# Rot at the Roots? Examining Public Timing Infrastructure

Kanthaiah Vijayalayan*, Darryl Veitch†
*Melbourne School of Engineering, University of Melbourne, Australia. Email: pvijayalayan@gmail.com
†FEIT, University of Technology Sydney, Australia, Email: Darryl.Veitch@uts.edu.au

***Abstract*—Timekeeping is central to network measurement. In typical systems, its accuracy is ultimately dependent on the forest of timeservers accessible over the network, whose roots are the stratum-1 timeservers, which benefit from reference hardware. It is essential that these servers are accurate and reliable, and it is commonly assumed that this is the case. We put this belief to the test through an examination of around 100 publicly accessible stratum-1 servers, using datasets spanning over 3 years, collected in a testbed with reference timestamping. We develop a methodology capable of disambiguating the effects of routing changes, congestion related variability, and server anomalies on timestamps. We use it to make a first assessment of the health of (public) network timing, by reporting on the type, severity, and frequency of anomalies we encounter.**

## I. Introduction

Timekeeping is a vital service provided by computer operating systems, essential for many kernel services and user space applications, and central to network measurement. The operating system's *system clock* is software built on local hardware, which is synchronized through timestamp exchange, via the Network Time Protocol (NTP), to a reference timeserver over the network. For scalability, timeservers are organised in a hierarchy, where a *stratum* $s$ timeserver itself synchronizes to a *stratum* $s$–1 server. Anchoring the system are the *stratum-1* timeservers, which have reference hardware such as GPS or atomic clocks locally attached. These roots of the timing forest 'hierarchy' can be PCs, or purpose built networked devices [1].

As stratum-1 servers are the gold references of the system, they are typically assumed to be highly reliable and accurate (nominally to $10\mu$s). If this were not true, the impact would be potentially significant, since a good proportion of the world's computers are synchronized, ultimately, through accessing public stratum-1 servers. These are hosted typically by institutions such as NIST and USNO (USA) and NMI (Australia), research institutes and some universities.[1]

The underlying issue here is: if a server's clock were to misbehave, how would one know? In practice stratum-1 servers act essentially as independent islands. There is support through NTP to compare internal diagnostics of server synchronization quality to select a preferred peer and to inform clients, but the performance of this mechanism is variable and does not in any case constitute independent validation. From the perspective of the typical client, judging its server is inherently problematic. Not only is the client's clock designed to trust and follow that of its server, the transported reference timestamps are viewed through the shroud of network latency. This can be highly variable, and anything from tens of microseconds to hundreds of milliseconds and beyond in size, depending on the minimum round trip-time (RTT) between the client and server, and path congestion levels.

It follows from the above that, even if one were looking for them, discrepancies between the client and server clocks can easily be attributed to network latency, and/or shortcomings of the client synchronization daemon/algorithm. If one is **not** looking for them, then only errors which are huge compared to the nominal $10\mu$s are likely to be noticed. In fact, suspicion is unlikely to be directed toward a stratum-1 unless it is simply unreachable, or the error gross, for examples minutes or hours.

Our broad goal is to discover what is actually going on in the public timing infrastructure, beginning with its stratum-1 heart. From discussions with owners of certain public stratum-1 servers, we know that not only do accuracy and reliability issues exist, but these are often not easily detected even internally. The aim of this paper is to develop an *external* methodology capable of detecting issues with a server, and to use it to shine a light on the public stratum-1 servers.

Our findings are based on experiments employing a client machine, located within a timing testbed in Australia, to exchange timestamps in parallel to over 100 stratum-1 servers spread globally. We study two datasets, each months long, collected over 3 years apart to the same server set. Using independent GPS synchronized DAG card monitoring [3], we are able to disambiguate events observed at the client in network latency, from server anomalies. Our findings include many examples of very significant errors occurring on a regular basis, a wide variety of anomaly types, and widely varying anomaly amplitudes.

Because of the single client vantage point, the large RTTs involved, and the limitations both of our current detector and manual inspection, we are not capable of detecting anomalies in all cases where they may exist. Even so, the findings are significant, and strongly suggest that there may be many other errors which, though smaller, are still well beyond the nominal $10\mu$s we might expect of a stratum-1. This paper represents a substantial first look at these datasets, and is the first step in plans for a more definitive study, described in more detail in the discussion. We know of no prior work examining the health of commonly used servers, in particular public stratum-

[1]The interesting question of exactly *how* large this proportion is has become much harder to answer following the blocking of diagnostic NTP queries since late 2013, due to their exploitation in DDOS attack amplification [2].

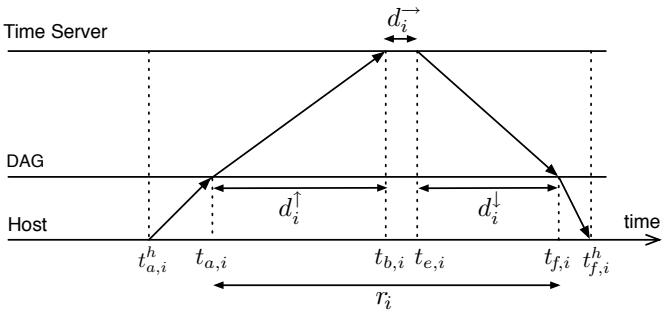


Fig. 1. Timeline of the $i$th host-server exchange including forward path delay from DAG ($d_i^{\uparrow}$), server delay ($d_i^{\rightarrow}$), backward path delay from DAG($d_i^{\downarrow}$) and RTT relative to DAG ($r_i = d_i^{\uparrow} + d_i^{\rightarrow} + d_i^{\downarrow}$).

1 servers, similar to what we present here. The closest related work is [4], which noted errors in a number of public servers as part of a survey, using a testbed of stratum-1 servers, and [5], which provides a simple survey and compares against others, including [6].

The paper is organised as follows. Section II describes our testbed and its operation. Section III describes the experiment, the selection of servers, and how the datasets were collected. Section IV outlines the principles, challenges and methodology of our server anomaly analysis. Section V details and discussed our findings. We conclude in Section VI.

## II. Testbed

The key event times in the timestamp exchange between a host and its timeserver are shown in Figure 1. Packet $i$ leaves the host at time $t_{a,i}^h$, arrives at the server at time $t_{b,i}$, leaves it at $t_{e,i}$ and returns at $t_{f,i}^h$. To provide reference time locally we employed a DAG3.7GP high performance measurement card to timestamp the packets just outside the host via a passive tap. The card was synchronized to a Trimble Acutime 2000 GPS receiver mounted on the laboratory roof, and was further stabilized via atomic clock for the first of our two experiments. Final timestamping accuracy was around 200ns ([3], [7]).

We use our RADclock clock synchronization daemon in the host, patches for which are available at [8]. As described in [9], RADclock is highly accurate and robust, however here we use it, not for host synchronization as such, but to provide an infrastructure for the running of the experiment, in particular to provide multiple independent streams of NTP packets, for logging, and as an independent sanity check on the DAG. Our primary data makes no use of host timestamps, but is based on the DAG and server timestamps only.

For each emitted NTP packet which completes its round-trip and is successfully matched, we obtain a 4-tuple *stamp* of timestamps $\{T_{a,i}, T_{b,i}, T_{e,i}, T_{f,i}\}$ as our basic data unit, where timestamp $T_{a,i}$ is made at time $t_{a,i}$, etc.. Here $T_{a,i}, T_{f,i}$ are recorded by the DAG, and $T_{b,i}, T_{e,i}$ are made by the server and are extracted from the returning NTP packet.

## III. The Experiment

In this section we describe the servers involved in the experiment, explain how the experiment was run, and give an overview of the resulting datasets.

### A. Server List

We performed two experiments: **Exp1** which uses a server list **List1**, and **Exp2** which uses **List2**. As seen in Table I, each experiment is of similar duration, and targets a similar number of servers.

The basis of our server choice is the list of Stratum-1 time servers maintained at *ntp.org* by the NTP Project. This list is subject to change. At the time of **Exp1** it contained around 188 servers, of which none were in Australia. In forming **List1** we selected a subset of 119 servers which were OpenAccess, and responded to NTP requests. Over three years later, we wanted to investigate whether these servers had changed behavior. **List2** was formed by removing from **List1** servers which were no longer listed at *ntp.org*[2] or which no longer responded, and adding a set of 13 Australian servers. These included one in our laboratory, and 3 from the National Measurement Institute (NMI), Australia's equivalent of the National Institute for Standards and Technology (NIST) in the US, which is responsible for maintaining national reference time.

There are 95 servers in **List1** $\cap$ **List2**. Of these, in the case of **Exp2**, only 89 had the same IP address at the beginning and end of the experiment (this information was not available for **Exp1**). To facilitate comparison between the two experiments, we report on **ListINFOCOM**, a list consisting of these 89 servers plus the 13 Australian servers from **List2**, 102 in total.

None of the servers used in our experiments are accessed via the Public NTP Pool. The ntppool system [10] provides load balancing and client configuration support by mapping server urls to physical server IP addresses transparently. It is important for this work that the server origins do not change.

| Exp | List(#servers) | Start | End | duration |
|---|---|---|---|---|
| **Exp1** | **List1**(119) | Sep. 30, 2011 | Aug. 3, 2012 | 151 days |
| **Exp2** | **List2**(117) | May 5, 2014 | April 2, 2015 | 124 days |

TABLE I
Details of the two experiments.

For interest we note that as of May 1st 2015 there were 214 servers listed as 'OpenAccess' at *ntp.org*, of which 192 had valid URLs, and none were in Australia.

### B. Data Sets

In each experiment a single laboratory host launched independent RADclock instances to each of the servers in its list in parallel. The generated dataset of the experiment consists of the set of well-matched 4-tuples for each instance/server collected by the DAG card (as well as complementary RADclock data). Each instance uses a polling period of 64 seconds, or 1350 4-tuples per day assuming no packet loss.

The experiments were allowed to run until external events (**Exp1**: over zealous building power testing; **Exp2**: hardware failure), intervened. There are also periods of significant missing data. In **Exp2** there is a gap of just over 15 days

[2]Although still readily available individually, in the wake of the NTP reflection attacks it has become difficult to obtain the full url list from *ntp.org*. Several of our IP addresses were blacklisted in the process of collecting it.

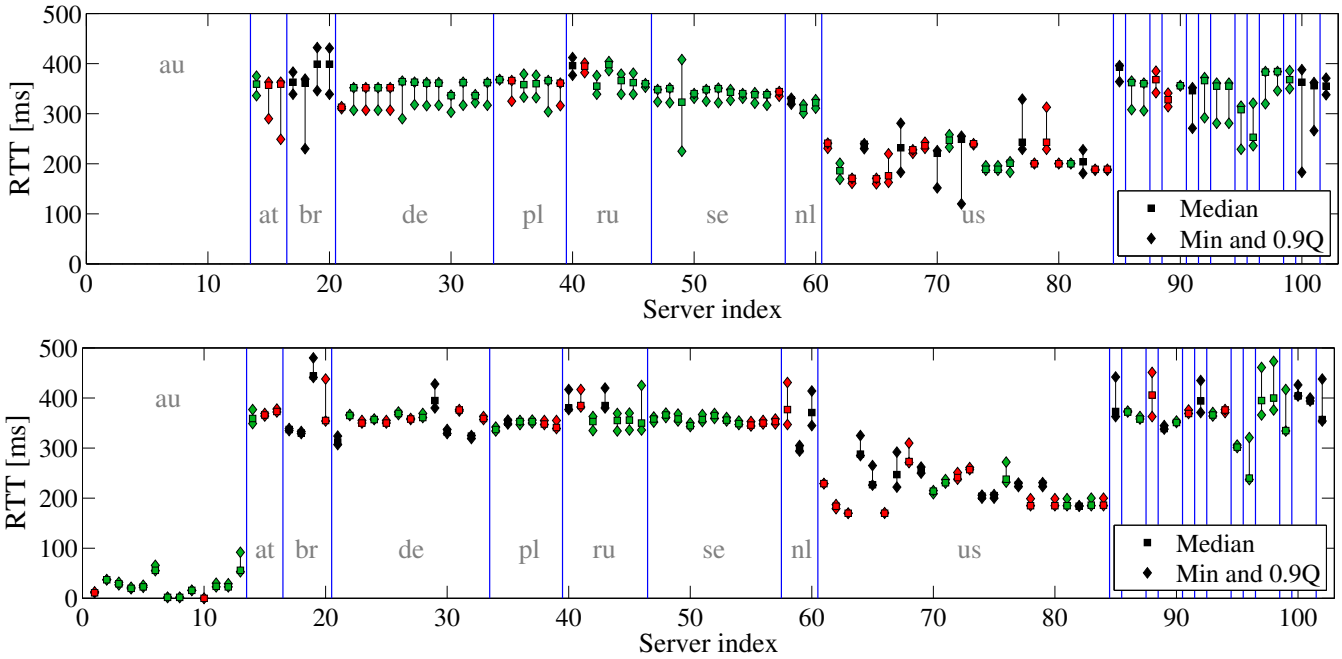


Fig. 2. RTT statistics and classification labels for the 102 **ListINFOCOM** servers. Countries with 3 or more servers are first, followed by: {am(1),be(2),bg(1),cz(2),fr(1),ie(1),it(2),jp(1),mx(1),ro(2),si(1),es(2),ua(1)}. The data spans over 3 years: Top plot: **Exp1** from 2011-12; Bottom: **Exp2** from 2014-15. The color coding follows the server anomaly classification: Red (Bad server), Green (Good) and Black (Ambiguous).

common to all servers due to the university blocking NTP traffic to and from our testbed network. In other cases server availability leads to variations. In a small number of cases data sets contain only a small number of 4-tuples, because of the aforementioned IP change issues.

One of the most important parameters impacting our experiments is the client–server RTT, and its variability. Figure 2 sets the scene by summarizing, for each experiment, the overall RTTs observed for each server. The servers are ordered alphabetically according to country code in each of two classes: first those countries with at least three servers, then the remainder. Within each country grouping the ordering is also lexicographical. We use this ordering as a unique key, in the range 1–102, to compactly identify the servers. The mapping to server url's is provided in Tables II and III below.

The results are broadly similar across the two experiments. For each a value of 100ms clearly separates the Australian servers from the others, and RTTs to the US are considerably lower than to Europe or Asia. The minimum RTT shown is often, in particular for **Exp1**, significantly lower than the median, a reflection of route changes variability as we see below. Whereas the 90% percentiles shown all fall below 500ms, the maximum RTT (not shown) is in many cases of the order of several seconds.

A related measure is that of hop count. For **Exp2** we used traceroute to measure client→server hop count, and TTL values from returning NTP packets to infer the server→client count. The result from each direction varied roughly as one would expect based on geography. The average relative difference over all servers was 7.5%.

The stratum of a server can be extracted from the returning NTP packet. In **Exp2** we found that a number of the servers, though appearing in the stratum-1 list, were actually not statum-1. In fact server #1 was always stratum-3, servers #14, #21, #35, #39 were always stratum-2, and 33 others (see Tables II and III for the full list) had a level which varied, although typically they were stratum-1 except for a small number of brief periods. These could be due to reboots, which are quite likely given the long experiment durations.

## IV. Methodology

In this section we establish the principles through which server errors can be detected, explain the key impediments to practical detection, and describe our procedure to perform our survey of server health. It is beyond the scope of this paper to develop an automated server anomaly detector. We take the first steps towards such a capability, in particular our survey constitutes an 'expert annotated' or labelled data set which can serve as pseudo ground-truth for detector development.

### A. Principles

From the measured 4-tuples $\{T_{a,i}, T_{b,i}, T_{e,i}, T_{f,i}\}$, five essential estimated path quantities can be defined:

$$\begin{aligned} \text{Forward delay} &: \; D_i^{\uparrow} = T_{b,i} - T_{a,i} \\ \text{Server delay} &: \; D_i^{\rightarrow} = T_{e,i} - T_{b,i} \\ \text{Backward delay} &: \; D_i^{\downarrow} = T_{f,i} - T_{e,i} \\ \text{Round Trip Time (RTT)} &: \; R_i = T_{f,i} - T_{a,i} \\ \text{Path Asymmetry} &: \; A_i = D_i^{\uparrow} - D_i^{\downarrow}. \end{aligned}$$

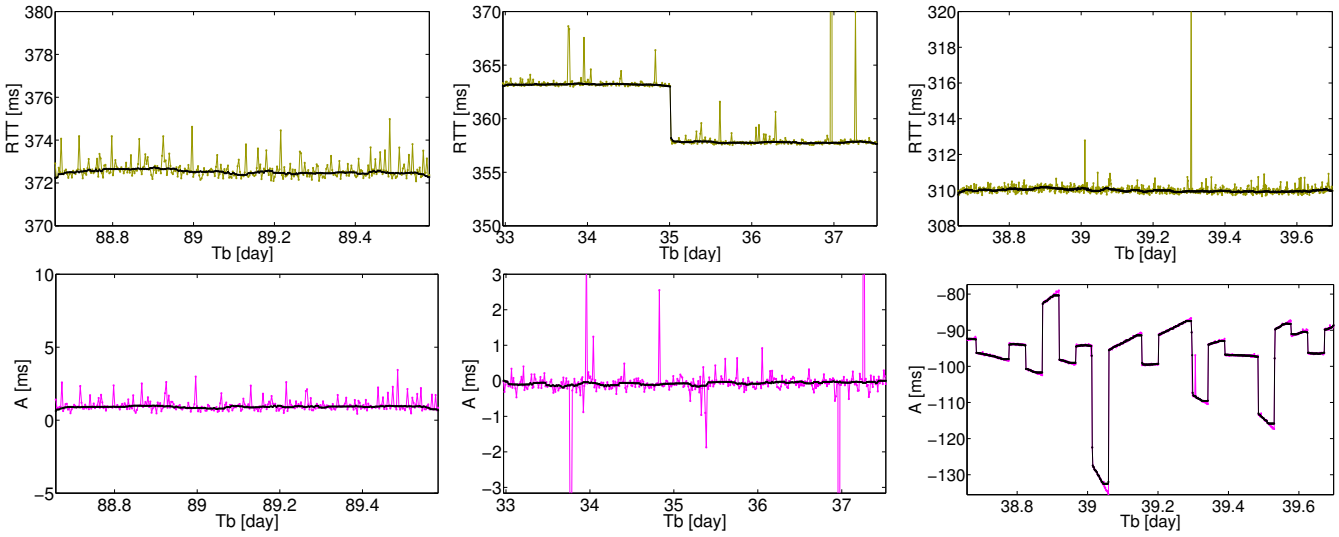


Fig. 3. Fundamentals of the $R_i$ and $A_i$ relationship. Left column (server #86 (*ntp1.oma.be*)): ideal scenario with constant $R_i$ baseline, short term congestion, and no server anomalies. Middle (server #27 (*ntps1-1.cs.tu-berlin.de*)): canonical route change scenario showing a level shift in $R_i$, which is cancelled in $A_i$ (and no anomalies). Right (server #68 (*nist.netservicesgroup.com*)): ideal $R_i$ behaviour but with $A_i$ showing large and persistent events (of 'Skew and Return' type), which must be server errors. Overlayed black curves are sliding median filterings of the time series, used to remove short term congestion variability.

The potential sources of variability in the above quantities are threefold: congestion, changes in routing, and server errors or *anomalies* (SAs). The forward and backward one-way delays (OWDs), as well as their estimates $D_i^{\uparrow}$, $D_i^{\downarrow}$ above, are subject to all three. They cannot therefore be effectively used to unambiguously detect anomalies. The exception is if they are negative, which contradicts causality (packet arrives before it is sent etc.), and immediately implicates the server, however this is a rare and extreme situation. The server delay is of limited use, except to allow us to confirm the expectation that, if a server is in error over stamp $i$, it affects both $T_{e,i}$ and $T_{b,i}$ and so is invisible in $D_i^{\rightarrow}$. Otherwise, in particular since it is typically of the order of $20\mu$s, which is negligible in most cases compared to other delays, we ignore it.

The remaining two quantities enjoy strong and complementary properties for our purpose. The most important is the RTT, as $R_i$ is *entirely independent of server timestamps*. This allows path conditions to be judged independently of server behaviour. In contrast, the asymmetry series $A_i$ directly carries the signal of any server errors: an error of size $e$ in a given stamp will appear as $2e$. Thus $A_i$ provides the basis of SA detection, with $R_i$ providing the basis of independent disambiguation, through observing whether events present in the $A_i$ time series have, or could possibly have, a network origin.

We illustrate the possible relationships between the $R_i$ and $A_i$ time series using Figure 3. The left column shows an ideal situation without anomalies. Here $R_i$ exhibits a clean baseline corresponding to a constant underlying minimum of $r_m \approx 372$ ms, plus the sum of noises due to congestion in each direction. The $A_i$ series is centered on the underlying path asymmetry $a \approx 1$ ms, which is small compared to $r_m$, about which we see the difference of these noises. Thus $A_i$ benefits from a partial cancellation of congestion related variability. In particular the variable component, $R_i - r_m$, of $R_i$, bounds the magnitude of the variable component $A_i - a$ of $A_i$ for each stamp $i$.

The middle column exhibits a canonical example of the impact of routing changes, again with no anomalies. The primary effect on $R_i$ is to induce level shifts in the minimum RTT baseline (congestion characteristics can also change with each shift, sometimes markedly). Here the single downward shift in $R_i$ is invisible in $A_i$ since each OWD changes by the same amount simultaneously. Furthermore the series is centered on $a = 0$, because each of the two routes taken are symmetric. In general, level shifts events from $R_i$ are still visible in $A_i$, but at reduced amplitude due to partial cancellation, and with jump directions which are unrelated to those of $R_i$. Partial to full cancellation of routing events is another advantage enjoyed by $A_i$.

The rightmost column in Figure 3 gives the case of an unmistakeable server anomaly. Whereas $R_i$ is well behaved with no evidence of route changes, leading one to expect an even better behaved $A_i$ as was the case in the leftmost column, instead we find an underlying variability of over 50ms, over two orders of magnitude higher than that of $R_i$, whose entire histogram (outliers aside) is well under 1ms in width. With routing and congestion sources of variability already accounted for, the cause of this can only be errors in the server itself.

Note that an alternative explanation for large changes in $A_i$ coupled to no changes in $R_i$ is that the baselines of $D_i^{\uparrow}$ and $D_i^{\downarrow}$ evolve in a precisely equal and opposite way. We consider this possibility to be pathological, and we know of no network mechanism that could give rise to it.

The anomalous server in the right column in Figure 3 is *nist.netservicesgroup.com* (server #68), which advertises as a

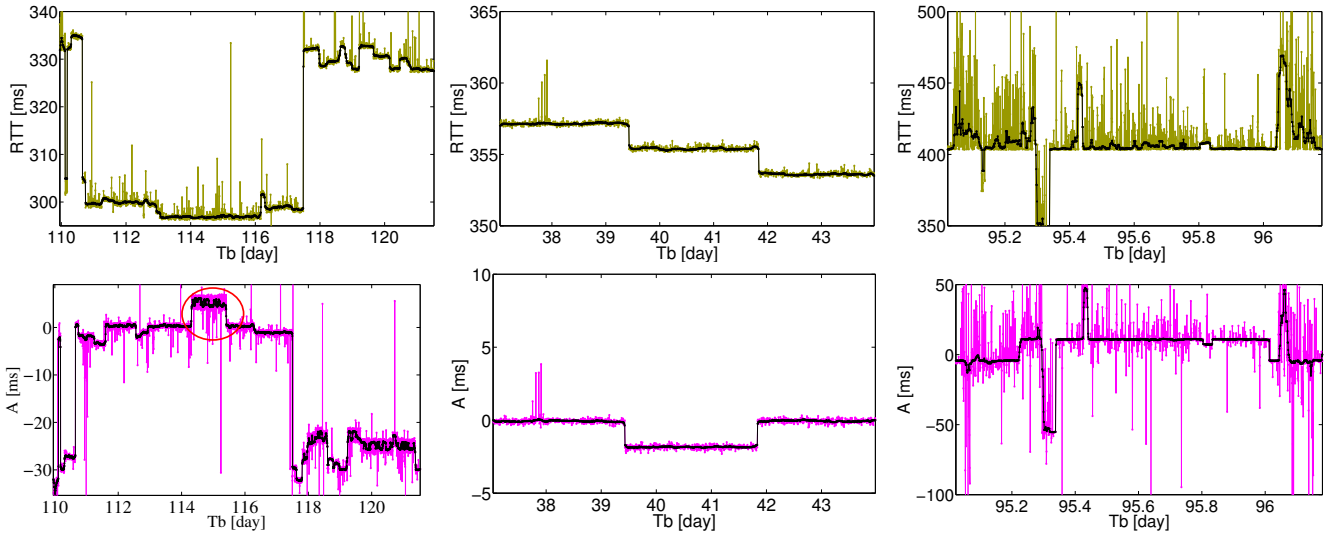


Fig. 4. Challenges to SA detection. Left column (Bad server #16 (*ntp1.oma.be*)): moderately complex $R_i$ baseline behavior, within which a server anomaly can nonetheless be detected as shown. Middle (Good server #3 (*ntp.brisbane.nmi.gov.au*)): high asymmetry so that the level change events in $R_i$ appear in A with the same instead of reduced (or zero) amplitude, but perhaps different sign. Right (Ambiguous server #100 (*hora.roa.es*)): hard to filter persistent and large congestion events present in both $R_i$ and $A_i$, making it too difficult to determine if the events seen in $A_i$ are anomalies.

stratum-1. As a NIST controlled server it could be expected to have a considerable number of stratum-2 clients, whose clocks would be directly impacted by their server's error. The anomaly is not an isolated event but follows a pattern of periods of skew (incorrect clock rate), usually linear but sometimes not, broken by sudden jumps, which is continuously present over the entire 28 hour period shown. In fact, it is present over the entire duration of both **Exp1** and **Exp2**, each months long.

### *B. Challenges*

The example of server *nist.netservicesgroup.com* above is one where the server errors were large and consistent, against a backdrop of consistently low levels of routing events and congestion. With such a high 'signal to noise' ratio the detection is simple, and quite definitive, even from the other side of the world. However, our server data exhibits a diverse set of complex behaviors, making detection, in particular automated detection, extremely challenging in the general case where anomalies may be small, rare, and buried in noise. We next describe the most important of these challenges to reliable detection.

The first challenge is that of *baseline* variability, by which we mean variations in the underlying minimum value $r_m$ of $R_i$. Given the large RTTs to the majority of our servers, and the long experiment durations, it is not surprising that routing changes are numerous, resulting in many cases in extremely complex baselines. The leftmost column of Figure 4 gives an example of moderate complexity 11 days long. It contains 10's of level shifts, with jump sizes ranging from a fraction of a millisecond to 10's of ms, and level durations rangings from days to just a few minutes or even a single stamp. Despite this complexity, it is possible to diagnose server anomalies with high confidence in certain circumstances. The black circle marks the location of a level shift event occurring in $A_i$, of amplitude 6ms, which can be diagnosed as anomalous because of the well behaved nature of $R_i$ in its immediate neighborhood. Baselines can be far more complex than the one shown here, and make the reliable automatic analysis of a long trace problematic, and manual analysis very time consuming. When there are too few events at fixed routing to distinguish congestion variability from routing events, then baselines cannot even be measured and conclusions cannot be drawn.

The second challenge is that of asymmetric routing changes. It is possible that a routing change affects one direction only. In that case there is no cancellation: the event manifests in $A_i$ with the same shape and amplitude as in $R_i$, but possibly with opposite sign. The middle column of Figure 4 gives an example where each of the jumps in $R_i$ appear in $A_i$ with the same amplitude, first in the same direction as that of $R_i$, and then the opposite one. In this case the two level changes in $R_i$ moved in the same direction and those in $A_i$ moved in opposite directions. If the reverse were true, the result would be $A_i$ having a *wider* range than $R_i$ (range$(R_i) = \max_i R_i - \min_i R_i$). Thus an event amplitude in $A_i$ which is the same size or larger than a simultaneous one in $R_i$ is not necessarily due to an anomaly, and cannot be used as a reliable signature of such, even though cancellation within $A_i$ is the more common scenario.

The final challenge is congestion. In $R_i$ this manifests as a positive noise which can be confused with, and hide, underlying level shifts, making baselines and true variability hard to measure. In $A_i$ it manifests as a bidirectional noise which could be confused with, or hide, level shifts from routing changes or SAs. When congestion arises mainly from one direction only it appears with the same shape and amplitude in both $R_i$ and $A_i$ (no cancellation). Sustained congestion

events are particularly problematic since during them the baseline is never sampled, and just as for routing changes, sustained congestion events which act in opposing directions can generate an overall amplitude range which is higher in $A_i$ than in $R_i$, even with no anomaly. The rightmost column of Figure 4 is an example of an Ambiguous labelling. Although there is a prominent LS event of amplitude 12ms in $A_i$ (which we believe is a SA), it intersects both routing and sustained congestion events. Without a clean 'ideal zone' neighborhood, we label this as Ambiguous rather than Bad.

### C. Detection Methodology

We perform, for each server in each experiment, an exhaustive manual joint assessment of the $R_i$ and $A_i$ time series to examine the evidence for server anomalies. The approach consists of five components.

**i) Congestion Filtering** With stamps being nominally 64 seconds apart, we expect the congestion component of each time series to be close to independent typically. Onto the graphs of each of $R_i$ and $A_i$ we superimpose filtered versions, $\tilde{R}_i$ and $\tilde{A}_i$, designed to suppress short term variability without impacting on events with longer timescales. We use a sliding median filter with window width $W$. This non-linear filter suppresses outliers very effectively, yet has the remarkable ability to preserve the positions of discontinuities, and leave largely invariant structures involving them (such as pairs of 'up and down' level shifts), provided their width exceeds $W/2$, and to surgically remove them if it does not.

Examples of the filtering are given as the black curves in all plots of $R_i$ and $A_i$. We use a window width of $W = 31$, large enough to dampen congestion very significantly, but short enough in timescale (33 minutes) to not significantly modify the drift of a typical free running oscillator, which forms a possible SA mode we do not wish to remove.

**ii) Ideal Zone Selection** We define an *Ideal Zone* as an interval in which, based on an examination of $R_i$ and $\tilde{R}_i$, there is no evidence of significant routing events or sustained congestion events. By 'sustained' we mean that $R_i$ remains well above the baseline over the duration in question, so that the baseline cannot be recovered, even by minimum filtering. Operationally here, we call congestion 'sustained' whether the median filtered $\tilde{R}_i$ detaches from the baseline over a time interval well beyond $W$. Restricting to such intervals deals directly and effectively with the challenges of routing changes, and congestion events which the filtering cannot remove. It returns us to the tractable situation where the variability in $A_i$ must be lower than that in $R_i$ in the absence of server error.

**iii) Server Event Amplitude Measurement** The variability, due to known sources, of $\tilde{A}_i$ from within an ideal zone should by design be very low. We measure the global amplitude of what is actually present using the range $\max_i \tilde{A}_i - \min_i \tilde{A}_i$ of $\tilde{A}_i$, as this will capture events no matter how rare they may be within the zone, and no matter what form/shape they may have (we do not wish to prejudge the forms that SAs may take), and requires no tuning parameters.

**iv) Ideal Zone Diagnosis** The final step compares the amplitude measurement (the server signal) against the degree of variability in $\tilde{R}_i$ (the path noise). If the signal clearly exceeds the noise we declare that an anomaly has been found. Expert judgement plays an important role here in deciding which features in $R_i$ and $\tilde{R}_i$ to include in the definition of the noise. For example variability due to small routing events can be tolerated (ignored) if it is seen that its effect in $A_i$ is negligible in any case. Another common example: it may be clear visually that an anomaly will be detected provided the ideal zone is cropped to exclude an unrelated level shift from $R_i$ of a competing or larger magnitude.

**v) Server Labelling** Given a server, each ideal zone in each experiment is tested as above, and the server labelled as:

**Bad:** clear evidence of at least one SA across the experiments,
**Good:** no sign of a server anomaly across either experiment,
**Ambiguous:** evidence of possible SAs, but inconclusive given noise levels, or simply too noisy for any determination.

**Bad** is a confident statement that an anomaly has been found. It is not a mere heuristic classification, but rather a direct measurement of error while controlling (informally but conservatively) for statistical errors. **Good** is a confident statement that there are no major errors, but does not imply that there are none. Anomalies falling outside ideal zones will be missed by our approach. and anomalies smaller than the underlying congestion noise level, or suppressed by the short range congestion filtering, will generally be indetectable.

### D. A Prototype Detection Tool: The Adjusted Range Test.

We briefly describe a detector which we have developed in parallel with, and in support of, the manual assessment. Our goal was to seek a tool which is simple, and yet capable of detecting anomalies at least in many of the less challenging cases. By simple, we mean capable of processing the $R_i$ and $A_i$ series fully and jointly without resorting to feature extraction/comparison, such as the localization of ideal zones, or measurement of LS positions, heights, and degrees of cancellation, which are all very difficult to achieve robustly.

The method is built on the median filtered $\tilde{R}_i$ and $\tilde{A}_i$ above, and is based on the idea that, at a per-stamp level, the value of $R_i$ above its baseline can be used as a bound on the variability of $A_i$, and so $\tilde{R}_i$ can be used to bound variability in $\tilde{A}_i$. Let $m_R$, $m_A$ be the (scalar) medians of the $\tilde{R}_i$ and $\tilde{A}_i$ series respectively. We define a conservative estimate of the per-stamp bound as $b_i = |\tilde{R}_i - m_R|$, and use $b_i$ it to push $\tilde{A}_i$ towards $m_A$ without overshoot for each $i$, which reduces its potential contribution to the range. The range of this *adjusted* $\tilde{A}_i$ is then used as the 'signal' amplitude. Essentially we dial-down $\tilde{A}_i$ in non-deal zones, instead of localizing them and excluding them altogether. The global RTT noise level is estimated as the width of the $\tilde{R}_i$ histogram excluding outliers (we use the interval between the 5th and 95th percentiles), then doubled to take into account the one-sided worst case described above. The final adjusted-range to RTT-noise ratio we denote by the test statistic $\mu$. Nominally values above unity signify detection, in practice we compare against a threshold.

The above method has clear limitations, but performs well enough to be useful. We use it to quantify anomaly size and S&R in the manually chosen ideal zones. In such an assisted environment it replicates the manual classification very closely.

## V. Results

We report on the results of our server classification in two ways. First in Figure 2 via a color code, and second in Table II for the 37 Bad servers only, and Table III for the servers which were not Bad (hence Good or Ambiguous in each experiment). Figure 2 makes it easy to see how the servers fares with respect to their country of origin and RTT, whereas Table II allows the two experiments per-server to be compared more easily. Of the countries with 3 or more servers, Australia and Sweden performs the best, and Brazil and the US the worst. Larger RTT implies longer paths, and so greater potential to pick up more routing events and congestion which make it harder to spot anomalies. Certain European countries have more Ambiguous ratings, reflecting their extremely complex baselines. The main systematic difference between **Exp1** and **Exp2** was a change in the character of path asymmetry, whereas the servers themselves were remarkably consistent as the tables show. In **Exp2** fully asymmetric route changes were more common, making detection more challenging both in terms of congestion and routing events in $A_i$.

Table II provide a breakdown of the Bad servers according to a number of criteria which we now describe. Note that we did not find any evidence of correlations between anomalies across servers.

*Prevalence* is a measure of the number of anomalies found. We observed just two main scenarios: **R**are, when only a very small number, typically 1 or 2, were found, and **H**igh, where anomalies are found 'almost everywhere'. We speculate that the latter are due to servers which have a systemic problem such as a failed GPS receiver. The latter case can be subdivided into two subcases. The first are those with high congestion and/or very complex baselines. Here the small number found is a reflection of how hard it is to find a viable ideal zone, and it seems likely that there are many other undiagnosed anomalies. The second are servers which would otherwise rate as Good. Here the anomalies may relate to convergence issues following reboots or other disturbances.

*Confidence* is a measure of how 'clear and obvious' the anomalies are. The default value is **H**igh, since by definition, we are confident of our conservative Bad classifications. We reserve **V**ery **H**igh for those servers which have High prevalence, and for which the anomaly stood out particularly clearly given (i) its amplitude compared to the network noise, and (ii) the degree of coherence of both network and congestion events between $R_i$ and $A_i$, giving confidence in the interpretation. Server *nist.netservicesgroup.com* from Figure 3 is a member of this 'unmistakeably anomalous' class.

*Anomaly Type* is an attempt to classify the form and shape of the observed anomalies. All types that are observed (in different locations within the time series) are listed for each server. The most common form, exemplified, again, by *nist.netservicesgroup.com*, we call **S&R**, that is a period of **S**kew followed by a **R**eturn (jump). This class contains many variations of detail, and we speculate that it is due to a misbehaving clock disciplining algorithm which is corrected (when error is detected as too great) by a jump. The **LS** anomalies take the form of a level shift, followed by second shift returning back to a similar level some time later. These are rare, and may be due to some error-control mechanism mistakenly believing that an error correction is required, and later changing its mind. Finally, **D**rift anomalies are those that take the characteristic random walk appearance of an oscillator free running under temperature variations, a clear symptom of a failure to lock onto a reference signal. An example is given in Figure 5 for server #69 (*ntp1.oma.be*) over an ideal zone where $R_i$ has a histogram width of just a few ms. In contrast, the drift anomaly has an amplitude of over 100ms and has High prevalence (present throughout the trace).

The final two fields, **Size** and **Test statistic**, are the values output by the detection tool described in Section IV-D, when applied to the most representative of the ideal zones found. This could be the widest such zone, or the one with the clearest interpretation coupled with a high amplitude (not necessarily the zone we have chosen to showcase in Figures 3, 4, and 5 for servers #68, #69, #16 respectively). For interest, we explored the robustness of the test statistic by applying it to the entire timeseries for each server in each experiment, treating the manual assessment as ground truth. A server was labelled as Bad if $\mu > 2$. The method labelled 37 out of the $2 \times 102 = 204$ cases as Bad, compared to 53 manually. Of these there were 9 false positives (labelled as Bad but actually not), and 25 missed detections (not labelled Bad but actually are).

Finally we compare results according to the Stratum level classification from the first column. We find no systematic difference between the Stratum-1 and Varying Stratum categories. This suggests that the latter may simply be due to Stratum-1's briefly dropping stratum level following occasional system reboots or resets (note however that the servers with varying stratum from Table III have a much high percentage of Ambiguous rankings). In contrast, the 4 Bad servers which are never stratum-1 perform consistently poorly as expected. They

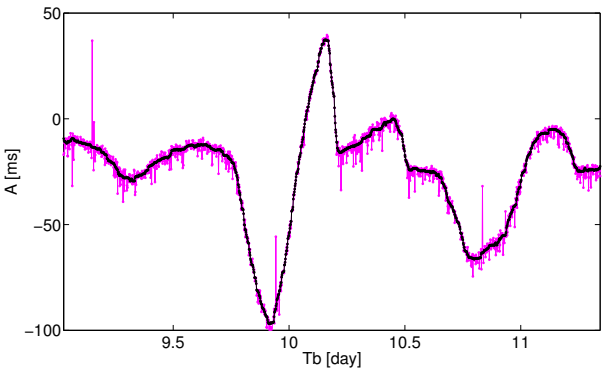


Fig. 5. An example of $A_i$ containing an anomaly of **D**rift type from server #69 (*ntp1.oma.be*). The anomaly is present throughout the entire **Exp1** trace.

| Stratum level | ServerId # | URL | Detection { G, Am, B } ( Exp1, Exp2 ) | Confidence { H, VH } | Prevalence { R, H } | Anomaly Type { D, LS, S&R } | Size [ ms ] | Test statistic |
|---|---|---|---|---|---|---|---|---|
| Stratum-1 (always) | 10 | server-in-our-lab.au | ( –, B ) | H | H | D | 0.8 | 6 |
| | 27 | ntps1-1.cs.tu-berlin.de | ( G, B ) | H | R | LS | 3 | 8 |
| | 38 | time.coi.pw.edu.pl | ( G, B ) | H | R | D, S&R | 19 | 30 |
| | 61 | bonehed.lcs.mit.edu | ( B, B ) | VH | H | S&R | 7 | 15 |
| | 65 | clock.sjc.he.net | ( B, Am ) | H | H | LS | 13 | 64 |
| | **68** | (Fig. 3) nist.netservicesgroup.com | **( B, B )** | **VH** | **H** | **S&R** | **27** | **25** |
| | **69** | (Fig. 5) **ntp.myfloridacity.us** | **( B, Am )** | **VH** | **H** | **D** | **130** | **29** |
| | 78 | time-a.timefreq.bldrdoc.gov | ( B, B ) | VH | H | S&R | 98 | 934 |
| | 79 | time-b.nist.gov | ( B, Am ) | H | R | LS | 18 | 6 |
| | 80 | time-b.timefreq.bldrdoc.gov | ( B, B ) | H | H | D, S&R | 2 | 8 |
| | 83 | utcnist.colorado.edu | ( B, G ) | VH | H | D, S&R | 6 | 29 |
| | 84 | utcnist2.colorado.edu | ( B, B ) | H | H | D, S&R | 4 | 23 |
| | 91 | canon.inria.fr | ( Am, B ) | H | R | S&R | 28 | 67 |
| Stratum-2 (always) | 21 | ntp.probe-networks.de | ( B, Am ) | H | H | LS | 169 | 372 |
| | 35 | ntp1.net.icm.edu.pl | ( B, Am ) | H | H | D | 12 | 21 |
| | 39 | vega.cbk.poznan.pl | ( B, B ) | VH | H | D, S&R | 208 | 1949 |
| Stratum-3 | 1 | augean.eleceng.adelaide.edu.au | ( –, B ) | VH | H | D | 26 | 62 |
| Varying Stratum | 15 | ts1.aco.net | ( B, B ) | H | R | LS | 3 | 11 |
| | **16** | (Fig. 4) **ts2.aco.net** | **( B, B )** | **H** | **R** | **LS** | **6** | **6** |
| | 20 | ntps1.pads.ufrj.br | ( Am, B ) | H | H | S&R | 3 | 11 |
| | 23 | ntp1.fau.de | ( B, B ) | H | H | LS | 3 | 6 |
| | 25 | ntp3.fau.de | ( B, B ) | H | H | LS, S&R | 3 | 23 |
| | 31 | time.fu-berlin.de | ( G, B ) | H | R | S&R | 108 | 253 |
| | 33 | zeit.fu-berlin.de | ( G, B ) | H | R | S&R | 14 | 50 |
| | 41 | ntp1.ntp-servers.net | ( B, B ) | VH | H | LS, S&R | 42 | 9 |
| | 55 | time1.stupi.se | ( G, B ) | H | R | S&R | 50 | 173 |
| | 56 | time2.stupi.se | ( G, B ) | H | R | S&R | 283 | 79 |
| | 57 | timehost.lysator.liu.se | ( B, B ) | H | H | D, S&R | 15 | 35 |
| | 58 | ntp.remco.org | ( Am, B ) | H | H | S&R | 9 | 10 |
| | 62 | clock.danplanet.com | ( G, B ) | VH | H | D, S&R | 1441 | 900 |
| | 63 | clock.isc.org | ( B, B ) | VH | H | D, S&R | 8 | 62 |
| | 66 | clock.via.net | ( B, B ) | H | H | D, LS | 45 | 93 |
| | 72 | ntp2.netwrx1.com | ( Am, B ) | H | H | D | 45 | 9 |
| | 73 | rackety.udel.edu | ( B, B ) | H | R | LS, S&R | 15 | 44 |
| | 88 | ntp.bsdbg.net | ( B, B ) | VH | H | D, S&R | 59 | 60 |
| | 89 | ntp.nic.cz | ( B, Am ) | H | R | S&R | 3 | 7 |
| | 94 | ntp2.inrim.it | ( G, B ) | H | R | S&R | 4 | 6 |

TABLE II

BREAKDOWN OF SERVER ANOMALY CHARACTERISTICS DISCOVERED IN THE 'BAD' SERVERS. NIST CONTROLLED SERVERS HAVE BEEN HIGHLIGHTED IN RED, AND SERVERS WITH AN ANOMALY DISPLAYED IN SOME FIGURE, IN BOLD.

all have High prevalence, 2/4 of them display VH confidence, and 3/4 of them display drift, which is consistent with a diagnosis of a failure to synchronize to their hardware (GPS). Finally, they contain some of the very highest values of Size and Test statistic.

Server #10 is is a rack mounted 1U DELL server in our laboratory running FreeBSD, whose system clock is disciplined (like many commodity stratum-1's) using the *ntpd* daemon with GPS pulse-per-second input. It was classified as Bad because of a small number of periods, up to 1 day long, where a very clear yet small, locally only 0.03 ms in amplitude, oscillatory anomalous behaviour was observed. The diagnosis was possible despite the very small amplitude, because, being on the LAN, the baseline was perfect and the $R_i$ was exceptionally small and clean. The anomaly is familiar to us (see [9]), being traceable to periodic forcing from the airconditioning cycle in the machine room. This example validates our approach, and shows the potential for increased sensitivity when the test client is closer to the server.

Given their mandated role as standard keepers in the US and Australia respectively, it is important to comment on those NIST and NMI controlled servers included in **ListINFOCOM**. The NMI servers are available to registered users only and so did not appear at *ntp.org*. All three were labelled as Good, however they are subject to strong diurnal congestion cycles which would disadvantage their clients. This is consistent with our knowledge of the infrastructure based on discussions with the maintainers at the NMI. The hardware consists of well monitored atomic standards, however network access is often bandwidth constrained. Out of the 8 NIST servers in the list, 6 were labelled Bad, and of these 4 were Bad in both **Exp1** and **Exp2**. All but one has high prevalence in both **Exp1** and **Exp2**, and 3 out of 6 were in the VH confidence class.

## VI. CONCLUSION

We have sought both to develop a unique capability to assess time servers remotely, and to use it to provide a first look at the health of the root of the public timing system.

In terms of capability, we have described in detail the principles, pitfalls, and practical approaches of a server analysis and labelling methodology capable of unambiguously detecting server errors of diverse types. Its effectiveness is such that it

| Stratum level | ServerId # | URL | Detection {G, Am} (Exp1, Exp2) |
|---|---|---|---|
| Stratum-1 (always) | 2 | csiro-nml.physics.uwa.edu.au | (–, G) |
| | 3 | ntp.brisbane.nmi.gov.au | (–, G) |
| | 4 | ntp.melbourne.nmi.gov.au | (–, G) |
| | 5 | ntp.sydney.nmi.gov.au | (–, G) |
| | 6 | ntp.waia.asn.au | (–, G) |
| | 7 | ntp1.net.monash.edu.au | (–, G) |
| | 8 | ntp10.net.monash.edu.au | (–, G) |
| | 9 | syd4gps0.syd.ops.aspac.uu.net | (–, G) |
| | 11 | tick.une.edu.au | (–, G) |
| | 12 | tock.une.edu.au | (–, G) |
| | 17 | a.st1.ntp.br | (Am, Am) |
| | 22 | ntp0.fau.de | (G, G) |
| | 24 | ntp2.fau.de | (G, G) |
| | 26 | ntps1-0.cs.tu-berlin.de | (G, G) |
| | 28 | ptbtime1.ptb.de | (G, G) |
| | 29 | ptbtime2.ptb.de | (G, Am) |
| | 30 | rustime01.rus.uni-stuttgart.de | (G, Am) |
| | 32 | time1.one4vision.de | (G, Am) |
| | 34 | ntp.certum.pl | (G, G) |
| | 37 | ntp2.tp.pl | (G, G) |
| | 42 | ntp1.vniiftri.ru | (G, G) |
| | 43 | ntp2.ntp-servers.net | (G, Am) |
| | 44 | ntp2.vniiftri.ru | (G, G) |
| | 45 | ntp3.vniiftri.ru | (G, G) |
| | 47 | ntp1.gbg.netnod.se | (G, G) |
| | 48 | ntp1.mmo.netnod.se | (G, G) |
| | 49 | ntp1.sp.se | (G, G) |
| | 50 | ntp1.sth.netnod.se | (G, G) |
| | 51 | ntp2.gbg.netnod.se | (G, G) |
| | 52 | ntp2.mmo.netnod.se | (G, G) |
| | 53 | ntp2.sp.se | (G, G) |
| | 54 | ntp2.sth.netnod.se | (G, G) |
| | 59 | ntp0.nl.uu.net | (G, Am) |
| | 60 | ntp4.linocomm.net | (G, Am) |
| | 70 | ntp.your.org | (Am, G) |
| | 71 | ntp1.conectiv.com | (G, G) |
| | 77 | time-a.nist.gov | (Am, Am) |
| | 81 | time-c.timefreq.bldrdoc.gov | (G, G) |
| | 82 | timekeeper.isi.edu | (Am, Am) |
| | 85 | ntp.amnic.net | (Am, Am) |
| | 86 | ntp1.oma.be | (G, G) |
| | 87 | ntp2.oma.be | (G, G) |
| | 90 | time.ufe.cz | (G, G) |
| | 95 | clock.nc.fukuoka-u.ac.jp | (G, G) |
| | 96 | cronos.cenam.mx | (G, G) |
| | 97 | ntp2.usv.ro | (G, G) |
| | 98 | ntp3.usv.ro | (G, G) |
| | 99 | ntp.mostovna.com | (G, G) |
| | 100 | hora.roa.es | (Am, Am) |
| | 101 | ntp.i2t.ehu.es | (Am, Am) |
| | 102 | ntp.time.in.ua | (Am, Am) |
| Stratum-2 | 14 | asynchronos.iiss.at | (G, G) |
| Varying Stratum | 13 | vk6hgr.echidna.id.au | (–, G) |
| | 18 | c.st1.ntp.br | (Am, Am) |
| | 19 | d.st1.ntp.br | (Am, Am) |
| | 36 | ntp1.tp.pl | (G, G) |
| | 40 | ntp0.ntp-servers.net | (Am, Am) |
| | 46 | ntp4.vniiftri.ru | (G, G) |
| | 64 | clock.nyc.he.net | (Am, Am) |
| | 67 | gps.layer42.net | (Am, Am) |
| | 74 | t1.timegps.net | (G, Am) |
| | 75 | t2.timegps.net | (G, Am) |
| | 76 | time.xmission.com | (G, G) |
| | 92 | ntp-galway.hea.net | (G, Am) |
| | 93 | ntp1.inrim.it | (G, G) |

TABLE III
NTP SERVERS NOT LABELLED AS BAD, HENCE GOOD OR AMBIGUOUS IN EACH EXPERIMENT. NIST (RESP. NMI) CONTROLLED SERVERS HAVE BEEN HIGHLIGHTED IN RED (RESP. PURPLE).

can operate under extremely challenging path environments, with large RTT and many and diverse routing and congestion events, although of course these factors limit its sensitivity.

In our server testing we found many instances of server error. In some cases these were small and rare, but in many others they were of an amplitude and frequency well beyond what one would expect and require from the root of the timing system. Our main findings include:
i) Errors were found in 37 of the 102 servers examined. In 24 of these, anomalies was continuously present for months at a time. In 22 cases the anomaly amplitude exceeded 10ms, and in 9 cases 50ms, enormous compared to the nominal time server error of 0.01ms.
ii) The most common anomaly type was 'Skew and Return' (seen in 24 out of 37) followed by Drift (14) then LS (8)
iii) A high proportion of NIST servers exhibit errors, and are among the worst servers in the list.

There are many exciting directions for future work. In terms of the methodology, we intend to develop an automated classifier capable of processing long, complex, traces with low false positive and high detection rates, and to make it available to the community. In terms of stratum-1 health, we intend to examine the current datasets more fully, and expand the study to include a much longer list. The main limitation in our current data however is the high path noise. By moving closer to the servers to reduce this, and by using multiple vantage points to further help resolve Ambiguous cases and cross-check conclusions, we can greatly enhance detection sensitivity. To this end we are planning a measurement campaign making use of CAIDA's Ark [11] monitoring network. Finally, we intend to contact the operators of servers where we found consistent errors.

## REFERENCES


[1] D. L. Mills, "Internet time synchronization: the Network Time Protocol," *IEEE Trans. Communications*, vol. 39, no. 10, pp. 1482–1493, October 1991.

[2] J. Czyz, M. Kallitsis, M. Gharaibeh, C. Papadopoulos, M. Bailey, and M. Karir, "Taming the 800 Pound Gorilla: The Rise and Decline of NTP DDoS Attacks," in *Proc. IMC 2014*, ser. IMC '14. New York, NY, USA: ACM, 2014, pp. 435–448. [Online]. Available: http://doi.acm.org/10.1145/2663716.2663717

[3] "Endace Measurement Systems," http://www.endace.com/.

[4] C.-Y. Hong, C.-C. Lin, and M. Caesar, "Clockscalpel: Understanding root causes of Internet clock synchronization inaccuracy," in *Proceedings of the 12th international conference on Passive and active measurement*, ser. PAM'11, N. Spring and G. Riley, Eds., vol. 6579. Berlin, Heidelberg: Springer-Verlag, 2011, pp. 204–213. [Online]. Available: http://dl.acm.org/citation.cfm?id=1987510.1987531

[5] N. Minar. (1999) A Survey of the NTP Network. Http://xenia.media.mit.edu/ nelson/research/ntp-survey99/ntp-survey99-minar.ps. [Online]. Available: http://alumni.media.mit.edu/~nelson/research/ntp-survey99/html/

[6] J. D. Guyton and M. F. Schwartz, "Experiences with a Survey Tool for Discovering Network Time Protocol Servers," 1994, [Online; accessed 31-July-2015]. [Online]. Available: http://static.usenix.org/publications/library/proceedings/bos94/full_papers/guyton.a

[7] J. Micheel, I. Graham, and S. Donnelly, "Precision Timestamping of Network Packets," in *Proc. ACM SIGCOMM Internet Measurement Conf.*, Nov. 2001, pp. 273–277.

[8] "RADclock Project webpage," http://www.synclab.org/radclock/. [Online]. Available: http://www.synclab.org/radclock/

[9] D. Veitch, J. Ridoux, and S. B. Korada, "Robust Synchronization of Absolute and Difference Clocks over Networks," *IEEE/ACM Transactions on Networking*, vol. 17, no. 2, pp. 417–430, April 2009. [Online]. Available: http://www.cubinlab.ee.unimelb.edu.au/~darryl/Publications/synch_ToN.pdf

[10] ntp.org, "NTP Pool Project," 2015, [Online; accessed 31-July-2015]. [Online]. Available: http://www.ntppool.org/

[11] Cooperative Association for Internet Data Analysis (CAIDA), http://www.caida.org/home.