Passively Monitoring Crowds to Detect and Isolate End-to-End Performance Issues in Wide-Area Services

He Yan[1] Ashley Flavel[2] Zihui Ge[2] Alexandre Gerber[2] Dan Massey[1]
Christos Papadopoulos[1] Dan Pei[2] Hiren Shah[2]

[1]Colorado State University [2]AT&T Labs - Research

ABSTRACT

Over the past few years, Internet Service Providers (ISPs) have been rolling out a wide range of value added services beyond basic connectivity, such as web hosting, content distribution network (CDN) service, database, gaming, cloud computing and e-commerce server hosting. In this paper, we focus on the detection and isolation of performance issues in such ISP-hosted wide-area services. In contrast with widely-used service performance monitoring approaches in which active probing devices need to be strategically placed in wide-area networks, our technique utilizes passively monitored traffic data at the server access routers, the ISP network topology, and BGP routing information to detect and isolate service impacting events. We first present an in-depth analysis and characterization of the TCP round trip latency dynamics observed from the client requests in a CDN service managed by a tier-1 ISP. Based on our observations, we design a passive hierarchical anomaly detection and isolation system (PHADIS) for service management operators and deploy it in the ISP. Our results demonstrate that PHADIS is very effective in accurately and quickly pinpointing important service problems, which could be easily missed by active probing approaches.

1. INTRODUCTION

Over the past few years, Internet Service Providers (ISPs) have been rolling out a wide range of value added services beyond basic connectivity, such as web hosting, content distribution network (CDN) service, database, gaming, cloud computing and e-commerce server hosting. These services have vast numbers of customers from throughout the Internet. They are typically hosted in geographically distributed data-centers that are often collocated with ISPs' Point of Presence (PoPs) and managed by the same ISPs. Detecting and localizing end-to-end performance issue in these wide-area services is critical for ISP operators to improve the service quality perceived at wide-area end users , for example, through fast service impairment detection and flexible mitigation control.

First, the services we are interested in studying cover a vast number of users from diverse locations. Without active probes from a vast number of network locations throughout the Internet, the monitoring coverage is limited and some performance issues will not be detected. Second, even when performance issues are identified, the of the performance issues is limited by the number, source location and frequency of the probes. Finally, a significant number of probe packets place additional overhead on the network infrastructure and may be treated differently than normal packets.

In this paper, we argue that the most effective way to detect and localize end-to-end performance issues in an ISP hosted wide-area service is to passively monitor the client IPs from inside the ISP network. As service and networking are seamlessly integrated together, it is advantageous to make use of the information available from inside the ISP network. Specifically, we propose a novel approach that utilizes passively monitored traffic data at the server access routers, the ISP network topology, geo-location information and the BGP routing information to detect and localize end-to-end performance impacting events. In order to design a practical passive monitoring approach for detecting and localizing end-to-end performance issues, there several open questions that must be answered:

- How many client IPs do we need to keep track of in a typical wide-area service? Will this pose a scalability issue? How diverse are the locations of these IPs?
- How frequently can we collect passive performance measurements from client IPs? This determines the timeliness of detecting performance issues.
- How accurately can we detect performance issues for individual client IPs given their passive performance measurements?
- How do we localize and prioritize service anomaly events?

Our main contributions can be summarized as follows.

1. To answer the above open questions, we conduct an in-depth analysis and characterization of the TCP round trip time (RTT) passively measured from a CDN service hosted and managed by a tier-1 ISP. [1]

[1]The passive measured data is anonymous and this study complies with the ISP's privacy policy.

2. Based on the observations from the above analysis, we identify a few fundamental challenges in client IP level anomaly detection and localization.

3. To address the challenges identified above, we propose a novel approach to aggregate client IP level measurements along a topological hierarchy built by using ISP internal information such as the ISP network topology, and the BGP routing information.

4. Based on the novel approach, we design and implement a passive hierarchical anomaly detection and localization system (SONAR) for detecting and localizing end-to-end performance in real-time.

5. SONAR has been successfully deployed in a tier-1 ISP to monitor millions of client IPs of its CDN service and helps operators to detect and localize performance issues.

This paper focuses on the RTT performance metric and a CDN service, but the data analysis and the SONAR system that we will present can be easily extended to other wide-area services and performance metrics.

The organization of the rest of the paper is as follows. Section 2 presents results from client IP level passive RTT measurements and introduces a few main challenges in client IP level anomaly detection and localization. Section 3 addresses these challenges by making use of the topological hierarchy that is built from the ISP's BGP routing information and network topology. In Section 4, we present the design of SONAR, which is a passive hierarchical anomaly detection and localization system In Section 5, we then present a few representative case examples from our operational experience with running the SONAR system over several months in a tier-1 ISP. Section 6 evaluates the accuracy of SONAR by using a list of labeled evens from the CDN service team. Finally we present the related work in Section 7 and conclude the paper in Section 8.

2. ANALYSIS OF PASSIVELY MEASURED RTTS

We apply passive measurement techniques to a CDN service operated by a tier-1 ISP. In particular, we monitor the end-to-end TCP round trip time (RTT) between client hosts and CDN servers as web service requests arrive. We focus on the RTT since many applications are extremely sensitive to network RTT (e.g., gaming). In the context of CDN service, the TCP throughput of large objects, which are more likely hosted by CDN, are expected to be inversely proportional to RTT [16], making it an important factor for CDN service providers.

A simple and common way to measure end-to-end RTT is to compare the timestamps of IP packets during the TCP handshake. In our case, one traffic monitor is installed for each CDN node (data center). The monitor observe the access links that connect the CDN node to the ISP backbone and it is configured to capture TCP handshake packets. When a request is observed, the traffic monitor calculates the time difference between the first SYN (from client to CDN server) and the corresponding ACK that completes the handshake (also from client to CDN server). This becomes the estimated RTT between the CDN node and the client. This RTT includes network propagation delay, any queuing delay (e.g., due to congestion inside network), and server side as well as client side processing delay.

Each TCP connection (e.g. successful handshake) made by a client IP results in a single RTT measurement. This paper simply refers to a series of passively measured RTTs associated with **a single client IP** as the **client RTT series**. These RTT series can be an important performance indicator quantifying the service quality perceived by the CDN clients over time.

We analyze the client RTT series from three CDN nodes over a 10-day period (April 1st to April 10th, 2010). These three CDN nodes are located in northeast, southeast and northwest regions of USA respectively. The three datasets are hence named Northeast, Southeast and Northwest. Table 1 summaries the details of the three datasets. To protect proprietary information, we cannot list actual numbers of connections, client IPs and egress routers here. For example, for dataset Northeast, tens of millions of connections were observed from several millions of client IPs, 202,252 subnets, 23,869 BGP prefixes, 5,116 different AS paths and several hundred different egress routers. As we can see, the client RTT series provides a wide coverage of client IPs, subnets, BGP prefixes, AS paths and egress routers. We should note that the differences in coverage among three datasets are not caused by CDN assignment strategy. Instead, they are due to the incomplete deployment of traffic monitoring devices (for Southeast and Northwest) at the time of this study.

2.1 Variability in Client RTT Series

2.1.1 Variability across Client RTT Series

Figure 1 shows the cumulative distribution of RTTs from all client RTT series in three different datasets. We normalize each RTT by the maximum RTT of all three datasets to protect proprietary information. We observe: (a) There is a large disparity in RTT's distribution for each dataset. All three datasets show significant variation (4 orders of magnitude) in per-connection RTT. (b) To a large extend the three datasets show a similar RTT distribution. In particular, for every dataset, a large faction of all RTTs has small or medium values while a small fraction of RTTs have large values.

The above observations suggest that the variability of RTTs across all client RTT series is large. On one hand, the large variability may be due to path diversity – connections from different client IPs traverse different paths and each path may have different typical RTT. For example, a client IP in South America assigned to the Southeast node would more

Dataset	# Connections	# Client IPs)	# Subnets	# Prefixes	# AS paths	# Egress Routers
Northeast	tens of millions	several millions	202,252	23,869	5,116	several hundreds
Southeast	tens of millions	several millions	41,784	3,613	649	several tens
Northwest	tens of millions	several millions	66,464	14,269	2,583	several hundreds

Table 1: Network Coverage of Three Datasets

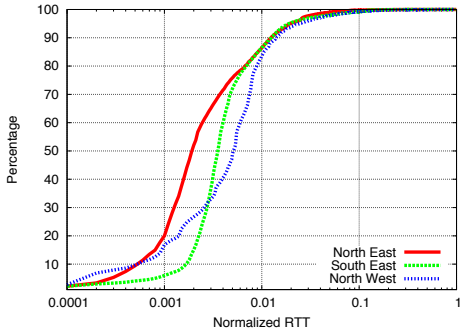

Figure 1: Distribution of normalized RTT across client RTT series

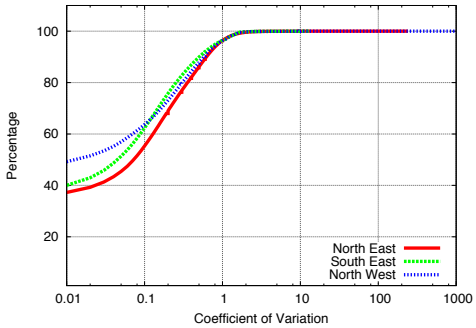

Figure 2: Distribution of coefficient of variation for individial client RTT series

likely experience a greater RTT than a client IP in Florida simply due to the longer distance. On the other hand, the large variability may be due to time dynamics – RTT from the same client IP varies over time. For example, different connections from the same client IP may have different RTTs because of routing change or queuing fluctuation during a day. In our context of anomaly detection, we are more interested in the latter case, where different connections from the same client IP have largely varying RTTs. These variations may indicate some potential RTT anomalies. But it is not clear from Figure 1 how largely RTTs of different connections from the same client IP vary. To better understand this issue, we next examine the variability in RTTs within individual client RTT series.

2.1.2 *Variability within Individual Client RTT Series*

We use the coefficient of variation (CV) metric to quantify the variability in RTTs within individual client RTT series. In other words, we are interested in the variability in RTTs measured from different requests of the same client IP address during the 10-day period. Figure 2 shows the cumulative distribution of CV for all client RTT series in each of the three datasets. Note that x axis is in log scale. We can see from Figure 2 that all three datasets have similar patterns. Specifically, around 60% of client RTT series have very small CV (less than 0.1), which implies good predictability when using average historical RTTs to forecast future RTTs. Almost 35% of client RTT series show medium CV (ranging from 0.1 to 1), which indicates reasonable predictability. However, we also noticed that 5% of client RTT series exhibit large CV (ranging from 1 to 240). The large variability observed in these client RTT series suggest that RTT anomaly detection for the corresponding client IPs may be challenging.

2.1.3 *Self-inflicted RTT Increase*

To better understand these 5% client RTT series with huge variability, we drill in on some examples. We discover an interesting phenomena that contributes to the large variability — consecutive requests within a very short time period (second or sub second level) have almost monotonic increasing RTT value. For example, in one case, we observe 32 requests from the same client IP having subsequent RTT value increased from 25.84 ms to 202.04 ms within one second.

Our conjecture to this behavior is that the RTT increase is *self-inflicted*. As recommended by HTTP 1.1 standard [13], modern browsers such as IE7, Firefox, Safari and Opera use multiple TCP connections in parallel to fetch different objects on the same page. Although HTTP 1.1 recommend 2 parallel TCP sessions, most of the latest releases of these browsers use much more concurrent connections: Firefox 3.5.9 and IE8 use 6 and Safari 4.0.5 uses 4 TCP sessions. Thus TCP SYN-ACKs from the CDN server are likely queued one after another at the client side access link or in processor buffer. Furthermore, data packets from different web servers may also get into the queue – for example, advertisement, javascripts, stylesheet file on the same webpage may not be hosted on the CDN server. Since each 100-byte packet queued over a 64 Kbps access link would increase the RTT of subsequent TCP sessions by 12.5 ms, it can quickly create a significant increase over several packets. Such self-inflicted RTT increases do not reflect any real performance problem for the CDN service, hence need to be carefully handled when we use client RTT series for performance impairment detection.

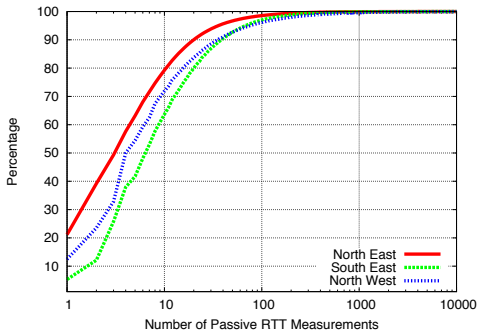

Figure 3: Distribution of number of measurements in client RTT series

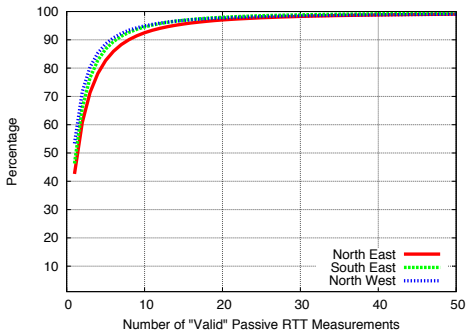

Figure 4: Distribution of number of "valuable" measurements in client RTT series

2.2 Sparsity of Client RTT Series

RTT measurements are only collected when a client IP contacts a CDN node. In order to have timely measurements to detect network performance issues along the path between client IPs and CDN nodes, client IPs need to communicate with CDN nodes often enough. In order words, if a client IP doesn't contact a CDN node very often, its client RTT series may be too sparse to reflect any problem along the path that client IP travels. In order to understand how often a client IP contacts a CDN node, we first plot the CDF of number of connections for individual client IPs using the three datasets.

Figure 3 shows that most of client IPs have very few connections over a 10-day period. This is true of all three data sets. More specifically, in Northeast dataset, 80% client IPs have less than 10 connections; in Southeast dataset, 70% client IPs have less than 10 RTT measurements; in Northwest dataset, 65% client IPs have less than 10 connections. In other words, 10 passive measurements from 10 connections are too few to reflect performance problems on the path over a period of 10 days. The number of RTT measurements alone may not be sufficient to determine the measurement sparsity. For example, even though a client IP contacts a CDN node many times within the same second (doesn't contact the CDN node at other times), its client RTT series is still considered sparse as all these measurements only reflect the path performance at that single second.

In order to better understand the sparsity of client RTT series, we further define a RTT measurement to be "valuable" only if it is at least 600 seconds later than the previous RTT measurement. As in general path RTT appears steady at least for 600 seconds[22], passive measurements are within a period of 600 seconds should be considered as a single sample of end-to-end RTT. Ideally, we would like to have one passive measurement every 600 seconds in order to better monitor the path RTT. Figure 4 shows most of client IPs have even fewer "valuable" RTT measurements over a 10-day period compared to Figure 3. For all three datasets, 90% client IPs have less than 10 "valuable" passive RTT measurements.

As our approach purely depends on passive monitoring, we cannot change how often client IPs contact CDN nodes to solve the sparsity problem. All of these suggest client RTT series are too sparse to detect network performance issues.

2.3 Summary

The naive approach of detecting end-to-end performance issues would be applying anomaly detection algorithms directly on the client RTT series. In other words, for each client IP, keep track of its client RTT series and detect abnormal RTTs deviated from its normal behavior that is built based on the history. But the above analysis suggests there are several limitations in this naive approach.

- First of all, It won't scale with respect to the number of client IPs. For example, in Northeast dataset, there are several millions of client IPs during a 10-day period. It is not trivial to keep track of several millions of client IPs.
- Secondly, the larger RTT variability within some client RTT series makes anomaly detection very challenging.
- Thirdly, client RTT series usually are too sparse to conduct a statistical anomaly detection.

3. AGGREGATION ALONG TOPOLOGICAL HIERARCHY

As anomaly detection based on client RTT series is not practical, we adopt a different approach by aggregating client RTT series into higher level clusters according to the topological hierarchy. In order to illustrate the idea of aggregation, we use the hierarchy shown in Figure 5 as an example, where the client RTT series are aggregated into subnet clusters, BGP prefix clusters, AS path clusters and egress router clusters. Here the AS path means the reverse AS path from the CDN node to the client IP and the egress router means the router at which data traffic from the CDN node to the client IP exits the ISP network. In other words, for each cluster in high levels, a **aggregate RTT series** is formed by aggregating the client RTT series from all the client IPs that are its children in the hierarchy. As a result, anomaly detection approach can be applied on the newly formed aggregate RTT series instead of the client RTT series.

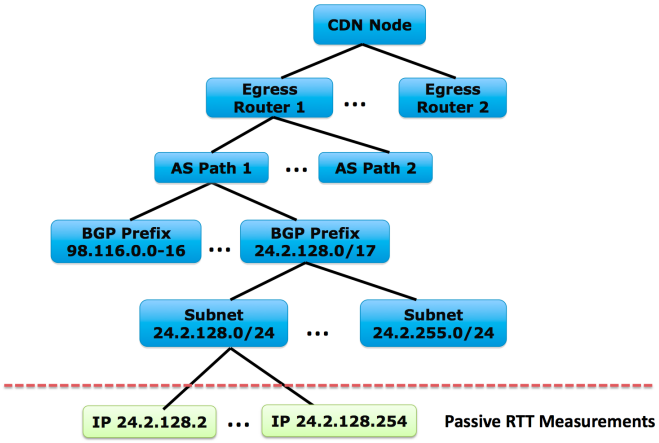

Figure 5: Topological hierarchy

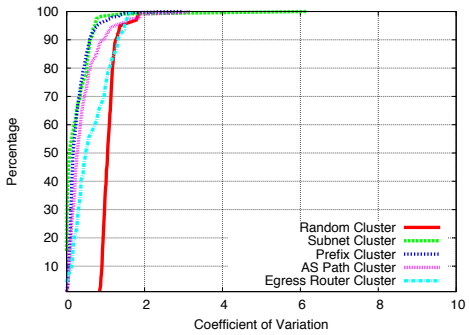

Figure 6: Distribution of coefficient of variation for clusters (Minimum is used as the key statistical indicator for each client RTT series.)

This approach immediately solves the first problem of anomaly detection for individual client IPs. More specifically, scalability is not a big issue here as there are much fewer higher level clusters we need to keep track of compared to the number of individual client IPs. Let's take Northeast dataset as an example. After aggregating, instead of monitoring several millions of client IPs, now we only keep track of 202,252 subnets, 23,869 BGP prefixes, 5,116 AS paths and several hundred egress routers.

Moreover, individual client IP level anomalies are not meaningful for isolating performance issues as operators are more interested in the network event that effects the RTTs of a large number of client IPs. For example, if most of client IPs that traverse the same AS path experienced abnormal RTTs during a time period, it is more meaningful to report a single AS path anomaly to operators compared with reporting many anomalies for individual client IPs. Due to aggregation, the anomalies are naturally reported for subnets, BGP prefixes, AS paths and egress routers. They are more useful to isolate performance issues compared to individual client IP anomalies.

3.1 Spatial Locality among Client RTT Series

In this section, we try to test an important assumption that is whether client IPs that are topologically close to each other have similar client RTT series. Only if this assumption holds, aggregating client RTT series that are topologically close to each other makes sense. Towards this end, we cluster client RTT series at different aggregation levels and examine whether client RTT series in the same cluster are similar. Specifically, for each client RTT series, one key statistical indicators such as median and minimum is extracted. Then the similarity test among client RTT series is done by using this key statistical indicators.

Here we consider the four different aggregation schemes, namely subnet aggregation, BGP prefix aggregation, AS path aggregation and egress router aggregation. We also conduct random aggregation for comparison. First client RTT series are aggregated into clusters according to different aggregation schemes. We only consider client RTT series that have at least 100 measurements to keep the computation meaningful. Then for each cluster, we calculate the median(or minimum) RTT for each client RTT series in it and the CV of these median(or minimum) RTTs. In other words, the smaller the CV is, the stronger spatial locality is.

Figure 7 and 6 plot CDF of CV for clusters using median and minimum as the statistical indicator respectively. These plots are generated based on Northeast dataset. Overall we can find from both plots that aggregations on subnet level, BGP prefix level, AS path level and egress router level all exhibit significant stronger spatial locality than random aggregation. Both plots also suggest that spatial locality is strongest in subnet level aggregation; BGP prefix aggregation and AS path aggregation show similar degree of spatial locality; egress router aggregation exhibits less significant degree of spatial locality than others. Even thought all aggregation levels in the topological hierarchy exhibit significant degree of spatial locality, we do notice the percentage of clusters that show no spatial locality or very limited degree of spatial locality increases as the aggregation level moves up in the hierarchy. For example, in Figure 7, the max coefficient of variation for random aggregation is 2.93 while there are almost 0.4% clusters at prefix aggregation level have coefficient of variation larger than 0.4%. The number for AS path and egress router aggregation levels are 0.4% and 2%. We also conduct the same experiments using Northwest and Southeast dataset and they show the similar results.

3.2 Sparsity of Aggregate RTT Series

Regarding the sparsity, we first check the distributions of number of RTT measurements at different aggregation levels. As one may expect, the number of RTT measurements at aggregation levels increases significantly compared to individual client IPs. Figure 8 shows that for Northeast dataset: only 20% client IPs have more than 10 measurements while 65% subnets, 60% prefixes, 65% AS paths and 85% egress routers have more than 10 measurements. The reason why

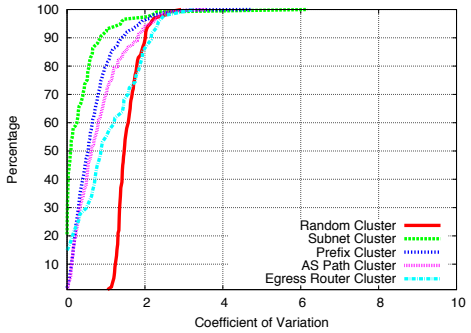

Figure 7: Distribution of coefficient of variation for clusters (Median is used as the key statistical indicator for each client RTT series.)

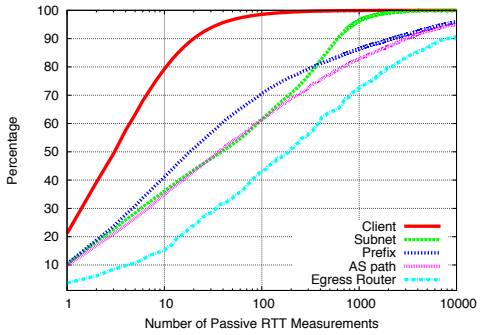

Figure 8: Distribution of number of RTT measurements at different aggregation levels

more BGP prefixes have less than 10 measurements compared to subnets is that there are many BGP prefixes have length longer than 24 (subnet) in our BGP data.

Similar to section 2.2, we further define a RTT measurement to be "valuable" only if it is at least 600 seconds later than the previous RTT measurement. Figure 9 shows that for Northeast dataset: most of clusters at all different aggregation levels have much more"valuable" RTT measurements compare to individual client IPs.

The above analysis suggests that sparsity is significantly improved at aggregation levels. We also conduct the same

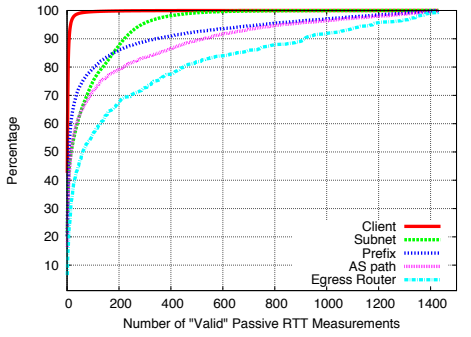

Figure 9: Distribution of number of "valuable" RTT measurements at different aggregation levels

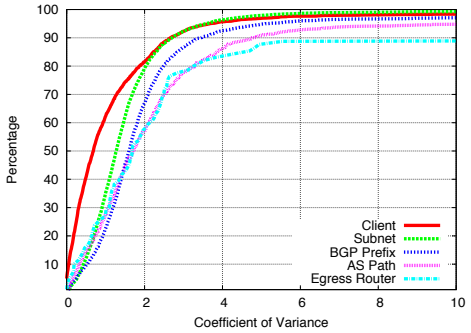

Figure 10: Distribution of coefficient of variation at different aggregation levels

experiments using Northwest and Southeast dataset and they show the similar results.

3.3 Variability in Aggregate RTT Series

The above analysis suggests that anomaly detection based on aggregated RTT series addresses the scalability and sparsity issues and make senses due to the existence of spatial locality. However, aggregating multiple client RTT series into a single aggregate RTT series will likely cause a larger variability than the variability in client RTT series as shown in Figure 2. Figure 10 shows the coefficient of variation at different aggregation levels using dataset Northeast. As we expect, as the aggregation level moves up in the hierarchy, we see a greater likelihood of larger variability. We also notice that the variability difference between AS path level and egress router is not significant.

3.4 Summary

Aggregating client RTT series along the topological hierarchy addresses the scalability issue and measurement sparsity issue It also naturally provides the ability of isolating performance anomalies due to the topological significance in the hierarchy. However, we also find aggregate RTT series have larger variabilities than client RTT series , which makes anomaly detection very challenging. We will discuss how to deal with the large variability in aggregate RTT series for better anomaly detection in the next section.

4. SONAR SYSTEM DESIGN

In this section, we describe our design of SONAR, a passive hierarchical anomaly detection and localization system. SONAR turns the individual client RTT series from an ISP's CDN service into prioritized and localized service anomaly events. SONAR operates in streaming fashion – as client RTT series stream arrives in real time, the severity and scope of on-going service anomaly events are updated.

Our design of SONAR is based upon the observation and insight we have acquired through data analyses in Section **??** and 3. Particularly, we have developed a five-stage approach that is tailored for the variability and sparsity of client RTT series.

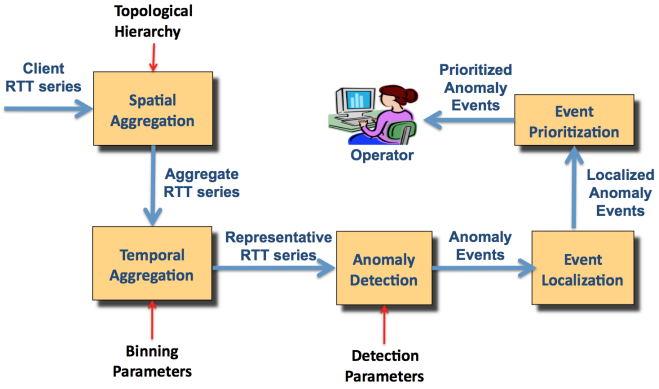

Figure 11: System architecture

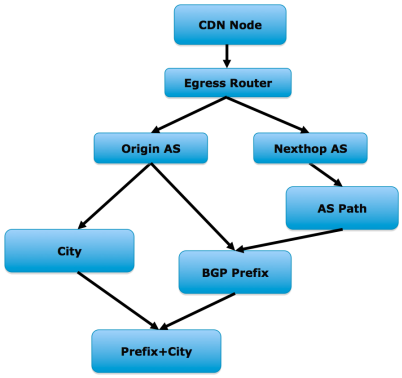

Figure 12: Topological hierarchy used in SONAR

- Spatial aggregation – group client RTT series according to the requesting client IP address into aggregate RTT series at various levels such as subnet, BGP prefix, AS path, and ISP egress router level.
- Temporal aggregation – at all levels, organize the aggregate RTT series into bins and compute a representative RTT from each bin to form a **representative RTT series**.
- Anomaly detection – use online anomaly detection scheme to extract service anomaly events from the representative RTT series.
- Event localization – localize the scope of service anomaly events
- Event prioritization – prioritize localized service anomaly events by factors such as severity, lasting duration, and impact scope.

We now describe each stage in SONAR in detail.

4.1 Spatial Aggregation

To cope with the self-inflicted RTT increase phenomena described in Section 2.1.3, we first perform an suppression on client RTT series – if multiple RTT measurements from the same client RTT series are observed within one second, only the minimum RTT value is taken. We find one second suppression window quite effective in removing the artifact due to client-side queuing while keeping the chance of false suppression (such as due to NAT) low.

We next group client RTT series according to the requesting client IP address into aggregate RTT series according to the hierarchy in Figure 12. While the mapping from IP address to city depends a static geo-location database, the mapping to BGP prefix, next-hop AS, origin AS, AS path and egress router requires dynamic correlation with BGP information. We do so by periodically collecting BGP dumps from the route reflectors co-located with the CDN servers. In our current system, the BGP dump is acquired on a hourly basis. This can be further improved using methods such as described in [20, 2].

4.2 Temporal Aggregation

In this step, we organize aggregate RTT series into bins at all levels of topological hierarchy. Binning is a classic data processing technique for data smoothing, which is much needed as demonstrated by the high variability of aggregate RTT series in Section 3.3. We use two types of binning methods – fixed size bin and fixed time bin. For fixed size bin, aggregate RTT series is divided into equal size (e.g., 100) groups of (IP level) RTT measurements. For fixed time bin, aggregate RTT series is divided into equal length (e.g., 10 minutes) groups. Comparing the two approaches, fixed time bin is more intuitive, however it is more sensitive to data sparsity – smoothing over one or a few data samples is ineffective. Fixed size bin on the other hand is more sensitive to variability due to changes in the composition of different client IPs across the Internet. In our current implementation, SONAR runs in either fixed time bin mode or fixed size bin mode, while it remains as our future work to evaluate how much benefit we can achieve to combine the two.

Once bins is formed, we compute a representative RTT value for each bin to form a representative RTT series. Several statistics can be used as the representative value – minimum, maximum, average, median or other percentile values. Different statistics may have advantage for tracking certain type of issues. For example, the minimum RTT may well capture baseline RTT due to network propagation delay while being oblivious to varying queuing delay that may be due to network congestion. The maximum or average RTT can capture poor performing individual requests performance. Since our goal is to detect general service performance issues that impact a relatively large collection of users, we pick median RTT as the representative for each bin. We find median RTT quite effective in tracking service side or network side issues while being robust to individual RTT variability due to client side processing or local access queuing delays.

4.3 Anomaly Detection

To transform the representative RTT series into anomaly events, it requires an online time series anomaly detection algorithm. In SONAR, we adopt a new **Shadow Holt-Winters (SHW)** algorithm . SHW algorithm is built based on the classic additive Holt-Winters algorithm, a widely used one-pass online anomaly detection method [9, 10, 8]. Holt-Winters algorithm has found many applications in Internet traffic analyses due to its simple yet effective model (as represented by three exponential smoothing processes).

At a high level, SHW decomposes the time series into three components: a baseline, a linear trend, and a seasonal effect. As its name suggests, SHW keeps two copies (a working copy and a shadow copy) of the three components and update each copy in parallel using different parameters (α, β and γ). Working copy gives more weight to the history compared to the recent observations and also ignores anomalies for updating itself, which are desirable for anomaly detection when the underlying RTT distribution is stable. Shadow copy gives more weight to the recent observations, so that when the RTT distribution changes (due to routing changes or some other events) it can quickly adapt to the new RTT distribution.

Specifically, upon seeing a new observation, SHW computes the deviation score of the observation from the time series forecast that is calculated from the three components in the working copy. If the observation is considered as normal, both working copy and shadow copy are updated in the same way as the classic additive Holt-Winters algorithm does. If the observation is considered as abnormal, only the shadow copy is updated. Once the number of (almost) consecutive abnormal observations exceeds some threshold (suggests that the underlying RTT distribution has changed), SHW copies the shadow copy over the working copy as shadow copy should have adapted to the new underlying RTT distribution.

As SHW has the same set of parameters as classic additive Holt-Winters algorithm, we follow the guidelines in [10] for the parameter selection and choose the ones corresponding to a low adaptability level.

For each observation, the output of SHW algorithm is a deviation score d that matches to that in a standard Guassian distribution. We descretize it into six levels (in preparation for a ranking algorithm in the next subsection). Abnormality level $\mathcal{A} = 0, 1, 2, 3, 4, 5$ when the absolute value of deviation score $|d|$ is in $[0, 0.5)$, $[0.5, 1)$, $[1, 1.5)$, $[1.5, 2)$, $[2, 2.5)$ and $[2.5, \infty)$ respectively. We consider $\mathcal{A}$ of 4 or above as anomalous. This is a relatively aggressive setting (i.e., more anomalies). However, it is an appropriate setting as our event localization and prioritization (next two stages) is robust to false positives.

We further combine consecutive anomalous bins into a single anomaly event. SONAR then keeps track of all ongoing anomaly events, with the begin time of the event being the begin time of the first anomalous bin. The anomaly events are detected and updated at all different levels of the topological hierarchy (as shown in Figure 12).

4.4 Event Localization

In SONAR, a single underlying network event such as link failure may manifest itself at different hierarchy levels. For example, if an underlying network event causes all the users from the same BGP prefix experience larger RTT, SONAR will definitely report an anomaly event for that BGP prefix and probably also report anomaly events for higher levels such as the AS path that is associated with that BGP prefix - if the users from that BGP prefix contribute a large fraction of all measurements for its AS path. In this case, we want SONAR to localize the anomaly event to the BGP prefix by only reporting a single anomaly event for the BGP prefix. Another example, if a network event has localized impact on an AS path, all its child locations at a level below (e.g., BGP prefixes) would observe the same anomalous pattern. In this case, SONAR should only report the anomaly event at the AS path. Given a set of anomalies reported at different levels in the hierarchy, our goal is to find the **smallest set** of representative anomaly events that can explain all the reported anomaly events. In the remaining of this subsection, we first formulate our event localization problem **(ELP)**, then show it is NP-hard and present a greedy solution.

4.4.1 Problem Formulation

We first introduce the notations used in the problem formulation. In ELP, the topological hierarchy (e.g. Figure 12) is a directed acyclic graph (DAG). Let N represent the set of nodes in the topological hierarchy. $\forall n \in N, D(n)$ denotes the set of n's descendants (i.e. nodes can be reached from n by traversing edges). $\forall n \in N, A(n)$ denotes the set of n's ancestors (i.e. nodes can reach n by traversing edges). $\forall n \in N, d(n)$ denotes the set of n's direct descendants (i.e. nodes can be reached from n by traversing only one edge). $\forall n \in N, a(n)$ denotes the set of n's direct ancestors (i.e. nodes can reach n by traversing only one edge). The topological hierarchy has the following two properties.

(**P1**) Each node is in one of the three status: abnormal, normal or insufficient measurements.

$$\forall n \in N : f(n) = \begin{cases} 1 & \text{if n is abnormal} \\ 0 & \text{if n is normal} \\ -1 & \text{if insufficient measurements for n} \end{cases}$$

(**P2**) Each abnormal node has at least one abnormal or "insufficient measurements" descendant.

$$\forall n \in N : f(n) = 1 \Rightarrow \exists x \in D(n) : f(x) = 1 \vee f(x) = -1$$

Objective Function:

$$\arg\min_{A \subseteq N} |A|$$

The goal of ELP is to find a smallest subset A that subjects to the following three constraints (C1-C3).

Constraints:

(**C1**) Each node in A must be abnormal.

$$\forall a \in A : f(a) = 1$$

(**C2**) Each abnormal node in N is either in A or is a descendent of a node in A or a ancestor of a node in A. In other words, all abnormal nodes are covered by the subset A.

$$\forall n \in N : f(n) = 1 \Rightarrow \exists a \in A : n = a \vee n \in D(a) \vee n \in A(a)$$

(**C3**) For any node in A, the number of its direct abnormal and "insufficient measurements" descendants is larger than the number of it direct normal descendants.

$$\forall a \in A : |\{x \in d(a) | f(x) = 1 \wedge f(x) = -1\}| > |\{x \in d(a) | f(x) = 0\}|$$

4.4.2 *Complexity Analysis*

Theorem 1. Event localization problem *is NP-hard.*

PROOF. It is easy to show that ELP $\in NP$. Given a subset $A \subseteq N$, validating that A satisfies the constraints (C1, C2 and C3) can be done in polynomial time.

To show that ELP is NP-hard, we prove that set-overing problem [14] is polynomially reducible to ELP. i.e. set-covering $\leq_p$ ELP. Given an instance of set-overing problem with the universe $\{u_1, u_2, ..., un\}$ and a family of subsets $\{s_1, s_2, ..., sm\}$ of the universe, we can construct an instance of ELP as follows:

- Create a two-level topological hierarchy with m abnormal nodes at the bottom level (one for each subset) and n abnormal nodes at the top level (one for each element in the universe).

- Add a directed edge from one abnormal node u_i at the top level to one abnormal node s_j at the bottom level if u_j is a member of S_i.

- For each one abnormal node u_i at the top level, add the same number of normal nodes as the number of sets that u_i belongs to at bottom level.

- At the top level, add a common ancestor abnormal node M that connects to all abnormal nodes $\{s_1, s_2, ..., sm\}$ at bottom level.

- At the bottom level, add one special abnormal node L whose ancestor is M.

Figure 13 shows how the polynomial reduction is done from a set-overing instance to an ELP instance.

We claim that our resulting construction of EPS instance has a feasible solution A with size at most $k+1$ if and only if the original set-covering instance is satisfiable with at most k subsets. Indeed, if the set-cover is satisfiable with k subsets, then all abnormal nodes (i.e., the universe $\{u_1, u_2, ..., un\}$ in the set-covering) at the top level can be covered by k abnormal nodes (i.e., k subsets in set-covering) at the bottom level. The remaining abnormal nodes at the bottom, including a special abnormal node L, can be covered by the common ancestor abnormal node M at the top level. Conversely, suppose our ELP instance has a feasible solution A with size $k+1$. Then, A must include i) the common ancestor node M to cover all the abnormal nodes at the bottom level, and ii) k abnormal nodes at the bottom to cover all the top level abnormal nodes (note: those abnormal nodes cannot be included in A, as each of them has normal children). By selecting the subsets each of which corresponds an abnormal node at the bottom in our EPS solution A, the set cover instance is satisfiable with a collection of k subsets. Since this reduction is in polynomial time and the set-overing problem has been shown to be NP-hard, the event localization problem is NP-hard[2]. This completes the proof. □

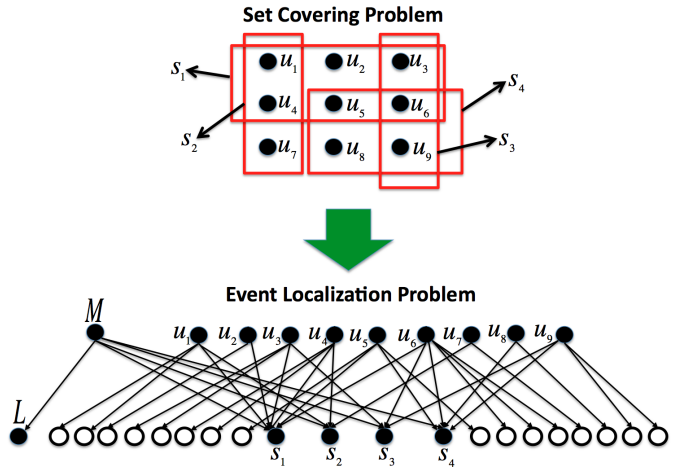

Figure 13: Polynomial Reduction Example

4.4.3 *A Greedy Solution*

Using the set-covering terminology, all the abnormal nodes in the topological hierarchy form the **universe**. By picking an abnormal node x that satisfies the constrains **C1** and **C3**, a **subnet** S_x ($S_x = x \cup D(x) \cup A(x)$) of the universe is formed. For each event localization problem instance, there is a family of n subsets $(S_1, S_2, ..., S_n)$ that are corresponding to n abnormal nodes that satisfy the constrains **C1** and **C3**. The goal of event localization problem is to find the smallest subfamily from whose union is the universe. [14] presents a simple greedy algorithm for it and proves the greedy algorithm is a factor-$\lceil \ln n \rceil$ approximation algorithm. Here we discuss a similar greedy algorithm 1 to the event localization problem, which keeps choosing the abnormal nodes (subsets in set-covering problem) that covers most uncovered abnormal nodes (elements in universe in set-covering problem) until all abnormal nodes (the whole universe in set-covering problem) are covered.

4.5 Event Prioritization

[2]If the topological hierarchy is a tree, then the event localization problem is a P problem. In this special case, one can simple do a breadth-fist search. Once a node that satisfy the constrains is found, one can immediately select it into output subset A and stop searching its descendants.

Algorithm 1 : Greedy Algoritm for Event Localization Problem

Let A denote the output subset
Initialize $A = \emptyset, UNCOV = \{x \in N | f(x) = 1\}$
for each $u \in UNCOV$ **do**
  $SET_u = u$
  for each $v \in D(u)$ **do**
    if $f(v) = 1$ **then**
      $SET_u = SET_u \cup v$
    end if
  end for
  for each $w \in A(u)$ **do**
    if $f(w) = 1$ **then**
      $SET_u = SET_u \cup w$
    end if
  end for
end for
while $UNCOV \neq \emptyset$ **do**
  Choose $u \in UNCOV$ such that $|SET_u|$ is maximized
  $A = A \cup u$
  $UNCOV = UNCOV - SET_u$
  for each $i \in UNCOV$ **do**
    $SET_i = SET_i - SET_u$
  end for
end while

After event localization stage, SONAR employes a ranking function to prioritize the localized anomaly events. The ranking function incorporates two factors – the significance of the relative size of the anomaly and the breadth of its impact scope. The former can be measured by the deviation score $|d|$ from Holt-Winters algorithm. The latter can be measured by the the number of distinct client IP addresses observed in the anomalous bin, which we denote as c. We choose distinct client IP addresses (as opposed to total request counts) since it is robust against anomalies dominated by a spike of requests from a few outlier clients. Since each anomaly event may contain multiple anomalous bins, we use the aggregate score of all bins for the score of the event. Specifically, for anomaly event e, its baseline ranking score r_e is defined as:

$$r_e = \sum_{b \in \text{ bins of } e} |d_b| \times c_b$$

where d_b and c_b is the deviation score and distinct IP count for bin b. In this way, long lasting events are likely given higher priority than short events.

4.6 Summary

Figure 11 summarizes the five stages involved in SONAR pipeline. As client RTT series stream arrives in real time, first they are grouped into many aggregate RTT series at different levels according to the topological hierarchy and each of these aggregate RTT series is converted into a smooth representative RTT series. Then service anomaly events are detected at different levels by running shadow Holt-Winters with the representative RTT series. Finally detected anomaly events are localized using our greedy heuristic and prioritized based on their severity and impact scope. All these stages operate in streaming fashion, which means events start with some initial RTT measurements and evolve in terms of priority and duration as more and more RTT measurements arrive. SONAR presents all on-going anomaly events with their priority and duration so that operators can keep track of them.

5. EVALUATION

Evaluating the accuracy (e.g., false positive and false negative) of an anomaly detection system needs a list of ground-truth anomaly events. However, ground truth is difficult to find in our context, as is typically the case in anomaly detection studies. Instead we cross check the anomalies detected by SONAR with two independent anomaly data sources for the same network:

- anomaly events detected by using RTTs actively measured from Keynote[4] agents
- authoritative DNS server change events provided by the CDN service team

Thus our evaluation can determine how many of the events from these two data sources are reported by SONAR, how many of them are missed by SONAR and investigate the reasons for the differences. In the remainder of this section, for each data source, we first describe the data source and then outline the evaluation results by using it. All timestamps in the section are in GMT.

5.1 Data Source 1: Keynote Anomaly Events

Each Keynote [4] agent machine has 3 unique IPs in the same /24 subnet, and each IP periodically sends probe packets to CDN servers in order to measure end-to-end performance. For the North-East CDN node, Keynote has six agents from different geo-locations and ISPs as shown in Table 2. While Keynote agent keeps track of many different metrics such as DNS lookup time, first byte download time and base page download time, we are only interested in the initial connection time in order to make direct comparison with the passively measured RTTs in SONAR. Initial connection time in Keynote measures the RTT between the Keynote agent and the server by calculating the time difference between the first SYN (from Keynote agent to CDN server) and the SYN+ACK (from CDN server back to Keynote agent) during the TCP handshake.

Agent ID	Country	State	City	ISP
1	US	MA	Boston	Sprint
2	US	MA	Boston	Verizon
3	US	CT	Hartford	Same ISP
4	CA	QC	Montreal	Peer1
5	US	NY	New York	Cogent
6	US	NY	New York	Sprint

Table 2: Locations of six Keynote agents

Note that Keynote does not directly provide the anomaly data. Therefore, for each of the six Keynote agents (each with 3 unique IP addresses in the same /24 subnet), we first collect the initial connection time over a two-month period (from 1st July 2010 to 31st August 2010), and group them into 10-minute bins. Then we run the Shadow Holt-Winters algorithm to detect anomaly events (called **Keynote anomaly events** in the rest of the section) for comparison with the anomalies detected by SONAR for the same time period and with the same bin mode and size.

5.1.1 Results

There are totally 310 anomaly events from all six Keynote agents during the two-month period. We compare Keynote anomaly events with the results regarding their corresponding /24 subnet reported by SONAR, and classify the results as follows.

- *Match*: SONAR reported an event on the same subnet that is temporarily overlapped with the Keynote anomaly event.
- *Miss*: SONAR didn't report anything for the same subnet around the period of the Keynote anomaly event.
- *More*: SONAR reports anomalies about a Keynote agent's subnet, but there are no temporally overlapped Keynote anomalies event

As shown in Table 3, SONAR in total successfully detected 91% (281 events) out of 310 Keynote anomaly events, missed 29 events (9%), and detected 51 more anomalies. As also shown in the table, SONAR observes RTT for more IPs in the subnet covering each Keynote agent subnet (each has 3 IPs). For example, in Keynote agent 2's subnet, in addition to the 3 IPs used by Keynote for probing, another 10 IPs also visited the North-East CDN node during the two-month period. As a result, even though an anomaly event is detected using keynote measurements from the 3 IPs, it may not show up in SONAR as a subnet anomaly by looking at the RTT measurements from all 13 IPs. Also, SONAR detects 20 more anomalies for the agent 2 subnet. These are the expected difference between SONAR and active measurement-based anomalies. Coverage of more IPs can detect anomalies not directly experienced by the probing IPs, and also anomalies experienced only by the probing IPs might not contribute enough to the subnet to have subnet level anomalies.

Agent	# IPs from its subnet seen by SONAR	# Keynote Anomaly Events	# Match	# Miss	# More
1	3	44	38	6	0
2	13	93	85	8	20
3	5	32	29	3	1
4	7	49	45	4	3
5	6	72	67	5	4
6	15	20	17	3	23

Table 3: Comparison with Keynote anomaly events

5.2 Comparison with Authoritative DNS Server Change Events

We first describe how CDN node assignment works in the tier-1 ISP and then how the authoritative DNS server change helps with our evaluation.

The tier-1 ISP CDN's authoritative DNS server, when receiving a DNS query from the client IP's local DNS server, responds with a CDN server address closet to the client's local DNS server, with the hope that this server address is also closet to the client IPs. As with most of DNS-based CDNs [1, 5], this approach assumes that the network location of a client IP can be approximated by that of its local DNS server since the authoritative server can see the local DNS server address, but not the client IP's address. In the tier-1 ISP CDN, all these authoritative DNS servers, regardless its location, assign the same CDN node for the same local DNS server address. This global assignment table can be adjusted over time.

The tier-1 ISP CDN uses anycast IP addresses for their authoritative DNS servers (i.e.,they have the same IP addresses) located at many PoPs in the ISP network. Therefore, a local DNS server's DNS query is naturally routed to the closed authoritative DNS server. The CDN service team monitors the local DNS server addresses that queried each authoritative DNS server in their daily operations. Since the authoritative DNS servers are located near the edge of the ISP network, a change in the authoritative DNS server 'hit' by a local DNS server is indicative of the routing from the local DNS server to the authoritative DNS server anycast address has changed its ingress point to the tier-1 ISP. And because the tier-1 ISP announces the same BGP path attributes for the anycast prefix and the CDN node prefix, the routing path from local DNS server (and also the client IPs) to the assigned CDN node might also (but not always) change its ingress point to the tier-1 ISP.

For example, if a local DNS server that was 'hitting' an authoritative DNS server in Boston PoP starts hitting an authoritative DNS server in Chicago PoP, it is likely the client IPs that local DNS server serves now enter the ISP network via Chicago PoP instead of Boston PoP, to reach the North-East CDN node and result in a longer RTT. Note that the returning path from the North-East node to the client IP might or might not change since sometimes the routing is asymmetric, but when the routing of this direction also changes, e.g., going through Chicago POP as well, the RTT increase will be even larger. In other words, a local DNS hit change at the authoritative servers is a good (although not perfect) indication of the RTT increase experienced by the client IPs, and we obtained a list of such change events from 1st April 2010 to 15th April 2010 for all the local DNS IPs that should be assigned to the North-East CDN node according to the global assignment table. Note that this list was manually compiled and may not be comprehensive.

5.2.1 Results

Using these labeled DNS events as an independent data source, we evaluate the accuracy of SONAR by running it over the same period. Specifically SONAR runs in fixed time bin mode with bin size as 10 minutes based on the RTT measurements passively collected at the North-East CDN node.

In this evaluation, we first extract the AS path of the local DNS server from each labeled event in the list and then classify each labeled event into three categories based the output of SONAR.

- *Match*: SONAR reported an event on the same AS path that is temporally overlapped with labeled event. The top plot in Figure 14 shows an example of "Match" labeled event. There is a labeled event from 3:45am to 6:00am on 8th April 2010 to for a local DNS server that uses AS path "3356 6079" to reach the ISP. The local DNS server was routed to an authoritative DNS server close to the North-East CDN node before 3:45am on 8th April 2010 and after that it was routed to another authoritative DNS server several hundred miles away from the North-East CDN node till 6:00am on 8th April 2010. It is a "match" as SONAR reported an event around the same period of the labeled event.

- *No Data*: As there were no passive RTT measurements on the AS path around the period of the labeled event, SONAR didn't report anything. The middle plot in Figure 14 shows an example of "No Data" labeled event as there is not measurement data around its period. That means during the period of labeled event, no traffic is seen on this AS path. We are still working with CDN service team to figure out the reason of this while it is not the focus of this paper. We don't worry about this case as SONAR didn't report anything simply due to lack of data.

- *Miss*: There were passive RTT measurements on the AS path around the period of the labeled event but SONAR didn't report anything. The bottom plot in Figure 14 shows an example of "Miss" labeled event. One plausible explanation, as we discussed earlier, only the local DNS server experienced a routing change to the authoritative server, but the clients' routing to the CDN nodes were unaffected. In other words, this provides an upper bound of the "ground truth" anomalies (related to these hit changes) missed by SONAR, and the actual SONAR performance can be better.

The duration of the 68 labeled events in the list varies from 15 minutes to 5 hours. We show the evaluation results by breaking down the 68 events into 3 classes according their durations. As you can see from table 4, out of 68 labeled events, SONAR successfully detected 72% (49 events) and missed 13%. The rest 15 % is hard to decide due to lack of data. In addition, SONAR tends to miss more short ($15m \leqslant D \leqslant 1h$) labeled events.

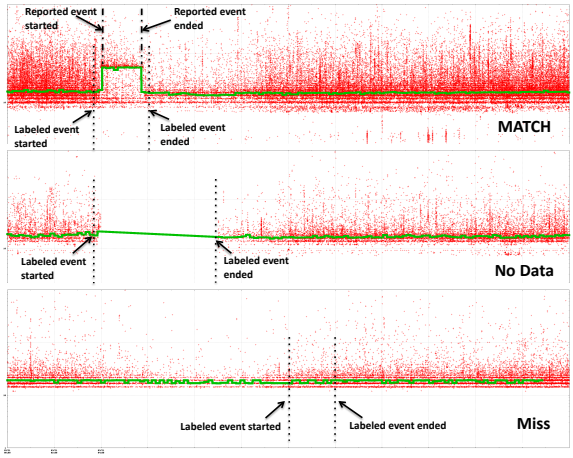

Figure 14: Examples of three categories of labeled DNS events

Duration (D)	# Labeled Events	# Match	# No Data	# Miss
$15m \leqslant D \leqslant 1h$	24	16	2	6
$1h < D \leqslant 2h$	32	25	5	2
$D > 2h$	12	8	3	1

Table 4: Evaluation results using authoritative DNS server change events

6. OPERATIONAL EXPERIENCE

In this section, we first summarize the results of running the SONAR system in a tier-1 ISP and then discuss two representative case examples in detail. This ISP is referred to as local ISP in the remaining of this section.

6.1 Overall Results

In order to have a basic understanding on how SONAR works, we focus on the anomaly events detected by SONAR from 20th July 2010 to 20th August 2010. Note these anomaly events were detected by running SONAR in fixed time bin mode with bin size as 3600 minutes based on the RTT measurements passively collected at the North-East CDN node.

During this one-month period, SONAR detected 2,909 anomaly events across all the levels in the topological hierarchy (Figure 12). Table 5 shows the anomaly event distribution across all hierarchy levels. In general, the lower level is responsible for more anomaly events as the lower level tends to have more elements. In addition, at each level only a small fraction (bad elements) of all elements are responsible for the anomaly events. As shown in Figure 15, generally there is no heavy hitter among the bad elements.

After analyzing the anomaly events' spatial characteristics, now we try to understand the time durations of the 2,909 anomaly events. Figure 16 clearly shows that the short events are common and long-lasting events are rare. Specifically, the events with duration 3600 minutes are the most com-

Hierarchy Level	# Total Elements	# Events	# Bad Elements
CDN Node	1	0	0
Egress Router	185	66	36
Nexthop AS	125	54	25
Origin AS	820	172	97
AS path	1055	213	119
City	1758	593	289
BGP prefix	4646	600	425
City+BGP prefix	8784	1211	851

Table 5: Anomaly events breakdown by hierarchy level

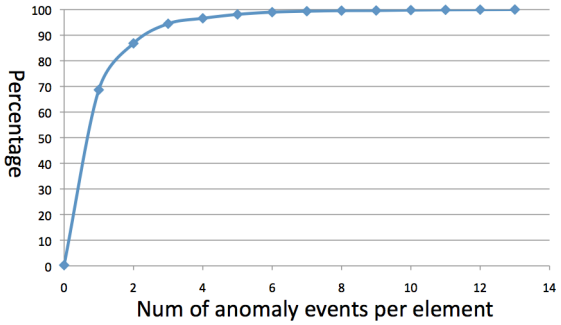

Figure 15: Distribution of number of anomaly events per element across all levels in the hierarchy

mon one and contribute 90% of all anomaly events. In addition, this statement holds true for every level in the hierarchy. Due to space limitation, we don't show the per-level distribution here. Note 3600 minutes is the minimal duration of an anomaly event as SONAR is running in fixed time bin mode with bin size as 3600 minutes.

As the final step towards understanding overall results, we dig into the detail of the top 100 anomaly events. Top 100 events were identified by SONAR using the ranking function describe in Section 4.5. Specially, we try to find the root cause of these events by spatially and temporally correlating them with the underlying network events inside the local ISP. Figure 17 shows the root causes of the top 100 anomaly events. 13 of them are confirmed to be caused by the events inside the local ISP such as BGP routing change, link failure or link congestion. For the rest (majority) of them, we couldn't find any evidence inside the local ISP, which suggests that they may be caused by other ISPs.

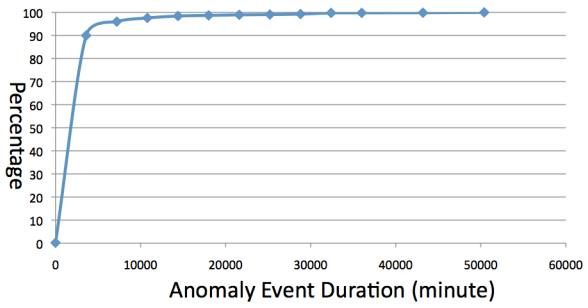

Figure 16: Distribution of anomaly event duration across all levels in the hierarchy

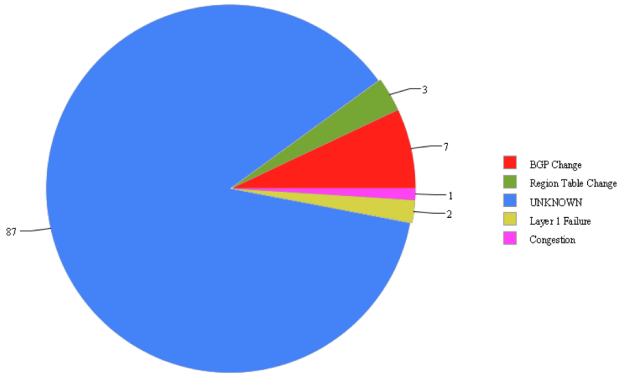

Figure 17: Root causes of the top 100 anomaly events

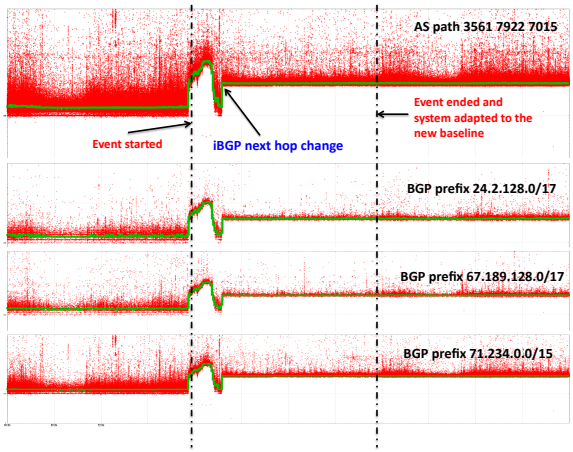

Figure 18: Aggregate RTT series, representative RTT series and the reported one-day event for AS path "3561 7922 7015" from 7th May 2010 to 9th May 2010.

6.2 Case Stuides

The following two case examples demonstrate the effectiveness of SONAR for detecting and localizing performance issues that effect the tier-1 ISP's CDN service. Both case examples are based on the RTT measurements passively collected at the North-East CDN node and fixed time bin mode is used with bin size as 10 minutes. All timestamps in the section are in GMT.

Case Example 1: Permanent RTT Level Shift Caused by iBGP Changes in the Local ISP.

SONAR reported an one-day event on AS path"3561 7922 7015" [3] from 11pm on 7th May 2010 to 11pm on 8th May 2010. We were interested in looking into this event as it ranked first among all events from 1st May 2010 to 10th May 2010. The AS path"3561 7922 7015" has 7 children (BGP prefixes) in total. Each child is a BGP prefix that

[3]This is the reverse AS path that CDN nodes use to reach clients.

can be reached via this AS path from the perspective of the local ISP. It turned out that these 7 BGP prefixes were experiencing the exact same event as the AS path around the same time. The highest ranking score of this event comes from three aspects. It spans 144 ten-minute intervals, each of which has a high "confidence" (several hundred unique clients) and a significant "severity"(2 to 4 standard deviations from predicted value). Moreover, because of the high anomaly prevalence among all its 7 children around the period of this event, the ranking score is not discounted at all.

The top plot in Figure 18 shows the AS path level aggregate RTT series and representative RTT series. We can see clearly that there was a RTT level shift started around 3:30am on 8th May 2010. It is a permanent RTT level shift as it didn't change back to the previous baseline. The period of the event is aslo marked by two vertical dashed lines in Figure 18. Note in SONAR one day is the maximum length of an event and SONAR will adapt to the new baseline if an event lasts longer than one day. The bottom three plots of Figure 18 show three BGP prefixes under AS path "3561 7922 7015" in the hierarchy. Due to space limitations, we omit the other 4 BGP prefixes that exhibited exact the same behavior as the three shown in Figure 18.

According to the BGP updates from a route reflector that is co-located with the North-East CDN node, we found iBGP next hop to reach all the 7 BGP prefix changed at 3:28am on 8th May 2010 from the same city as the North-East CDN node to another city 1000 miles away from it. Thus, iBGP next hop change is very likely to be the reason of RTT level shift in Figure 18 and the bump before the level shift may indicate some network instabilities that caused the iBGP next hop change.

Case Example 2: Temporary RTT Level Shift Related to Remote ASes.

Among all events in 2010 March reported by SONAR, the top first is an event on AS path "7922 7015" from 4:10am to 5:50pm on 21st May 2010. This event is so significant as it effected all 65 BGP prefixes on the AS path "7922 7015" for more than 10 hours. The top plot in Figure shows the aggregate RTT series and representative RTT series for AS path "7922 7015". The event reported by SONAR is marked by the two vertical dashed lines.

More interestingly, SONAR didn't report anything for other two AS paths "7922 33287" and "7922 33657" that share the same *next-hop AS* and *egress router* with the problematic AS path "7922 7015" from the view of North-East CDN node. The two plot at the bottom in Figure show that the corresponding aggregate RTT series and representative RTT series for AS paths "7922 33287" and "7922 33657". Even though these two AS paths didn't have as many RTT measurements as the problematic AS path, we can still infer that the RTT level shift might be caused by something in remote ASes 7922 or 7015. The limitation of the above inference is that SONAR doesnt incorporate forward AS path information that clients use to reach CDN nodes. For example, suppose AS 7015 is multi-homed to AS 7922 and another AS X. If AS X is used for clients in AS 7015 to reach the North-East CDN node. Then the problem may be in AS X. No information in SONAR is able to identify this possible cause. Incorporating forward path information into SONAR forms part of our future work.

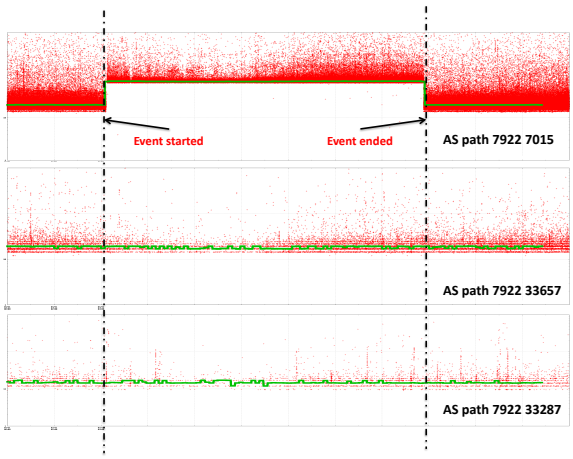

Figure 19: Aggregate RTT series and representative RTT series for AS path "7922 7015" and other 2 AS paths that share the same *next-hop AS* and *egress router*. The period of this plot is the entire day of 21st May 2010.

7. RELATED WORK

There has been extensive prior work on characterizing, detecting and isolating the end-to-end performance issues. Broadly they can be classified into three categories: active, passive and hybrid active/passive.

Active approach requires the injection of probe packets into the network. The pioneering active approach[18] treats the Internet as a complete black box and end- to-end traceroutes between 37 participating sites are collected and analyzed to characterize the end-to-end performance issues. Similar to [18], [12] detects path outage among hosts using pings and isolate the observed outage to a specific location using traceroutes. [23] focuses on identifying routing disruptions inside an ISP network use active probing (traceroutes) from end hosts and evaluating the impact of routing disruptions on end-to-end path performance such as latency. Commercial network monitoring service such as Keynote [4] and Gomez [3] are also available to detect the end-to-end performance from the end-users perspective in real time by active probing.

Passive approach purely depends the existing traffic in the network. Our approach belongs to the category. [11] studied how network failures affect the availability of end-to-end wide-area service using several large-scale traceroute datasets [18, 19]. [7] analyzed the spatial and temporal sta-

bility of end to end throughput using traceroute data from the 1996 Olympic Games Web site. They are similar as both of them inferred the end-to-end characteristics based on static traceroute data. In contrast to them, we built an online system that detects and isolates end-to-end RTT anomalies for a wide-area service in real time based on the passive measurement stream. A more recent work [6] proposed to push end-to-end performance monitoring to the end systems themselves and implemented such a prototype system based on BitTorrent. Specifically, first each BitTorrent peer detects its local events individually based on some passively measured performance metric and then local events from different peers are correlated spatially and temporally to determine the scope of the problem. The effectiveness in [6] actually depends the sparsity of passive measurements for individual end systems. An end system with very few measurements would be able to detect event effectively. The deployment is another issue as there is no strong incentive for end systems to corporate. Different from [6], our approach collects the end-to-end performance measurements corresponding to individual end systems at server side, aggregates them along the topological hierarchy, detects events at different levels and finally isolate the performance issue to the right level. Unlike [6], our approach is not limited by the sparsity of measurements for individual end systems and by the deployment issue.

Hybrid approach typically uses passive monitoring to detect performance issue and employes active probing to help isolate the issues. [15] first identifies the inflated prefixes by passively measuring the RTT between clients and Google's CDN servers and then uses traceroutes to Identifying causes of latency inflation. Our approach is similar to [15] as we also passively measure the RTT between clients and CDN servers. [15] focuses on detecting and diagnosing clients which experience latencies much higher than other clients in the same region. In contrast to [15], our approach monitors the temporal change in RTT of individual clients and detect anomalies according to the temporal changes. Another related work PlanetSeer [21] detects possible Internet path failures by passive monitoring the clients of a CDN service deployed on PlanetLab and relies on active probes to narrow down the scope of path failure. Compared to [21], our approach focuses on the temporal change of end-to-end performance and doesn't use active probing. [17] is a network tomography approach that infers the internal network characteristics based on end-to-end observations. [17] is classified as a hybrid approach because observations are obtained by passive monitoring the busy microsoft.com web site and traceroute is used from server to clients to construct the topology. [17] identifies the lossy links in the topology constructed above while our approach aims to isolate problems to the right level in the hierarchy.

8. CONCLUSION AND FUTURE WORK

Detecting and isolating end-to-end performance issues in wide-area services is critical for ISP operators. This paper argues that the most effective way to detect and isolate end-to-end performance issues in a wide-area service is to passively monitor the client IPs from inside the ISP network. However, by analyzing RTTs passively measured from a CDN service hosted and managed by a tier-1 ISP, we found that client IP level passive detection and isolation is not trivial. To address the challenges of client IP level detection and isolation, we developed PHADIS, a novel approach that aggregates client RTT series along a topological hierarchy. PHADIS has been successfully deployed in a tier-1 ISP and proved to be effective and accurate in detecting and isolating end-to-end performance issues.

Our future work can be divided into two directions. First, we plan to extend the hierarchical aggregation along more axes. For example, now only the information (reverse AS path and egress router) along the direction from the CDN node to the client IP is used for aggregation. We plan to aggregate the client RTT series using the information (forward AS path and ingress router) along the another direction (from the client IP to the CDN node) In addition, we also plan to do aggregation by individual ASes in the AS path in order to be able to detect performance issues localized to individual ASes. Secondly, we plan to monitor and detect changes in the topological hierarchy. For example, a client IP used to be reached via "AS path 1" from a CDN node now is reachable via "AS path 2". In the hierarchy, this client IP used to be a child of "AS path 1" and now is a child of "As path 2". If many client IPs change their parents in the hierarchy, this may indicate some network event and the service may be need to react to the event. Overall, we believe the PHADIS passive monitoring approach can continue to be refined and approved. This system continues to be in use and is being expanded to cover other services as well.

9. REFERENCES

[1] Akamai, inc. website. `http://www.akamai.com/`.

[2] Bgpmon project website. `http://bgpmon.netsec.colostate.edu/`.

[3] Gomez, inc. website. `http://www.gomez.com/`.

[4] Keynote systems, inc. website. `http://www.keynote.com/`.

[5] Limelight, inc. website. `http://www.LimelightNetworks.com/`.

[6] Using the crowd to monitor the cloud: Detecting network events from edge systems. `http://www.aqualab.cs.northwestern.edu/projects/NEWS.html`.

[7] H. Balakrishnan, M. Stemm, S. Seshan, and R. Katz. Analyzing stability in wide-area network performance. *ACM SIGMETRICS Performance Evaluation Review*, 25(1):2–12, 1997.

[8] P. Barford, J. Kline, D. Plonka, and A. Ron. A signal analysis of network traffic anomalies. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 71–82. ACM, 2002.

[9] P. Brockwell and R. Davis. *Time series: theory and methods*. Springer Verlag, 2009.

[10] J. Brutag. Aberrant behavior detection and control in time series for network monitoring. In *Proceedings of the 14th Systems Administration Conference (LISA 2000)*.

[11] B. Chandra, M. Dahlin, L. Gao, and A. Nayate. End-to-end WAN service availability. In *Proceedings of the 3rd conference on USENIX Symposium on Internet Technologies and Systems-Volume 3*, page 9. USENIX Association, 2001.

[12] N. Feamster, D. Andersen, H. Balakrishnan, and M. Kaashoek. Measuring the effects of internet path faults on reactive routing. In *Proceedings of the 2003 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, page 137. ACM, 2003.

[13] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee. Hypertext transfer protocol–HTTP/1.1, 1999.

[14] D. Johnson. Approximation algorithms for combinatorial problems*. *Journal of Computer and System Sciences*, 9(3):256–278, 1974.

[15] R. Krishnan, H. Madhyastha, S. Srinivasan, S. Jain, A. Krishnamurthy, T. Anderson, and J. Gao. Moving beyond end-to-end path information to optimize CDN performance. In *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, pages 190–201. ACM, 2009.

[16] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling TCP throughput: A simple model and its empirical validation. In *Proceedings of the ACM SIGCOMM'98 conference on Applications, technologies, architectures, and protocols for computer communication*, page 314. ACM, 1998.

[17] V. Padmanabhan, L. Qiu, and H. Wang. Server-based inference of Internet performance. In *IEEE INFOCOM*. Citeseer, 2003.

[18] V. Paxson. End-to-end routing behavior in the Internet. *ACM SIGCOMM Computer Communication Review*, 36(5):56, 2006.

[19] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson. The end-to-end effects of Internet path selection. *ACM SIGCOMM Computer Communication Review*, 29(4):289–299, 1999.

[20] H. Yan, R. Oliveira, K. Burnett, D. Matthews, L. Zhang, and D. Massey. BGPmon: A real-time, scalable, extensible monitoring system. In *Cybersecurity Applications and Technologies Conference for Homeland Security (CATCH)*, 2009.

[21] M. Zhang, C. Zhang, V. Pai, L. Peterson, and R. Wang. PlanetSeer: Internet path failure monitoring and characterization in wide-area services. In *Proc. USENIX OSDI*, 2004.

[22] Y. Zhang. *Characterizing End-to-End Internet Performance*. PhD thesis, Citeseer, 2001.

[23] Y. Zhang, Z. Mao, and M. Zhang. Effective diagnosis of routing disruptions from end systems. In *Proceedings of USENIX NSDI*, volume 8.