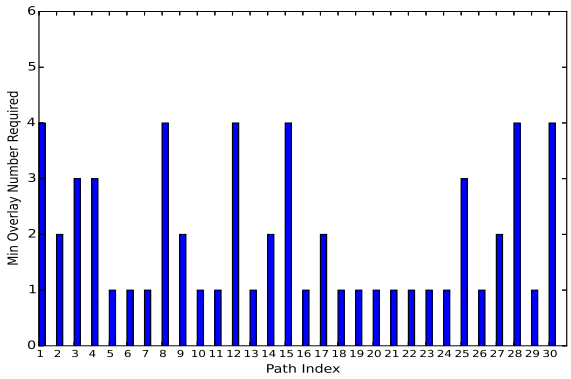

Fig. 5: Minimum number of required overlay nodes for each Internet path to obtain the largest throughput across the measurement periods.

1, 2 and 4, all direct Internet paths consistently obtain a larger average throughput through the overlay network, throughput the one-week period. In other words, *90% of the 30 selected Internet paths get significant improvements across the measurement period, with an average improvement ratio of 8.39, and a median improvement ratio of 7.58*. We look more closely at paths indexes 1, 2, 3 and 4, and all of them have the same destination. In the previous measurement (Section III-B), those paths experienced the largest improvements. However, we observe that the throughputs of path indexes 1, 2, and 4, are now very close to the maximum achievable value of 80 mbps. This explains the reason the overlay network cannot further increase the throughput. We speculate that an intermediate ISP, common to path indexes 1, 2, and 4, was experiencing transient events during the time of the previous measurement. This case highlights the benefit of an overlay network to route around AS(es) with congestion or failure. Second, we observe that for majority of the 30 selected paths the standard deviation values are small, which indicates that *the performance gains are consistent over time.*

Next, we seek to understand how many overlay nodes are needed. Figure 5 illustrates the minimum number of required overlay nodes for each Internet path to obtain the largest throughput across all the observed paths throughout the measurement period. We note that 70% of the 30 selected paths requires only one or two overlay nodes: for example, with only two overlay nodes (O_1, O_2), path index 2 is able to obtain the largest throughput across the measurement period. For some parts of the measurement period, the overlay path through O_1 provides the largest throughput, while for the rest time, the overlay path through through O_2 offers the best performance.

Number of Overlay Nodes	Mean of Ave. Improvement Factors	Median of Ave. Improvement Factors
1	8.19	7.51
2	8.36	7.58
3	8.38	7.58
4	8.39	7.58

TABLE I: Overlay Node Number vs Mean and Median of Ave. Improvement Factors

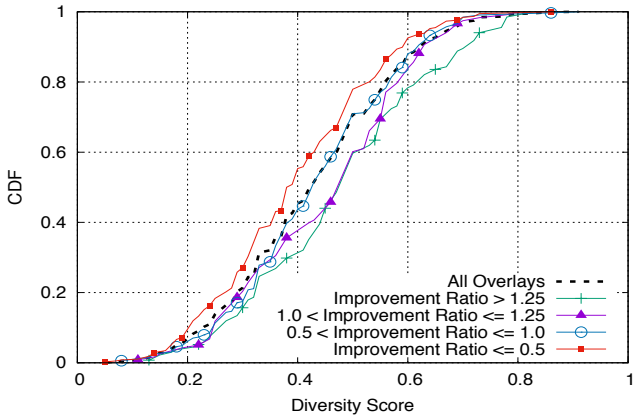

Fig. 6: Overlay paths achieving higher throughput improvement tend to have higher diversity scores.

We then vary the number of considered overlay nodes from 1 to 4 choosing for each path its set of overlay nodes that provides the highest average throughput, and compute the mean of the average improvement factors across the observed 30 Internet paths. The results, summarized in Table I, highlight that *a small number of overlay nodes needs to be deployed; and only one to two overlay nodes can provide most of the benefits*.

Another key question to answer is what locations should customers of overlay network deploy the overlay nodes at. We discuss more about this issue in Section VI.

V. Understanding the Gains

To understand the key factors behind the observed performance gains, We have performed a series of analyses on the data collected from the Planetlab experiment with controlled servers (Sec. III-B). The results are reported in this section. Our focus is on throughput gains as most TCP applications will benefit from an increased throughput.

A. How Much Path Diversity Can an Overlay Create?

The basic motivation for using overlays is to create alternative paths that bypass performance bottlenecks in the direct

paths. Naturally, the effectiveness of those alternative paths is dependent on how "different" they are from the direct paths. In the extreme case, if an overlay path contains all the routers from the direct path, then it will not bypass any bottleneck and thus will not be able to improve performance at all. To capture the path diversity, we define the *diversity score* of a given overlay path as follows:

$$\textbf{diversity\_score} = 1 - \frac{\text{\# of common routers}}{\text{total \# of routers in direct path}}$$

where "common routers" refer to the ones that appear on both the overlay path and the corresponding direct path. The routers on each path are identified from the traceroute output, which is part of the data-set collected from the Planetlab experiment with controlled servers.

It is easy to observe that the diversity score ranges between 0 and 1. Figure 6 plots the CDFs of diversity scores for all overlay paths, as well as for only those that achieve specific throughput improvement ratios. We make two observations. First, most overlay paths are substantially different from the direct paths: 60% have a diversity score of 0.38 or higher, and 25% have a score of 0.55 or higher. Second, overlay paths achieving higher throughput improvement tend to have higher diversity scores as well. For example, among the overlay paths achieving an improvement ratio of 1.25 or higher, 70% have a diversity score of 0.4 or higher, compared to 64%, 56% and 45% having the same score among overlay paths achieving an improvement ratio between 1 and 1.25, between 0.5 and 1, and below 0.5, respectively. An intuitive explanation is that, the more different an overlay path is from the direct path, the more likely that it can bypass some or all bottleneck routers on the direct path.

We have further studied the location of the routers that appear on both overlay and direct paths. For this purpose, we divide each direct path into three equal length segments. We find that the majority (87% averaged across all paths) of those common routers are in the two segments containing the end points, and only a small fraction (13%) are in the middle segment. This result shows that, the use of overlay can create alternative paths that are significantly different from the direct paths in the middle section. This observation confirms the effectiveness of the overlay in bypassing potential bottleneck routers in the network core. As prior works (e.g., [1], [19]) have pointed out, most performance bottlenecks are indeed in the Internet core, making our cloud-routed overlay solution particularly appealing.

B. What Types of Paths Are More Likely To See Improvement?

While CRONets is capable of achieving substantial path diversity, in our Planetlab experiments the level of throughput improvement achieved for different communication end points varies significantly, as shown in Figure 2. In this subsection, we seek to understand where this difference comes from and what kind of Internet paths may benefit the most from CRONets. Specifically, we want to understand how the various attributes (e.g., hop counts, RTT, loss rate) of a path may affect the level of throughput improvement it can potentially get from the use of overlay.

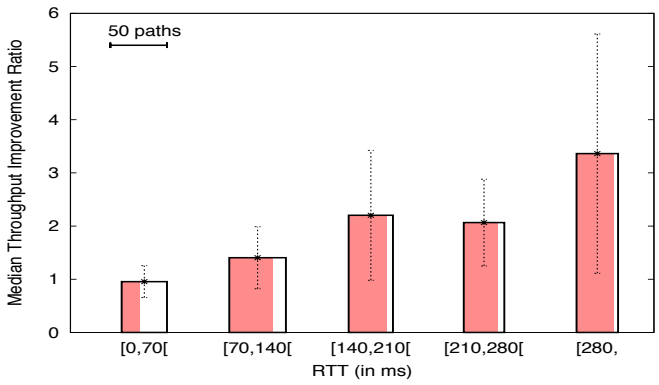

Fig. 7: Paths with higher RTTs are more likely to get throughput improvement by overlay.

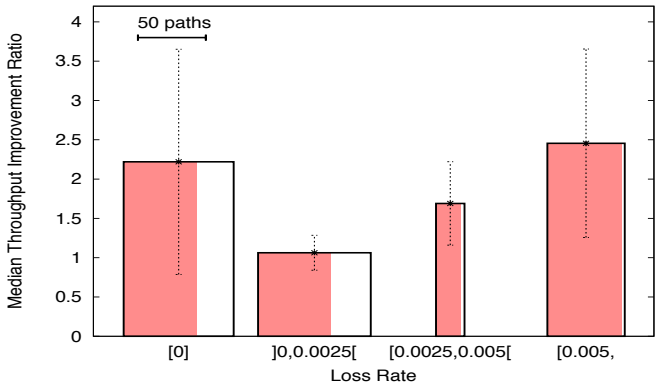

Fig. 8: Paths with higher packet loss are more likely to be improved; but even paths with 0 loss can often be improved.

We first perform a router-level hop count analysis. Surprisingly, we found that, 96% of the overlay paths with throughput improved by more than 25% have a longer hop count than that of the corresponding direct paths, and 45% of those paths have a hop count that is 1.5 times or more than the direct paths. We have also examined the AS level hop counts for a subset of the paths, and the same trend seems to hold.

We then turn our attention to the packet loss and RTT characteristics of the direct paths and seek to understand how they may affect the throughput gain. Figure 7 categorizes all direct paths based on their RTT values, using the five RTT bins as shown (x-axis). The width of a bar represents the number of paths whose RTT values fall into the corresponding bin, and the height of a bar represents the median throughput improvement ratio for all those paths when the overlay network is used. The error bar represents the median absolute deviation. The shade (pink part) within each bar represents the fraction of paths that get positive improvement (i.e., having an improvement ratio greater than one), and this fraction is represented by the fraction of the bar that the shade covers. Figure 8 is a similar figure, but instead, it is based on packet loss rate. We make the following observations. First, for most RTT bins (except for the one of less than 70ms), and for all

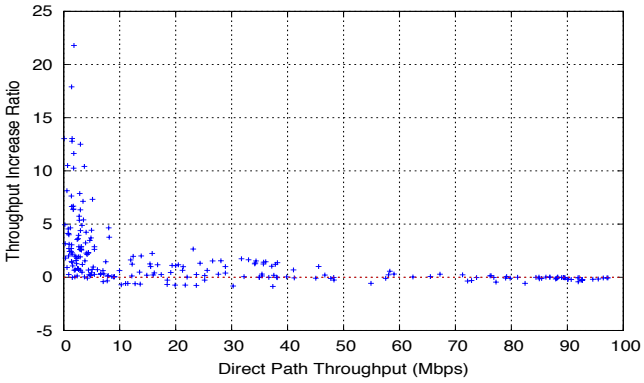

Fig. 9: Paths with smaller throughput get more improvement by using overlay.

loss rate bins, the use of the overlay improves throughput of the majority of the paths. The faction of improved paths is higher for bins of higher RTT or loss rate values. In particular, use of overlay improves more than 84% of the direct paths with RTT of 140ms or higher, and more than 86% of the direct paths with loss rate 0.25% or higher. Second, the throughput improvement ratio is greater for bins with larger RTT values. In particular, the median throughput is more than doubled for paths with RTT of 140ms or higher, and more than tripled for paths with RTT of 280ms or higher. The same trend also applies to loss rate when it is non-zero. For paths with zero loss, however, we observed an interesting polarity: paths either do not get improved by using overlay at all, or they improve significantly as evidenced by the high median improvement factor. Further investigation on the zero-loss paths with significant throughput improvement show that most of those paths have large RTTs, and the improvement is due to the RTT reduction achieved by the overlay.

Given these observations, it is clear that paths with higher RTT and/or packet loss rate are more likely to get throughput improvement when the overlay network is used, and the improvement ratio is also typically higher for those paths. We believe this makes the overlay network particularly useful, as paths with high RTT/loss rate typically have small throughput and thus would benefit the most from the improvement. To validate our intuition, we plot Figure 9. Each point represents a pair of source and destination nodes; the X axis plots the original throughput of the direct path, and the Y axis plots the maximum *throughput increase ratio* achieved by the best overlay path. The ratio is defined as $\frac{T_{overlay}-T_{direct}}{T_{direct}}$, where $T_{overlay}$ is the best throughput achieved by the overlay path and T_{direct} is the throughput of the direct path. The figure shows that direct paths with lower throughput are more likely to get improvement, and the level of improvement is also higher. In particular, almost all direct paths with throughput less than 10 Mbps would be improved with the overlay, and the majority of them would see an improvement ratio higher than one, meaning that the overlay would more than doubled their throughput.

Finally, we have used the C4.5 algorithm (one of the most popular classification algorithms [25]) to *quantitatively* characterize how the combined changes in packet loss and RTT may impact throughput gain. The results indicate that, when an overlay path decreases both the RTT and the loss rate, by at least 10.5% and 12.1%, respectively, it has a high likelihood to increase the throughput. This result is encouraging because such low reduction thresholds are likely to be met for most scenarios involving a direct path with inadequate performance, particularly given our findings that cloud routed overlay is capable of creating substantially different paths as shown in Section V-A. While we omit the classification details here due to lack of space, readers are referred to the Technical Report [6] to see the full analysis.

VI. Adapting to Network Dynamics

Previous sections have demonstrated the potential benefits of a cloud-routed overlay network. To deploy and benefit from it, end-users need to address an additional key question: *Given the dynamic nature of Internet paths, how to determine the best path to use?* To address this question, researchers have traditionally developed algorithms to verify if a path is alive, and evaluate the quality of potential paths. Those algorithms typically rely on active probing, and therefore introduce overhead. In this section, we instead propose a novel solution, based on the newly-introduced MPTCP extensions, to automatically select the best path(s). The solution only requires the support of MPTCP at the end points; therefore, it is applicable to the connectivity between branch offices, and the connectivity between remote users and their corporate office. Section VI-A describes the proposed solution, and Section VI-B presents the conducted validation experiments.

A. Path selection through MPTCP

MPTCP is a new set of TCP extensions that allow two hosts to concurrently use multiple paths between them to improve the overall throughput and reliability [11]. MPTCP is supported by major operating systems including Linux, Android, and iOS. The basic approach in using MPTCP is to run MPTCP proxies [9], (e.g., at each branch office or within the remote users' VPN software) and map the end user originated TCP connections into MPTCP connections. Data packets are tunneled over an MPTCP connection between the MPTCP proxies, and then mapped back to the TCP connections at the egress MPTCP proxy. These MPTCP proxies would support robust communications in the presence of network failures, and adapt to network dynamics, transparently to end users and applications, and with minimal network overhead.

A MPTCP connection set up begins similarly to that of a TCP connection, with a TCP SYN, but the TCP SYN includes a new option, MP_CAPABLE, for the end point to advertise its support of MPTCP. Once successfully established, this connection can be used to transfer data. Through TCP options, each endpoint can optionally advertise additional network addresses. Then, additional MPTCP subflows can be created, and combined to the existing MPTCP connection, to appear as a single connection to the user applications. Also, MPTCP includes connection-level sequence numbers to allow the end

points to reassemble the segments arriving on the different subflows, and potentially out-of-order or duplicated. MPTCP has its own congestion control algorithm [32], which ensures that the total MPTCP throughput is at least as high as that of a single-path TCP connection on the best available path. At the same time, the MPTCP congestion control algorithm is designed not to take up more capacity on its different paths than if it were a single-path TCP connection using only one of these paths. This is to ensure that MPTCP will not degrade the performance of applications that use single-path TCP at shared bottlenecks. If one of the paths fails, MPTCP detects it and falls backs to using the remaining paths. MPTCP takes advantage of path diversity, and guarantees a throughput equal to that of a single-path TCP connection on the best available path.

We exploit these capabilities of MPTCP by creating proxies at the sites to be connected. Each MPTCP proxy has access to $N + 1$ paths, where N is the number of overlay nodes. One of these paths goes directly to the other proxy, while the other paths are reflected off the overlay nodes. If the default Internet path fails, the two proxies can still continue their connections through the overlay paths. This solution should incur minimal overhead as there is no separate need to probe the different paths and discover the available bandwidth on each of them. Instead, the MPTCP congestion control will infer this information based on the received ACKs for every sent data segment.

B. Validation

In this section, we present the validation experiments to verify that MPTCP can be used to automatically select the best performance path.

Setup. We deploy 9 virtual servers across USA, Europe and Asia. We select a pair of servers to act as MPTCP proxies, and use the other 7 servers as overlay servers. A pair of MPTCP proxies therefore has 8 paths (1 direct path, and 7 overlay paths) between them. We then compare the single-path TCP throughput via the direct path with the MPTCP throughput between MPTCP proxies. We measure the throughput for 1 minute using iperf. We repeat the measurement for 5 iterations with a 6-hour interval between iterations. For MPTCP, we configure the congestion control to OLIA [21].

Results. We measure 72 Internet paths among the 9 servers and focus on the 15 Internet paths where the throughput over the direct path presents the lowest values. For each of these paths, we perform four measurements. Figure 10 compares the results for four configurations: (i) the throughput over the direct path, (ii) the maximum observed throughput through the overlay network, (iii) the maximum observed throughput through the overlay network where each overlay node also acts as a split-Overlay proxy, and (iv) the throughput when using MPTCP. We observe that MPTCP can achieve the maximum throughput of the overlay network reliably with small variation for a majority of the paths. These results alleviate the need to correctly identify the best performing

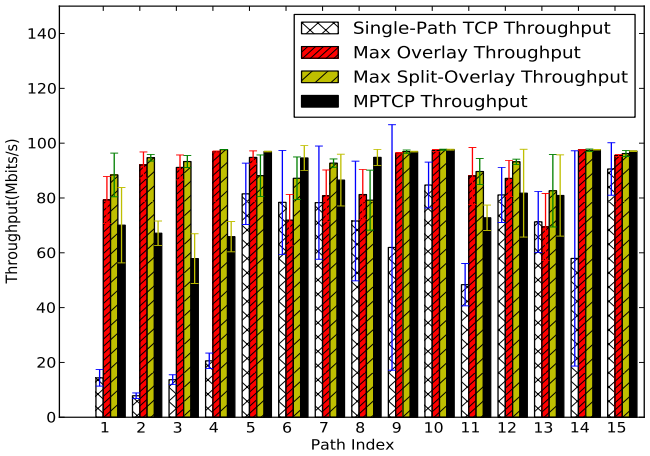

Fig. 10: MPTCP versus Direct, plain overlay and split-Overlay, using OLIA as the congestion control

overlay node(s) (Section IV). As long as the best performing overlay node(s) belong to the set of overlay nodes, MPTCP will provide the maximum throughput across all the paths. For some instances (e.g., path index 1, 2, 3, 4), we observe that the MPTCP throughput is lower than the expected result. However, for other instances (e.g., path index 5, 6, 7), we observe the opposite. As such, we speculate the Internet variations (i.e., available bandwidth in the Internet may vary over time) to be responsible for those differences, but further investigation will be required to identify the causes.

C. Congestion Control

The rationale behind the MPTCP congestion control [32] is not to degrade the performance of applications that use single-path TCP at shared bottlenecks. As a result, MPTCP congestion control is designed to achieve a throughput not higher than that of a single-path TCP connection on the best available path. However, preliminary users of CRONets have questioned this restriction in the specific context of overlay networks: because users of CRONets rent the overlay nodes and their associated bandwidth (e.g., the network connectivity of the overlay nodes can be upgraded to 1 Gbps or 10 Gbps at additional cost), those users have questionned why not letting each MPTCP subflow run independently, and the total MPTCP throughput be the sum of the throughputs on the different paths. Figure 11 represents the results when we modify the congestion control to Cubic [17]. We observe that MPTCP consistently achieves a throughput close to 100 Mbps, which is the limit of the NIC cards at the endpoints.

VII. Related work

The Detour framework [7], [27] first showed that a large fraction of Internet paths could get improved performance through indirect routing [28]. A number of proposals then ensued. For example, Andersen *et al.* proposed Resilient Overlay Networks (RON) [3]: Through a wide-area deployment, RON was shown to improve the loss rate, latency or throughput of data transfers. Miyao proposed an overlay architecture to accelerate content delivery between data centers [24]. Akamai's

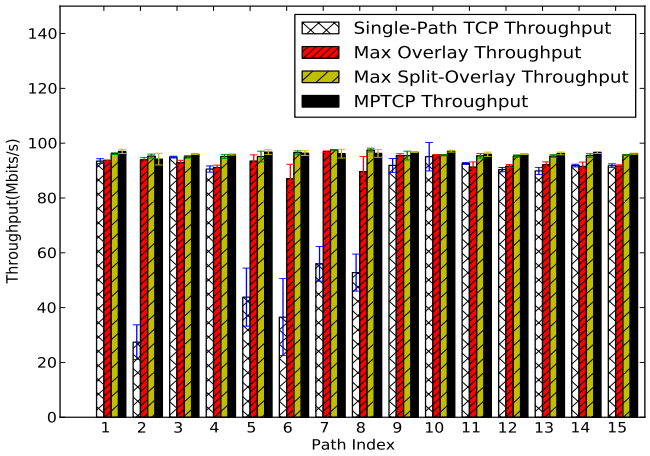

Fig. 11: MPTCP versus Direct, plain overlay and split-overlay using Cubic as the congestion control.

commercial offering SureRoute [31] to route around failures and improve performance is also based on an overlay system. More recently, Peter *et al.* developed and evaluated a system called ARROW [26] that allows end users to create tunnels to participating ISPs, and stitch together end-to-end paths. ARROW's main focus is to enhance the robustness of end-to-end paths against attacks and failure events. Building upon these efforts, we investigate the feasibility of leveraging the *relatively inexpensive and readily available* resources from the cloud and the emerging MPTCP technology to significantly lower the deployment barrier. Furthermore, we examine the low-level path characteristics to understand the key factors behind the performance gains.

Source control over AS level routing has also been suggested to improve the reliability of Internet paths. For example, Yang *et al.* presented a solution that gives users the ability to choose the sequence of providers the data packets traverse [33]. Gummadi *et al.* implemented a one hop source routing solution to recover from Internet path failures [16]. While viable as a long term option, such source routing currently is often blocked by intermediate ISPs due to security concerns [20].

VIII. Conclusion

In this paper, we proposed CRONets, an overlay layer built using virtual nodes from cloud providers with global footprints. Using extensive evaluation, we demonstrate that CRONets achieve three goals: (1) consistently improve the performance of most network paths by an average of 3.26 times, (2) uses MPTCP to solve the overlay path selection problem, and (3) is incrementally deployable in today's Internet without requiring support from ISPs. For future work, we will explore the adoption and deployment of MPTCP proxies to enable non-MPTCP enabled endpoints (e.g., Internet servers) to benefit from MPTCP and CRONets.

References

[1] Aditya Akella, Srinivasan Seshan, and Anees Shaikh. An empirical evaluation of wide-area internet bottlenecks.

[2] L. A. Alt, J. P. Rohrer, and G. G. Xie. Demo: Application-transparent deployment of dtn via smartnet.

[3] D. Andersen, H. Balakrishnan, F. Kaashoek, and M. Robert. Resilient overlay networks. In *ACM SOSP*, 2001.

[4] A. Bakre and B. R. Badrinath. I-tcp: Indirect tcp for mobile hosts. In *IEEE ICDCS*, 1995.

[5] Suman Banerjee, Timothy G. Griffin, and Marcelo Pias. The interdomain connectivity of planetlab nodes.

[6] Chris Cai, Franck Le, Xin Sun, and Geoffrey Xie. BYOO: Build Your Own Overlay. Technical report, 2015.

[7] A. Collins. The detour framework for packet rerouting. 1998. Master's thesis, University of Washington.

[8] L. Crosby. What's next? $1.2 billion investment. 15 new data centers. 2014. Available at http://blog.softlayer.com/2014/whats-next-1-2-billion-investment-15-new-data-centers/.

[9] L. Deng, D. Liu, T. Sun, M. Boucadair, and G. Cauchie. Use-cases and requirements for mptcp proxy in isp networks.

[10] C. Dovrolis, P. Ramanathan, and D. Moore. Packet-dispersion techniques and a capacity-estimation methodology. *IEEE/ACM Trans. Netw*, 2004.

[11] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure. Tcp extensions for multipath operation with multiple addresses. RFC6824.

[12] The Eclipse Foundation. Eclipse. https://eclipse.org.

[13] E. Goldoni, G. Rossi, and A. Torelli. Assolo, a new method for available bandwidth estimation. 2009.

[14] E. Goldoni and M. Schivi. End-to-end available bandwidth estimation tools, an experimental comparison. In *International Conference on Traffic Monitoring and Analysis*, 2010.

[15] Andy Gottlie. Why does MPLS cost so much more than internet connectivity? 2012. Network World.

[16] K. Gummadi, H. Madhyastha, S. Gribble, H. Levy, and D. Wetherall. Improving the reliability of internet paths with one-hop source routing. In *USENIX OSDI*, 2004.

[17] Sangtae Ha, Injong Rhee, and Lisong Xu. Cubic: A new tcp-friendly high-speed tcp variant. *SIGOPS Oper. Syst. Rev.*

[18] M. Jain and C. Dovrolis. Pathload: A measurement tool for end-to-end available bandwidth.

[19] Min Suk Kang and Virgil D. Gligor. Routing bottlenecks in the internet causes, exploits, and countermeasures. In *Technical Report: CMU-CyLab-14-010*, 2014.

[20] E. Katz-Bassett, H. V. Madhyastha, V. K. Adhikari, C. Scott, J. Sherry, P. van Wesep, T. Anderson, and A. Krishnamurthy. Reverse traceroute. In *USENIX NSDI*, 2010.

[21] Ramin Khalili, Nicolas Gast, Miroslav Popovic, Utkarsh Upadhyay, and Jean-Yves Le Boudec. Mptcp is not pareto-optimal: Performance issues and a possible solution.

[22] M. Mathis, J. Semke, J. Mahdavi, and T. Ott. The macroscopic behavior of the tcp congestion avoidance algorithm. 1997.

[23] M. Mellia. Tcp statistic and analysis tool. Available at http://tstat.tlc.polito.it/publications.php.

[24] Y. Miyao. An overlay architecture of global inter-data center networking for fast content delivery. In *IEEE ICC*, 2011.

[25] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers Inc., 1993.

[26] Q. Zhang D. Woos T. Anderson A. Krishnamurthy S. Peter, U. Javed. One tunnel is (often) enough. In *ACM SIGCOMM*, 2014.

[27] S. Savage, T. Anderson, A. Aggarwal, D. Becker, N. Cardwell, A. Collins, E. Hoffman, J. Snell, A. Vahdat, G. Voelker, and J. Zahorjan. Detour: a case for informed internet routing and transport. *IEEE Micro*, 1999.

[28] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. E. Anderson. The end-to-end effects of internet path selection. In *ACM SIGCOMM*, 1999.

[29] Jonathan Shieber. Virtualizing wide area networks, velocloud rings up $21 million. 2014.

[30] J. Sommers, P. Barford, and W. Willinger. A proposed framework for calibration of available bandwidth estimation tools. 2006.

[31] Akamai Technologies. Sureroute. Available at www.akamai.com/dl/feature_sheets/fs_edgesuite_sureroute.pdf.

[32] D. Wischik, C. Raiciu, A. Greenhalgh, and M. Handley. Design, implementation and evaluation of congestion control for multipath tcp. In *USENIX NSDI*, 2011.

[33] X. Yang, D.Clark, and A. Berger. Nira: A new inter-domain routing architecture. *IEEE/ACM Trans. Netw.*, 2007.