Addendum to the White paper on Windows Time Sync Accuracy

Contents

Summary

This is an addendum referenced by the Windows 2016 Accurate Time article. The following documentation provides more details about our testing and measurement methodologies.

Windows Time Service in Server 2012R2 and prior releases was mainly focused on keeping time in an Active Directory domain to within a 5-minute window to allow Kerberos authentication protocol to function. This was a pretty low bar to meet and once set up, the time synchronization worked smoothly. In case of any issues with time synchronization, customers could use the in-box tool W32tm.exe to run a set of diagnostic commands for troubleshooting. One of the commands is "stripchart", which sent NTP pings to a specified time server and printed the time offset as its output (The command line format looked like this: *w32tm.exe /stripchart /computer:<NTPServer> ….*). This seemed to be reasonably accurate in a short time span and served its purpose in prior releases.

We have made significant improvements in the time synchronization accuracy in Server 2016 in order to achieve sub-millisecond time sync accuracy to meet the requirements in various new scenarios besides Kerberos.

Sub-millisecond time synchronization can only be achieved under certain conditions. Accurately measuring the time synchronization correctly down to sub-millisecond values is not straight forward, due to noise that can creep in to the measurement process.

This addendum describes the topologies we set up to measure the accuracy and the different techniques used to measure the very small time offsets. Next it describes a simple post-process filtering of data collected using NTP pings (by W32tm.exe “stripchart” or other means) in Server 2016, which helps simplify the measurement of time sync accuracy in any generic topology.

Next several comparisons are made between time sync accuracy in Server2012R2 AD forest and Server2016 AD forest and one comparison that shows degradation of NTP accuracy in Server2016 due to increasing number of network hops. This should help the reader visualize the improvements that have been made in Server 2016 and to some extent, understand the impact the topology has on the accuracy.

The rest of the document goes into details of measurements taken by 3 different ways and demonstrates the validity and effectiveness of the simple filtering of NTP ping data in Server2016 topologies. The data from these measurements has been used in the comparisons made in this document.

Measuring Time Sync Accuracy

Measurement Tools

Our teammate Alan Jowett has published a set of tools to measure time sync accuracy on GitHub (https://github.com/Microsoft/Windows-Time-Calibration-Tools). Of these, we will be using the tools for TSC based offset in virtual machines in our measurements (specifically OsTimeSampler, Scripts and TimeSampleCorrelation).

These tools are based on the idea that we can compute the time offset between a Hyper-V host and a VM guest over a period by recording the Time Stamp Counter (TSC) and System time in both machines simultaneously, recording the fixed difference between the TSC values in both machines and processing the data offline.

The processing involves correlating TSC data points from both machines and interpolating the data within the sampling interval using LaGrange polynomial curve fitting. For this process to measure the time offset of the VM’s system clock reliably and accurately, the Hyper-V host system clock needs to be maintained reliably and accurately.

The other means to measure time accuracy is using simple NTP pings. This functionality is conveniently available in the inbox tool W32tm.exe using the switch “stripchart”. Here is are examples of this command:

W32tm.exe /stripchart /computer:time.windows.com /period:1

W32tm.exe /stripchart /computer:<YourNTPServer> /period:1

Comparing the time offset data from TSC with the time offset data gathered from W32time stripchart (or any other software that can send NTP pings) would help us ascertain the reliability of both methods and show us any limitations therein. The NTP pings could be made in either direction (from accurate time server to the test machine and vice versa) and the results can be compared in a similar manner as well. If the three data sets provide us with the same data, we can be very certain they are pointing at the same common truth.

Note: Alan's GitHub repository includes NTP ping based tools, but those were not used to generate the data presented here. The results from those tools should be very close to the results demonstrated here.

The tests we ran are mostly geared to measure the accuracy of secure NTP in an AD Forest, though we have one test to measure the performance of simple/unsecured NTP as well.

For our AD topology tests, we have used two Hyper-V hosts. These are server blades with lots of memory and processing power and multiple hard disks each and each is outfitted with Sync-n-Scale high accuracy time hardware/drivers. We built our AD topologies using guest VMs running on these host machines. The variety of topologies is endless. Our topologies were set up to demonstrate 2 strata of time servers in the domain as well as various scenarios for networking stack/OS versions etc., to give a peek into the potential of the time sync accuracy in Server 2016.

We have one non-AD simple NTP test which used a generic physical machine connected to our corporate network. This did not involve collecting TSC data.

Before collecting data on any computer being tested, we first configure it and let the time synchronization run on it for 2-3 hours to allow the computer's time to converge with the reference time. Time offset/error measurements are taken after this period. We followed the same process for data collection on all computers in a complex topology, allowing 2-3 hours for the entire topology to converge.

We collected the following four data streams simultaneously for measuring time on a VM:

1. TSC related data using OsTimeSampler tool (see github link above).
    a. TSC and System Time inside the VM, sampled every second
    b. TSC and System Time in the Hyper-V, host sampled every second
2. Ntp ping data using W32tm stripchart command, as measured inside the VM against the W32time NTP server running on its Hyper-V host. (*eg: w32tm.exe /stripchart /computer:<hyper-v-host> /period:1*)
3. Ntp ping data using W32tm stripchart command, as measured inside the Hyper-V host against the W32time NTP server running on the VM. (*eg: w32tm.exe /stripchart /computer:<VMt> /period:1*)

We collected only the following data stream for measuring time inside a physical machine for the simple NTP test:

4. NTP ping data using W32tm stripchart command, as measured in the test machine against the accurate NTP server. (eg: *w32tm.exe /stripchart /computer:<AccurateNTPServer> /period:1)*

Note: In Server 2016, data streams (1a, 2) and (1b, 3) can be collected simultaneously using the new "rdtsc" switch for stripchart command. The command line looks like this:

> *W32tm.exe /stripchart /computer:time.windows.com /rdtsc /period:1*

However, this requires a modified TimeSampleCorrelation tool in order for it to be used with this data. The modified version of the tool published on GitHub shortly.

The time offset data collected using NTP pings is typically noisy due to asymmetry in network transit times. W32time service uses complex filtering to reduce or eliminate this noise. One of the goals of this document is to demonstrate the effectiveness of a simple filter described next to eliminate noise in NTP measurements and show that the filtered data can be used for reliable time offset determination.

Popcorn Filter

The output of w32tm stripchart command is the raw time offset derived from NTP pings as specified in the SNTP protocol. You can write a custom tool to implement this function as well. (Alan's github share has his custom NTP ping tool),

NTP time offset computation assumes the request/response roundtrip times are symmetric. This does not always hold true, even in simple topologies. In the worst case, NTP can add up to half of the roundtrip time as noise to the offset measurement. This noise can be up to ~100-200 microseconds on average and it can spike up to *several milliseconds* in some cases.

If we have two computers whose time is synchronized with each other within, say 200 microseconds, the noisy NTP measurements can be a significant compared to this offset. The readings we get from NTP will be incorrect if we include the noisy samples and the raw data cannot be used for measuring time offset reliably.

Based on the research shared in this document, you can see that the NTP Round Trip Times (RTT) falls in a wide range with a pseudo-Gaussian distribution. We can see that there are some values that are way above the range of RTT seen "normally". These samples may or may not contain valid time offset data and they are the noise in the raw data. They occur randomly during measurement, much like kernels starting to pop while making popcorn. This noise in NTP is being referred to here as the "*Popcorn Noise*" and the simple filter being described to eliminate the noise is named the "*Popcorn filter*".

Each data point measured with NTP pings should contain this information: (RTT, Offset). Once we collect a data set for an interval, we do the following:

1. Determine the minimum RTT value (Min_RTT) (which is the absolute minimum RTT value possible)
2. Determine the average RTT value (average of all RTT values in the set) (Avg_RTT).

Based on our observations, it seems like It is *very unusual* for RTT to exceed (Avg_RTT + (Avg_RTT - Min_RTT)). In all cases with large RTTs, it is very likely the measured time offset is incorrect due to NTP noise.

The Popcorn filter uses the hypothesis that legitimate NTP measurements come from this limited range:

> **[Min_RTT, (Avg_RTT + (Avg_RTT – Min_RTT))]**

and if an NTP ping sample has RTT outside the above range, it can be excluded from consideration without any loss of information.

Note that we can apply this filter only to data from Server 2016 and not on data from Server2012R2 because the time offsets in Server2012R2 exceeds the roundtrip times in all cases. Applying this filter would have very little effect on Server 2012R2 data. Additionally, the RTT cannot be accurately computed on Server2012R2 because it is in the sub-millisecond range for our set up and the W32tm.exe stripchart utility on Server 2012R2 does not support this level of accuracy.

Notes Server vs Client Terminology and Version information

W32time implementation ships with NTP Server, NTP Client and VMIC providers in-box on both Server and Client SKUs. The improvements to W32time are available in both Server 2016 and Windows 10 Anniversary Update releases.

The term "client" is being used in the topology to refer to a domain member/client, which in all cases is a Server 2016 machine in our tests. It should be noted that identical results could be obtained when using a Windows 10 Anniversary update domain client.

Server 2016 Forest Test Topology

The underlying server hardware machines have 8 processor cores and 16 GB of memory each. The VMs are each assigned 2 processor cores and 4GB of hardware each, with the remaining available for the Hyper-V host partition's use. All the machines are collecting time related data described above and not doing much else.

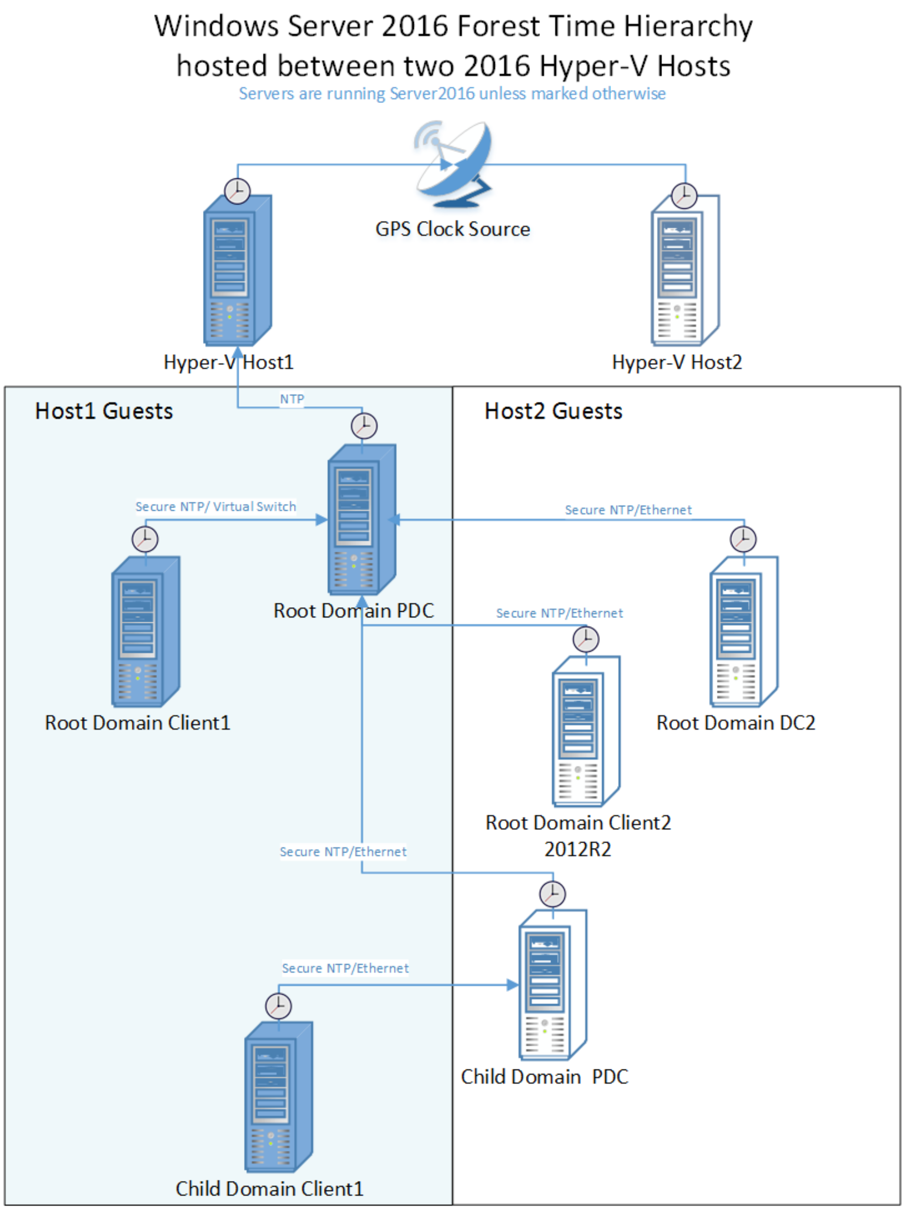


Figure 1 Server 2016 Forest Test Topology

Server2012R2 Forest Test Topology

This topology was configured on the same hardware as the Server2016 topology, allocating identical resources to the VMs running therein. The data collection is identical to Server2016 topology as well.

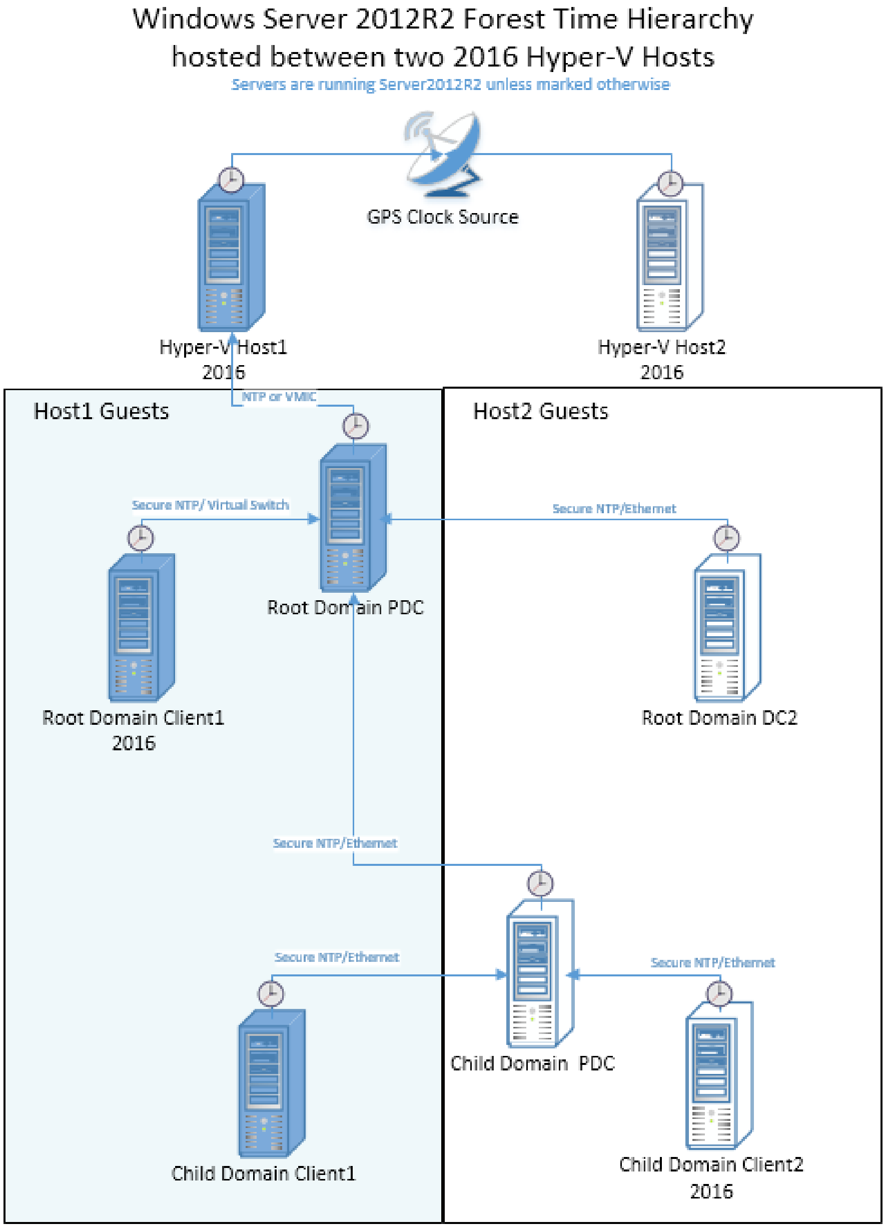


Figure 2 Server 2012R2 Forest Test Topology

Methodology for AD Time Sync Accuracy Measurements

- We know that the Hyper-V hosts keep a very accurate time using the Sync-n-Scale hardware/software.
- All the computers in the above topology have the recommended high accuracy time settings. (See Script #1 Setting up common high accuracy settings)
- Two scenarios are tested out for each topology:
    - Root PDC syncing time over VMIC (See Script #2 Enable VMIC Provider Only)
    - Root PDC syncing time over NTP from its Hyper-V host (See Script #3 Enable NTP Client Only with manual server)

- In both scenarios, all computers in the forest besides the PDC are configured to sync from the AD hierarchy and not use VMIC. (See Script #4 Enable NTP Client Only with Domain Sync)
- In both scenarios, all the computers in the forest have W32time NTP Server enabled to remotely query their time (See Script #5 Enable NTP Server)
- The data streams specified earlier in the section are collected for each VM for a period of 1.5-2 days.
- The data collected and processed offline to generate the graphs shown in this doc.
    - TSC data from Hyper-V host and VM are matched up and the time offset is computed using the TimeSampleCorrelation tool (see Github link above).
    - Unfiltered NTP Offsets measurements are shown in the first chart for each VM.
    - In Server 2016 VMs, a "Popcorn filter" is applied to the NTP offsets to eliminate noisy samples and the resulting data is shown along with TSC data in a separate graph for each VM.
    - The data from different sources is aligned using the timestamps (which are down to the second) and excess data in any source is trimmed to ensure we are looking at common data from all sources.

Methodology for Simple NTP accuracy measurements

The AD measurements Scenario #2 involves client PDC syncing time from its time source over simple NTP, crossing a single virtual network switch. This is the best-case scenario for NTP synchronization.

We set up a physical machine to sync time from an accurate time source about a quarter mile away, across 6 network switches in a generic corporate environment. We used these scripts to set up this machine

- Script #1 Setting up common high accuracy settings
- Script #3 Enable NTP Client Only with manual server

Although this is dissimilar to the above scenario in many respects, it demonstrates the loss of accuracy in NTP with increase in network distance.

The measurement methodology in the both cases is the same.

1. Collect W32time stripchart data on the test machine, against its accurate time source.
2. Apply Popcorn filter on the NTP data and examine the resulting offset data.

Comparison between Server 2012R2 and Server 2016

Before we dive into the nitty gritty, let us look at side-by-side comparison charts of time sync performance between Server 2012R2 and Server 2016. These will showcase the improvements made in Server 2016 in near-best case scenarios in both forests.

There are many different comparisons that could be made even with the small test topologies. Only the interesting comparisons are being included here for clarity. The reader can perform other kinds of comparisons with the charts included in this document.

Most comparisons shown here are made between TSC offset data collected from both topologies.

We are using TSC data for simplicity purposes. One could make nearly identical comparisons between NTP data in 2012R2 and filtered NTP data in 2016.

The last comparison is between filtered NTP data, since we don't have TSC data to compare in that case.

Note that the data presented here was collected separately for each (topology + scenario) and not all at the same time.

Forest with a VMIC Root Time Source

This is a comparison of two scenarios – one with Server 2016 Forest and another with Server 2012R2 forest, where the Forest Root Domain PDCs are syncing time from an accurate Server 2016 Hyper-V host/time server through the VMIC time provider available inbox.

Server2016 performs significantly better in the first three comparisons between identical nodes in both the topologies. The last comparison in this scenario illustrates the performance of Server2016 Client in a Server2012R2 domain and compare it with Server 2016 Domain/Server2016 Client. You can see the impact of an inaccurate time source on Server2016 deployment here.

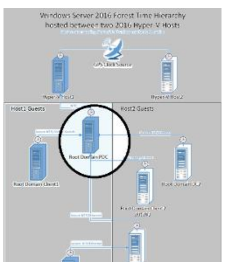


Figure 3 Root PDC Topology Reference

The following chart compares the time offset on 2016 Forest Root Domain PDC against 2012R2 Forest Root Domain PDC. (Stratum == 2)

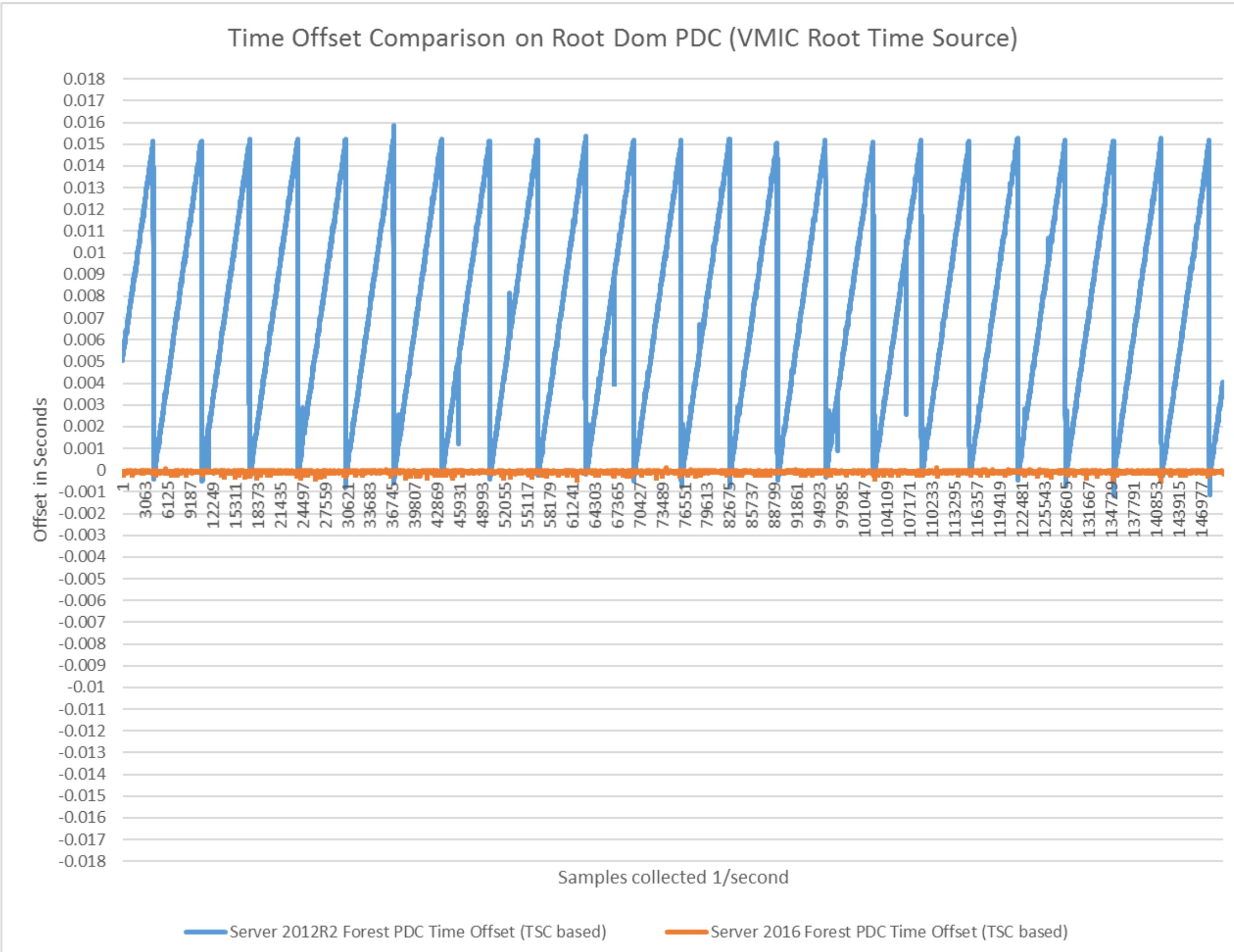


Figure 4 Root PDC Time sync comparison (VMIC Root Time Source)

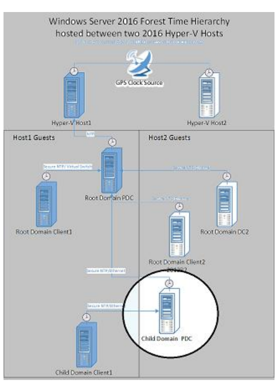


Figure 5 Child PDC Topology Reference

This chart compares the time offset on 2016 Child Domain PDC against 2012R2 Forest Child Domain PDC. (Stratum == 3)

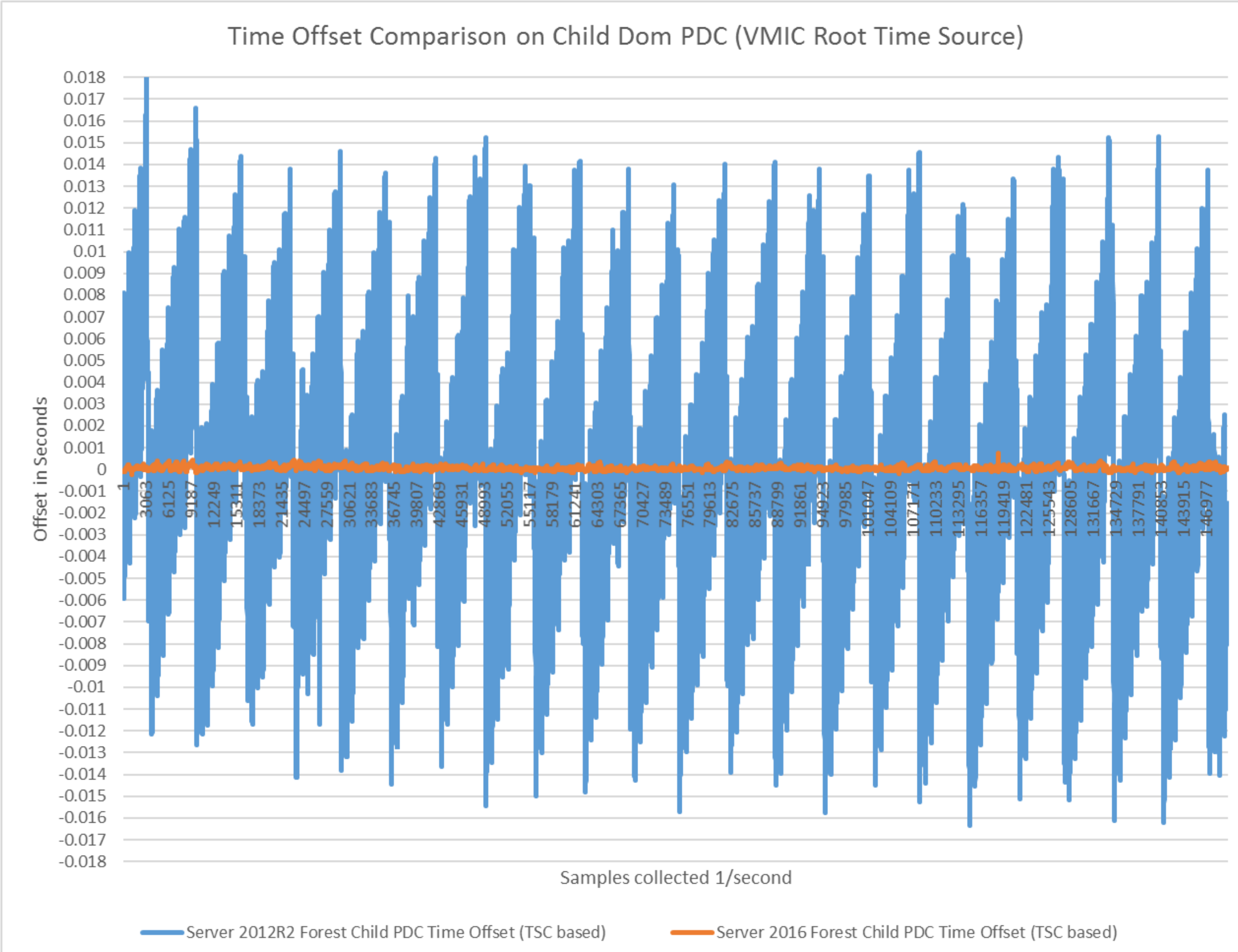


Figure 6 Child PDC comparison (VMIC Root Time Source)

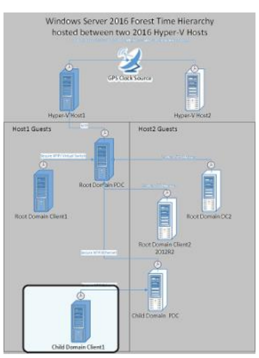


Figure 7 Child Domain Client Topology Reference

This chart compares the time offset on 2016 Forest Child Domain 2016 Client against 2012R2 Forest Child Domain 2012R2 Client. (Stratum == 4)

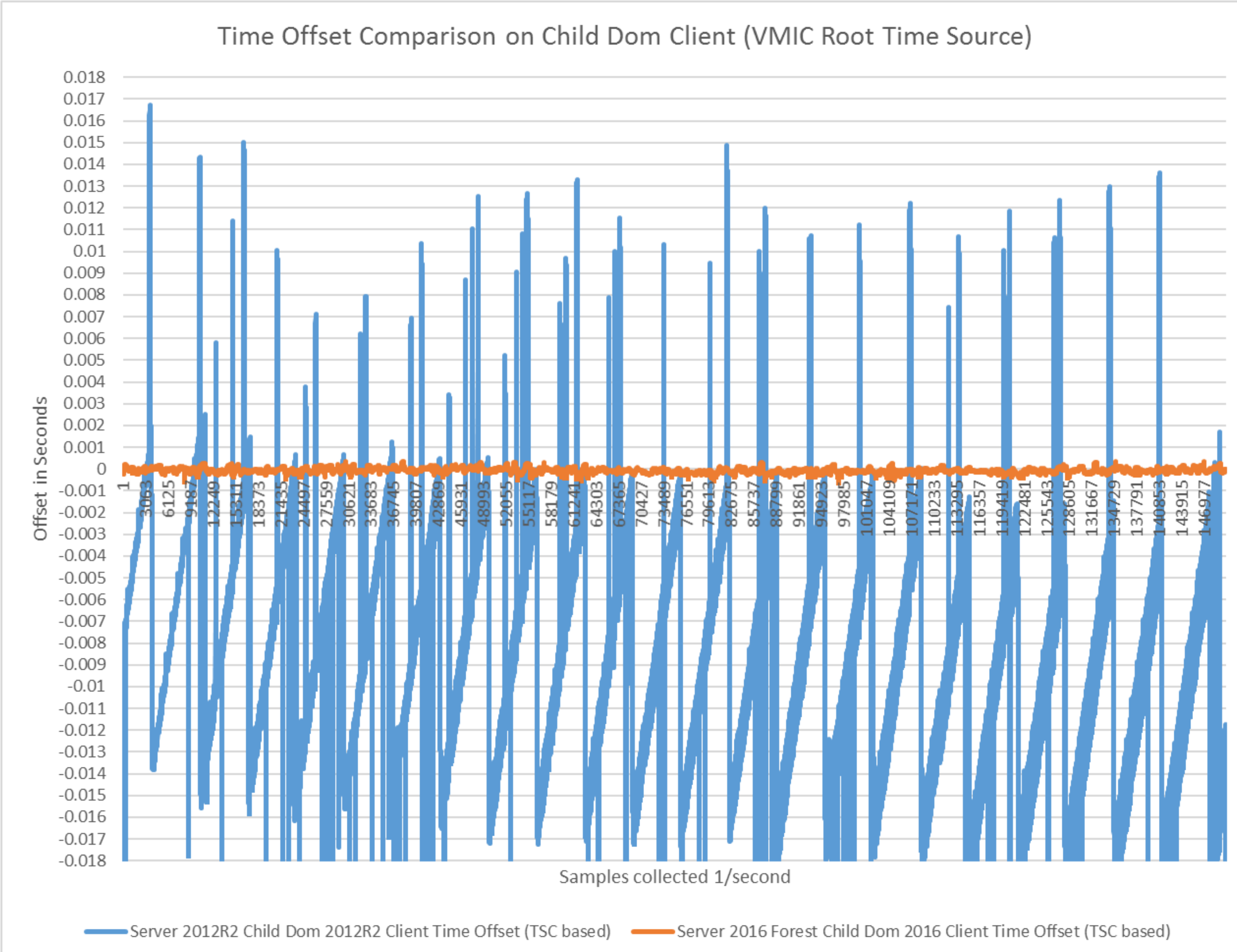


Figure 8 Child Domain Client Comparison (VMIC Root time source)

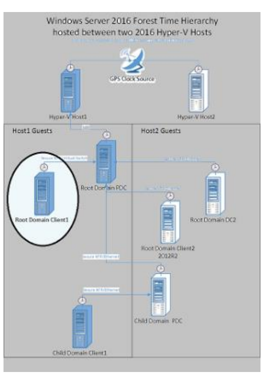


Figure 9 Server 2016 Client topology reference

The next chart compares the time offset on 2016 Client in a 2016 Forest Root Domain against a 2016 Client in a 2012R2 Forest Root Domain, essentially illustrating the importance of having an accurate time source for keeping reliable and accurate clock. (Stratum == 3)

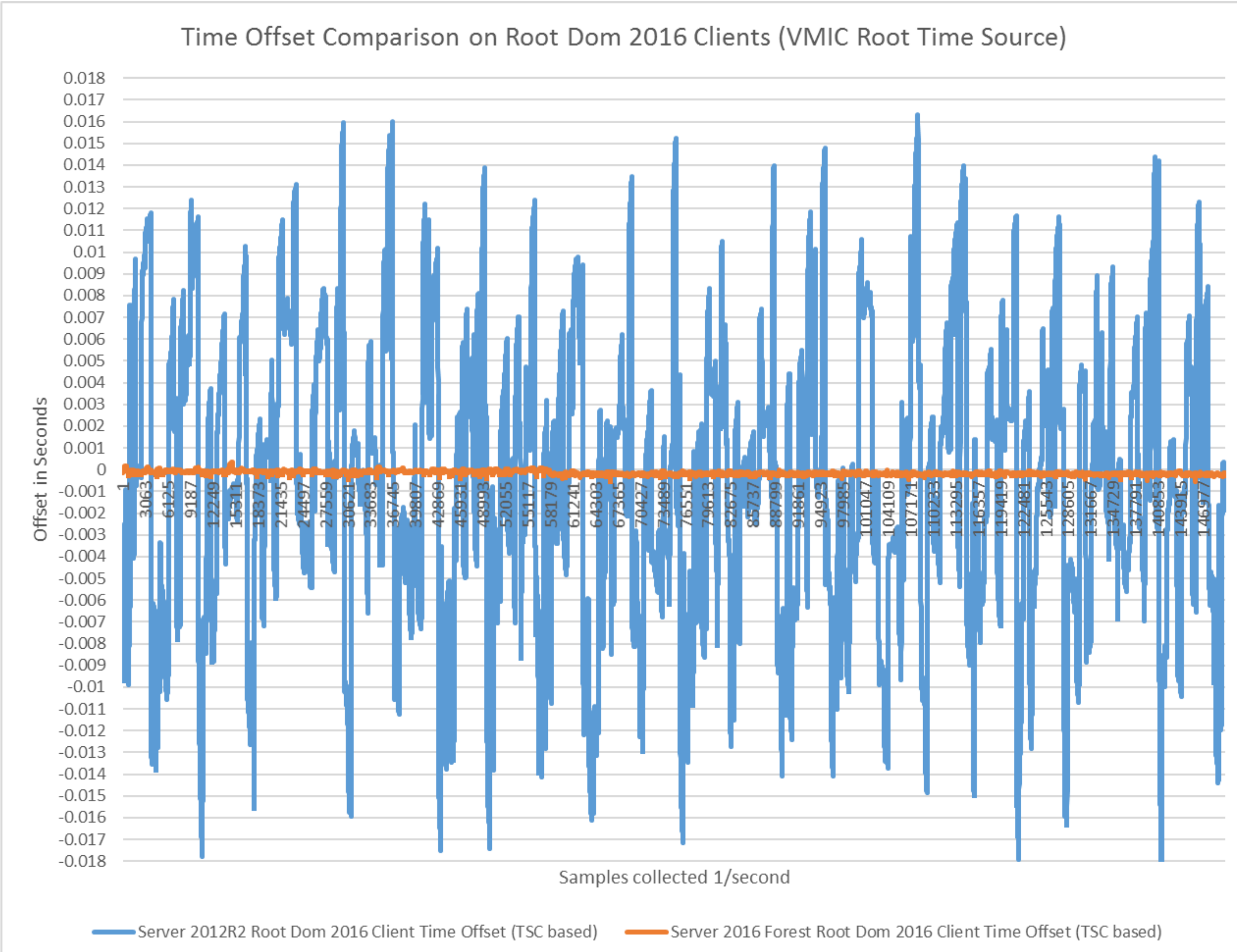


Figure 10 Server 2016 Client comparison (VMIC Root Time Source)

Forest with a NTP Root Time Source

The comparisons for this scenario are very similar to the comparisons in the prior scenario.. The main difference between the two sets of comparisons is that the time sync on Server2012R2 Root PDC in the current scenario is better than in the VMIC scenario. However, this is where the noticeable improvement stops in Server2012R2 forest. The rest of the comparisons look similar in both scenarios

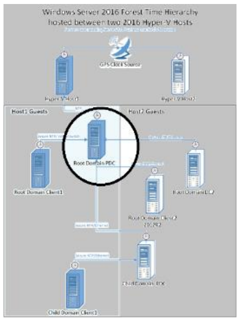


Figure 11 Root PDC Topology Reference

This chart compares the time offset on 2016 Forest Root Domain PDC against 2012R2 Forest Root Domain PDC. The latter performs better with a NTP time source vs a VMIC time source. (Stratum == 2)

Figure 12 Root PDC Comparison (NTP Root Time Source)

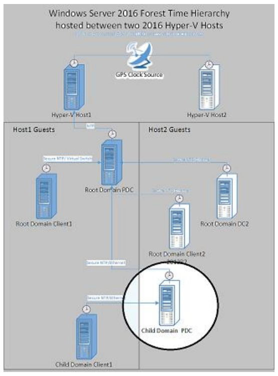


Figure 13 Child PDC Topology Reference

The next chart compares the time offset on 2016 Child Domain PDC against 2012R2 Forest Child Domain PDC. The improvement in Root PDC is not passed on to the Child PDC in Server 2012R2 scenario. (Stratum == 3)

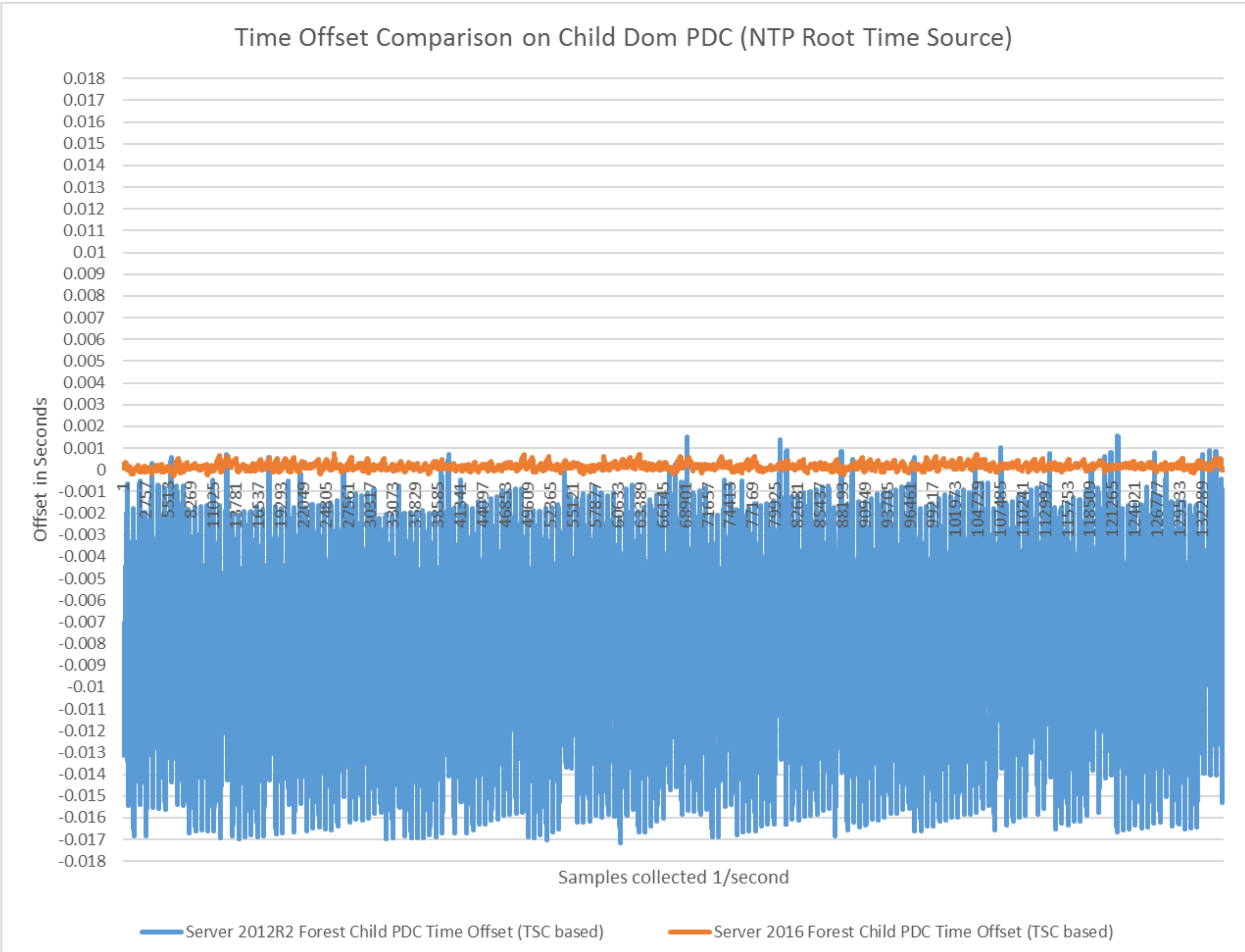


Figure 14 Child PDC Comparison (NTP Root Time Source)

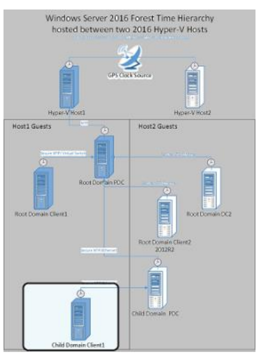


Figure 15 Child Domain Client Topology Reference

This chart compares the time offset on 2016 Forest Child Domain 2016 Client against 2012R2 Forest Child Domain 2012R2 Client. (Stratum == 4)

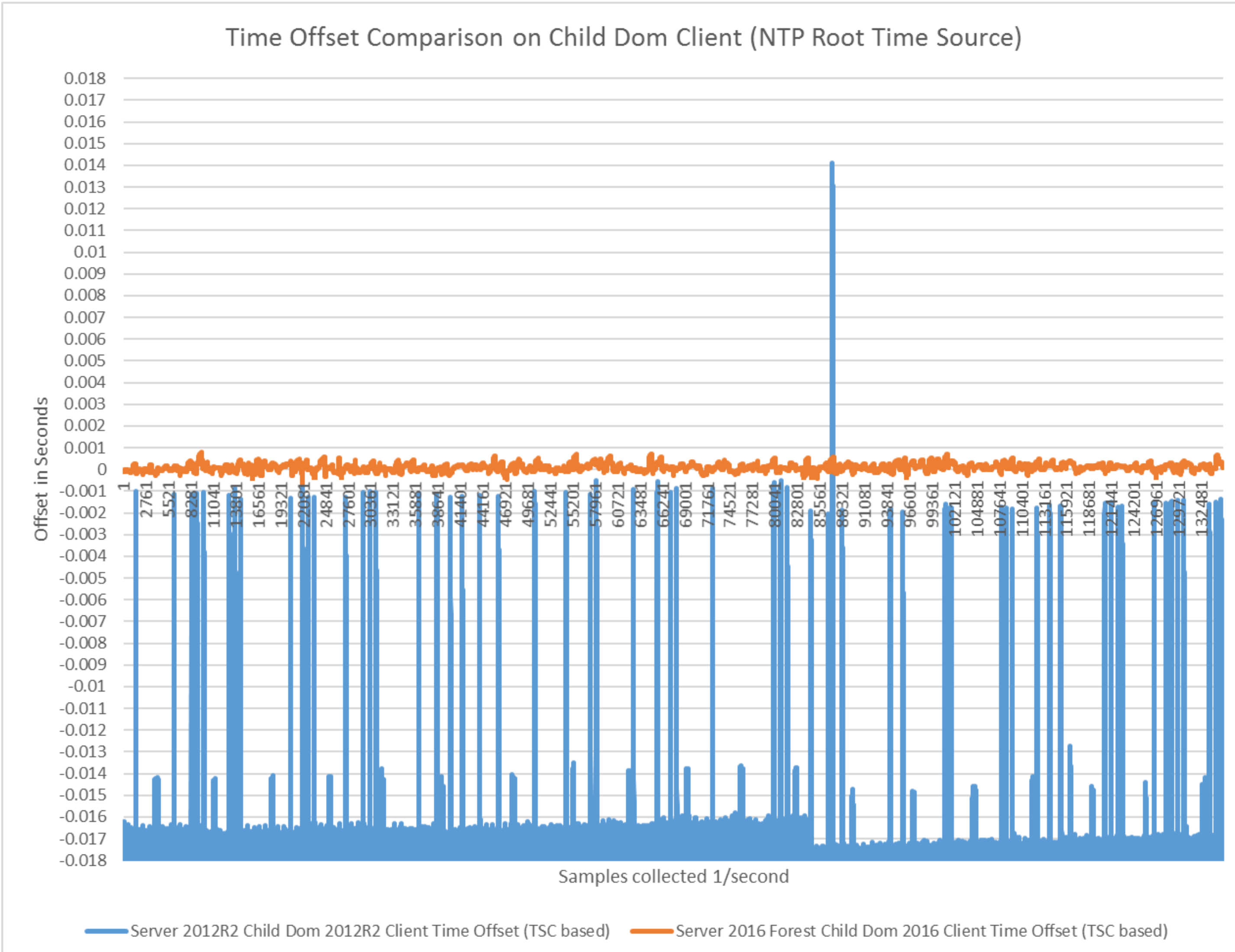


Figure 16 Child Dom Client Comparison (NTP Root Time Source)

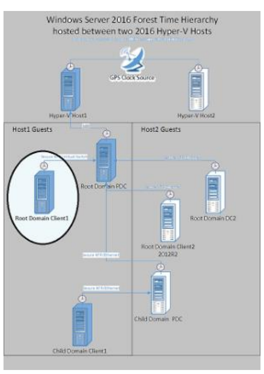


Figure 17 Server 2016 Client Topology Reference

The next chart compares the time offset on 2016 Client in a 2016 Forest Root Domain against a 2016 Client in a 2012R2 Forest Root Domain and the results are like the same comparison in the VMIC topology scenario. (Stratum == 3)

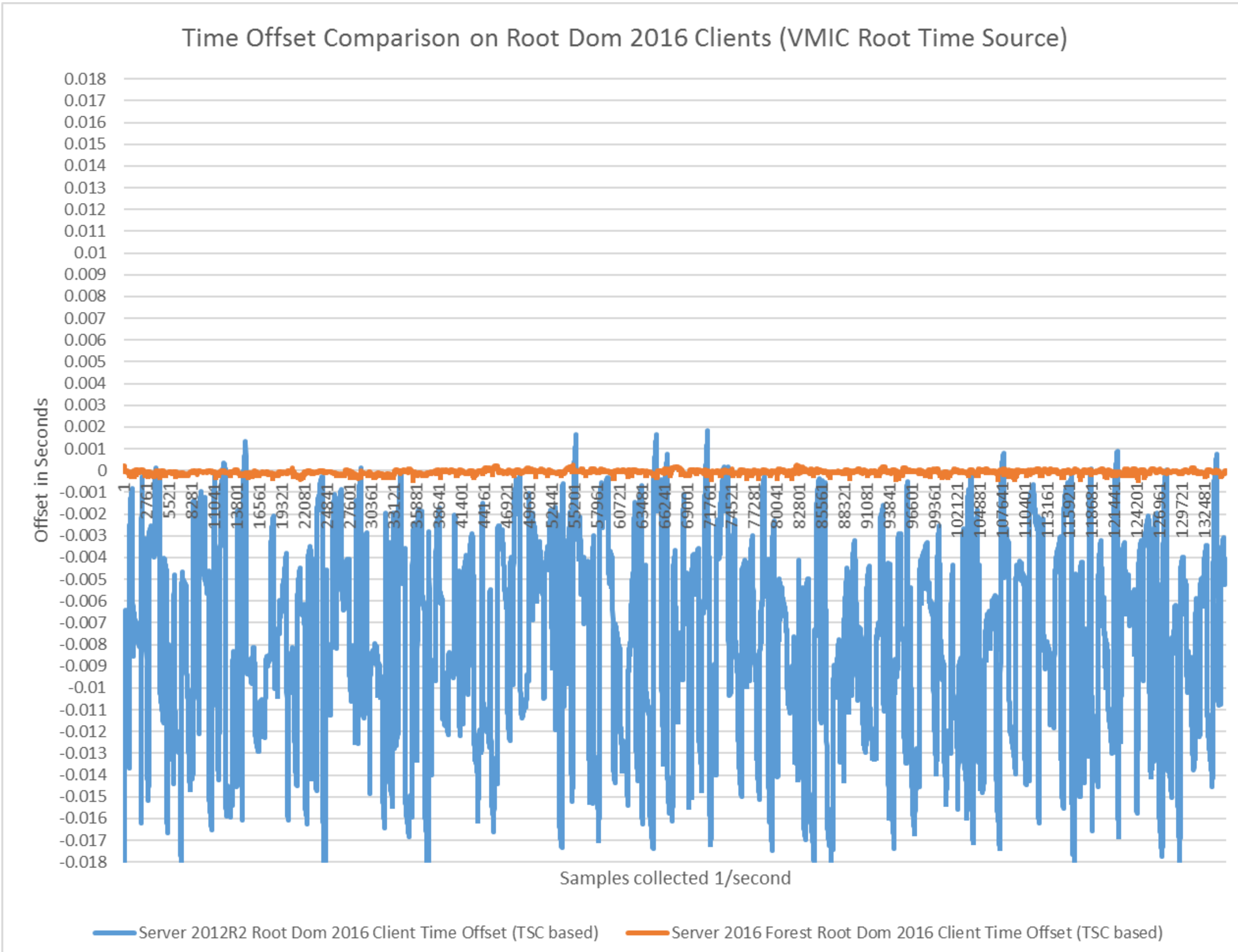


Figure 18 2016 Client Comparison (NTP Root Time Source)

The key takeaway from these comparisons is that Server 2016 DCs provide a vast improvement in time sync accuracy over Server2012R2 and it is possible to get 1ms accuracy up to 3 NTP strata away from a high accuracy time source.

The following sections go into a lot of detail about the measurements obtained from each scenario, compare the three different ways to measure time offsets and show a simple way to filter out noise inherent to NTP for analysis purposes. The charts provided are scaled similarly (as much as possible) to allow you to perform other kinds of comparisons not done in this section.

Comparison of NTP in Server 2016

This is a direct comparison of NTP time synchronization performance in Server 2016 between the case where there is only one virtual network switch in the path vs the case when there are 6 network switches in the way. The data was collected from unrelated networks at different times. The key takeaway here is that the time sync accuracy degrades as the "network distance" increases. This is a critical factor to be taken into account for planning time distribution in a given network topology.

The data shown here is the filtered NTP offset data from the Server 2016 Root PDC w/NTP sync and the NTP Quarter mile test cases. (Stratum == 2)

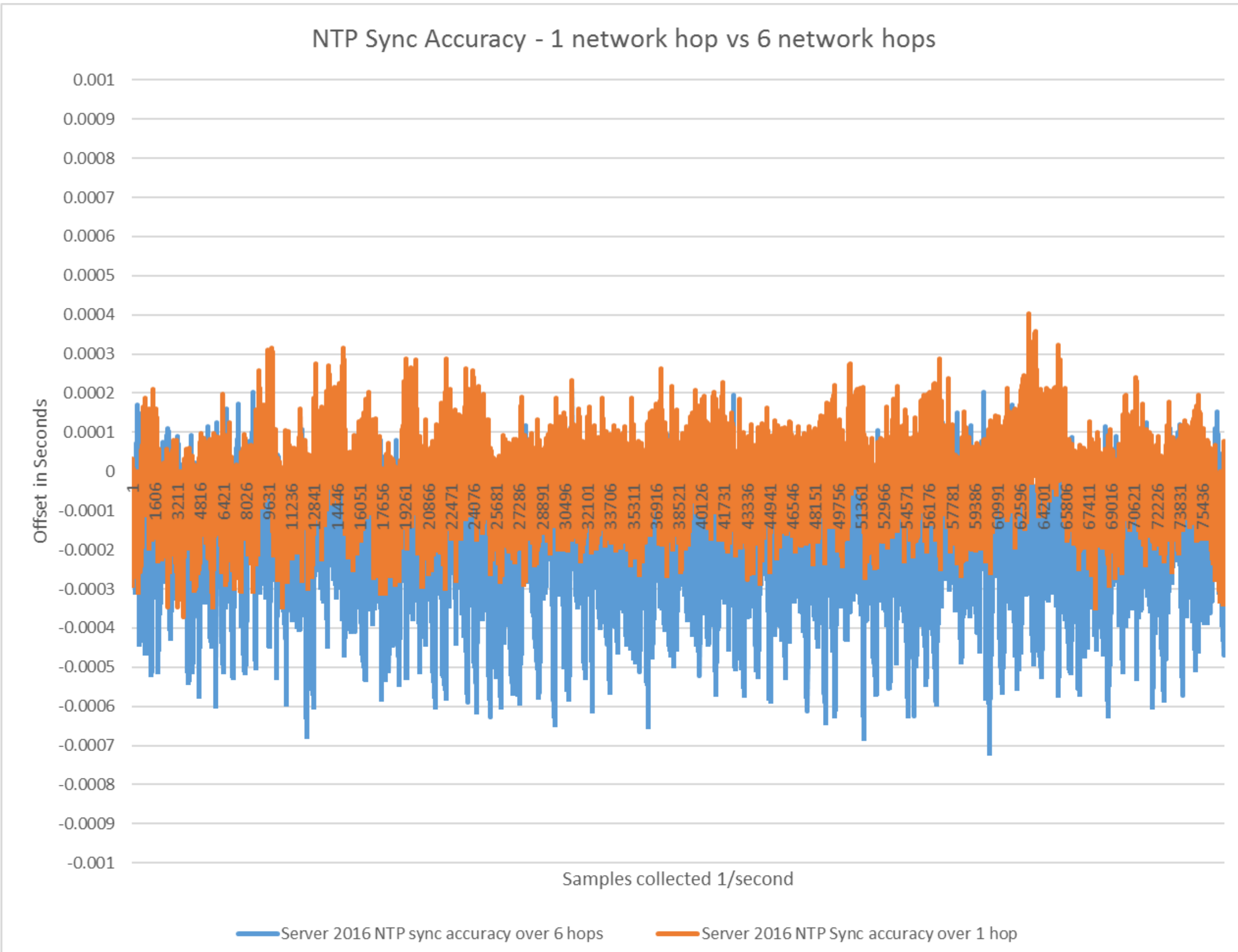


Figure 19 Simple NTP accuracy comparison in Server 2016 (1 hop vs 6 hops)

Note that simple NTP is used to sync time from manually configured NTPServers.

Server 2016 Forest using VMIC root time source (Scenario #1)

In this scenario, the Server 2016 Root PDC of the forest is syncing its time using VMIC provider from its Hyper-V host machine. The host partition's clock is synchronized to a very high accuracy using Sync-n-Scale hardware.

Root PDC (Server 2016)

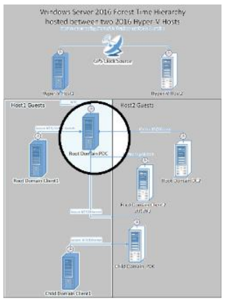


Figure 20 Topology Reference

You can clearly see that the TSC based offset calculations indicate the time offset to be well within 1ms range here. The NTP data in this chart looks noisy like before and we have to put it through the Popcorn filter.

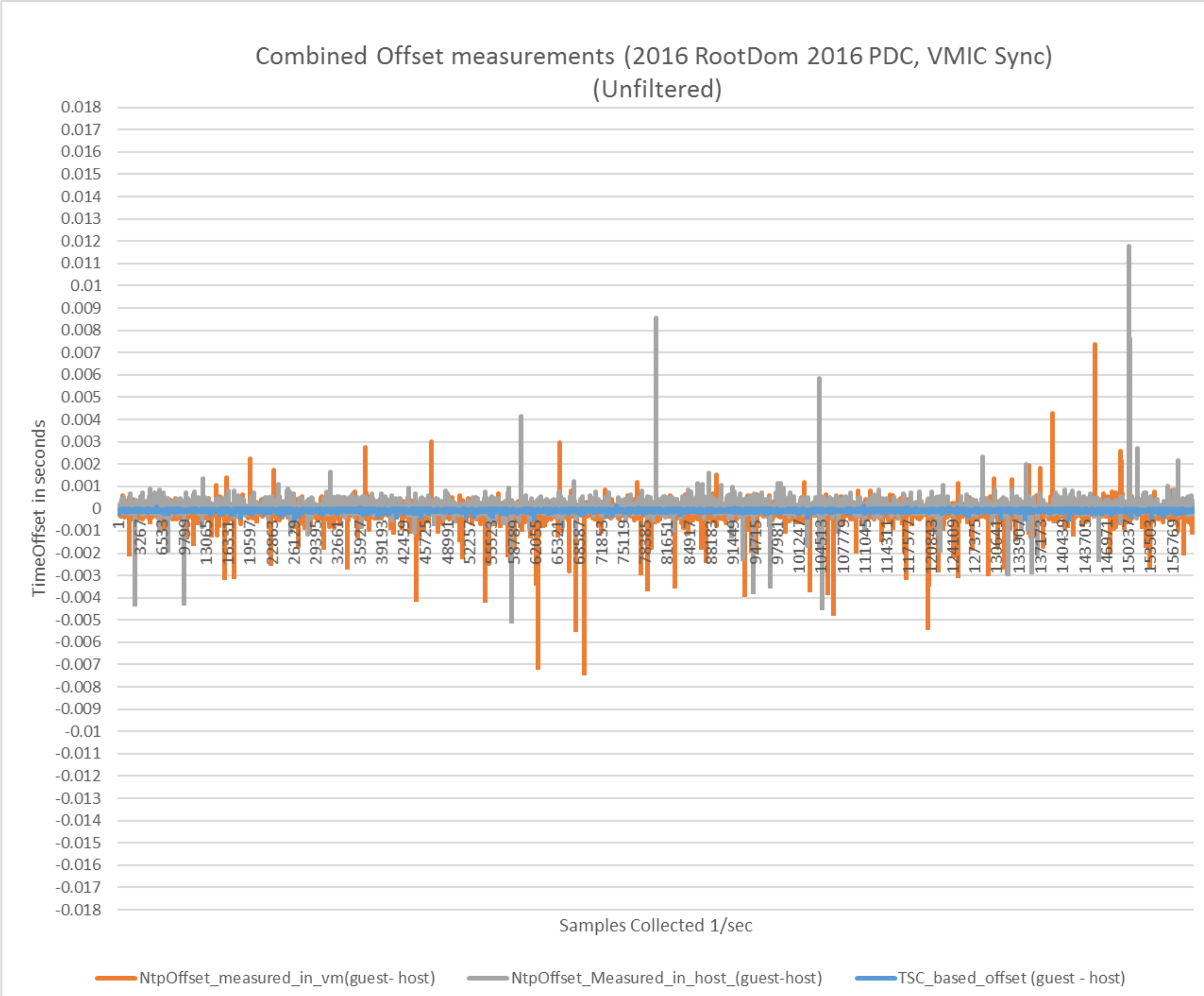


Figure 21 Offset Times - Server 2016 Root Dom 2016 PDC (VIMIC scenario)

Below are the histograms for the NTP roundtrip times for VM->Host measurements and Host -> VM measurements. If we apply the Popcorn filter on each of these data sets, a small (~1%) number of samples with very large roundtrip times (RTTs) in each data set will be excluded.

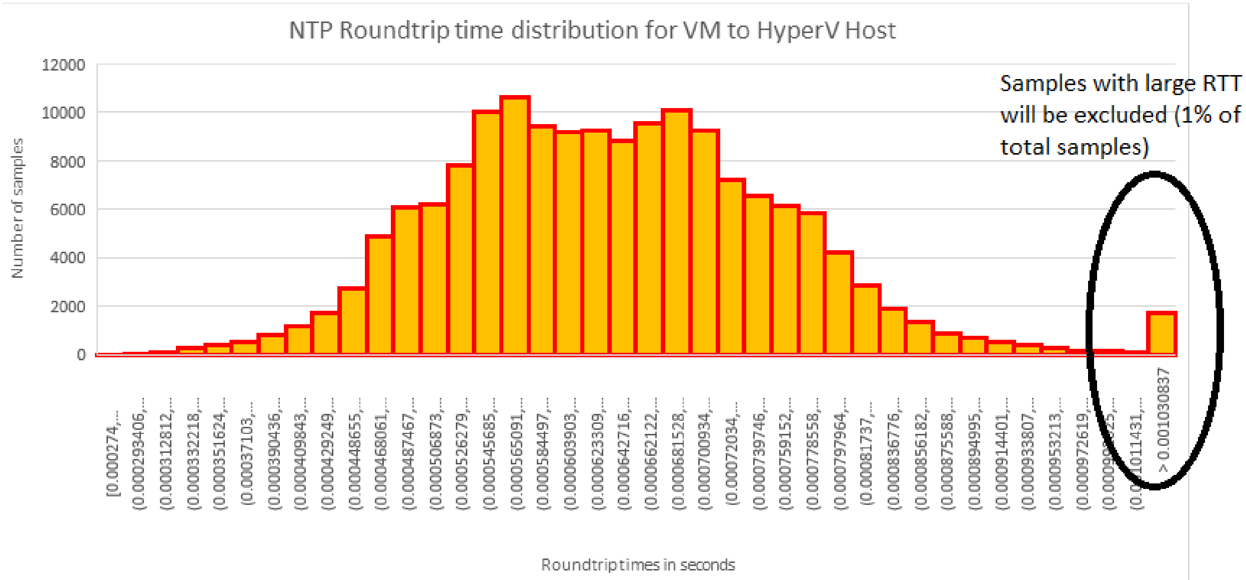


Figure 22 NTP Ping Roundtrip time distribution (VM to Hyper-V Host), Server 2016 Root Dom 2016 PDC (VMIC scenario)

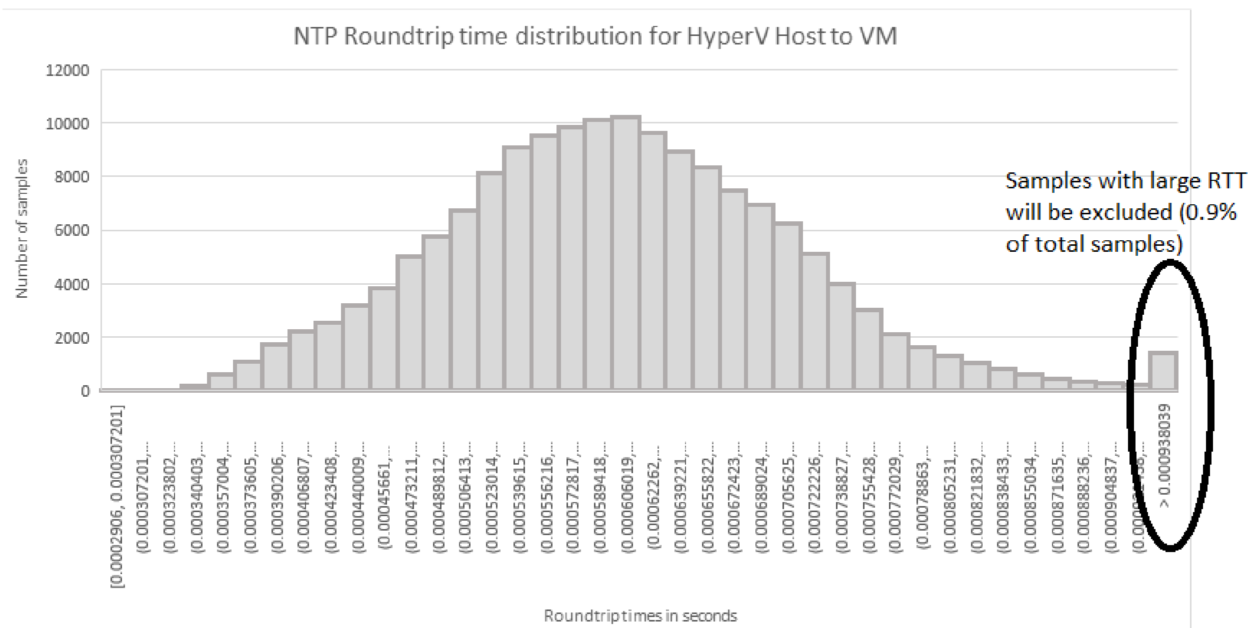


Figure 23 NTP Ping Roundtrip time distribution (Hyper-V Host to VM), Server 2016 Root Dom 2016 PDC (VMIC scenario)

Below are the charts with the NTP offset data through the Popcorn filter.

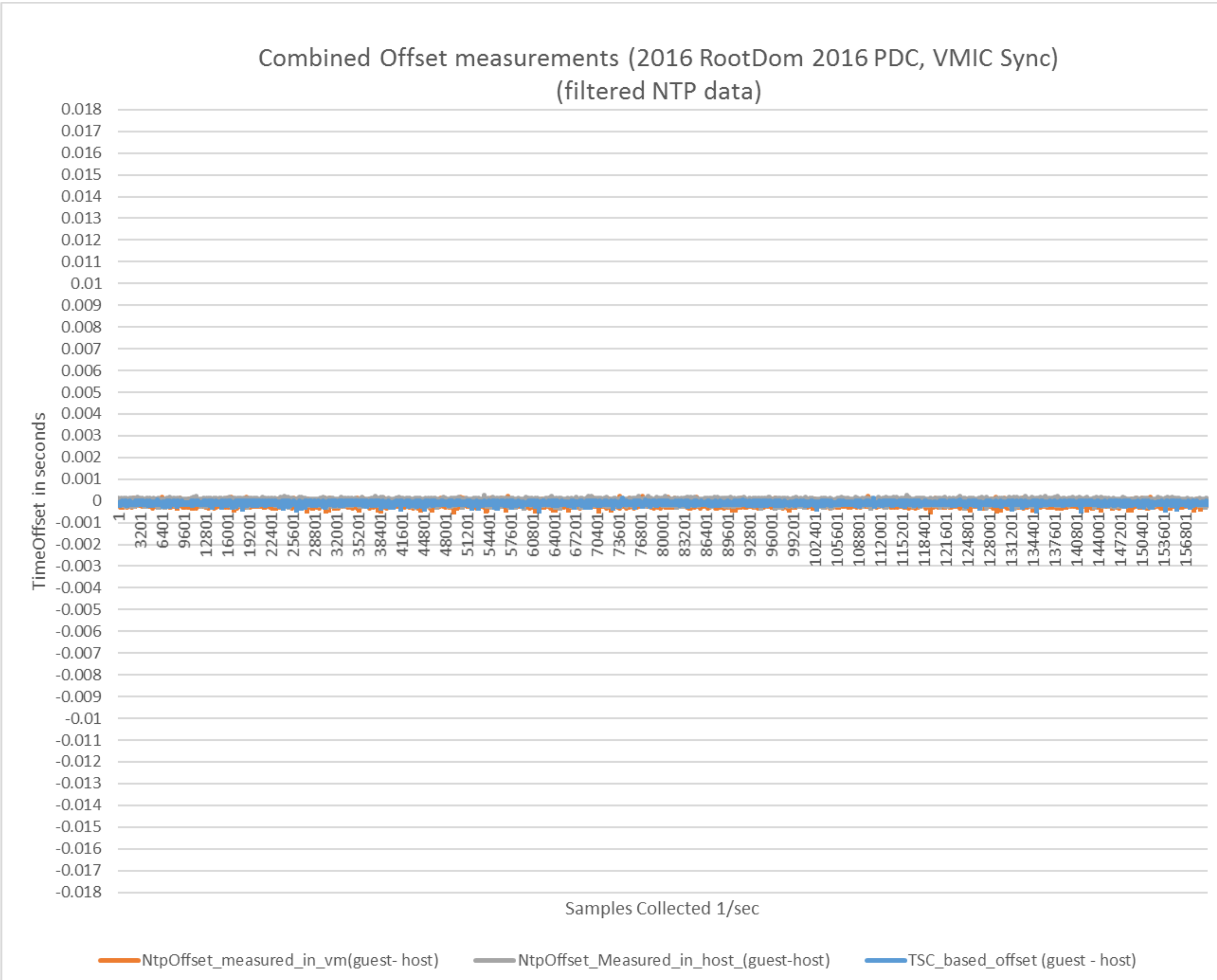


Figure 24 Filtered Offset Times - Server 2016 Root Dom 2016 PDC (VIMIC scenario)

This is the same chart, zoomed in for a better look at the values. Note that the offset measured by the 3 methods is within several hundred microseconds range off of zero.

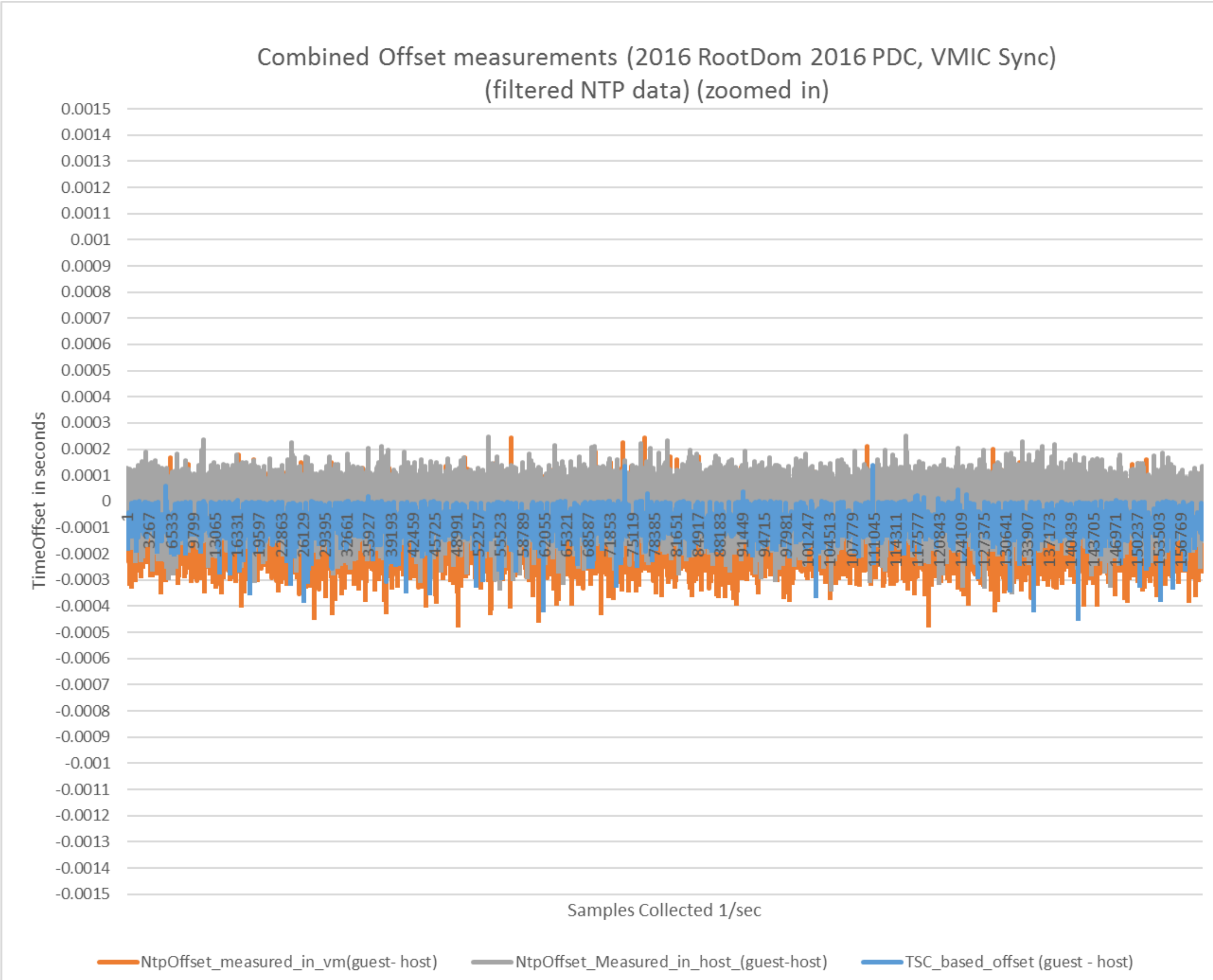


Figure 25 Filtered Offset Times - Server 2016 Root Dom 2016 PDC (VIMIC scenario) (Zoomed in)

Root Domain Replica DC (Server 2016)

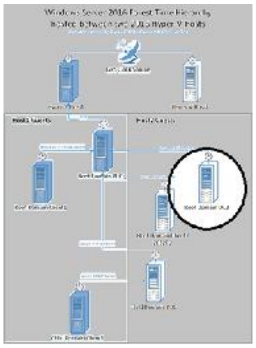


Figure 26 Topology Reference

The root domain replica DC syncs time from the PDC using secure NT5DS protocol. Its performance in this scenario is comparable/closely matches the data recorded in the other Server2016 Forest scenario.

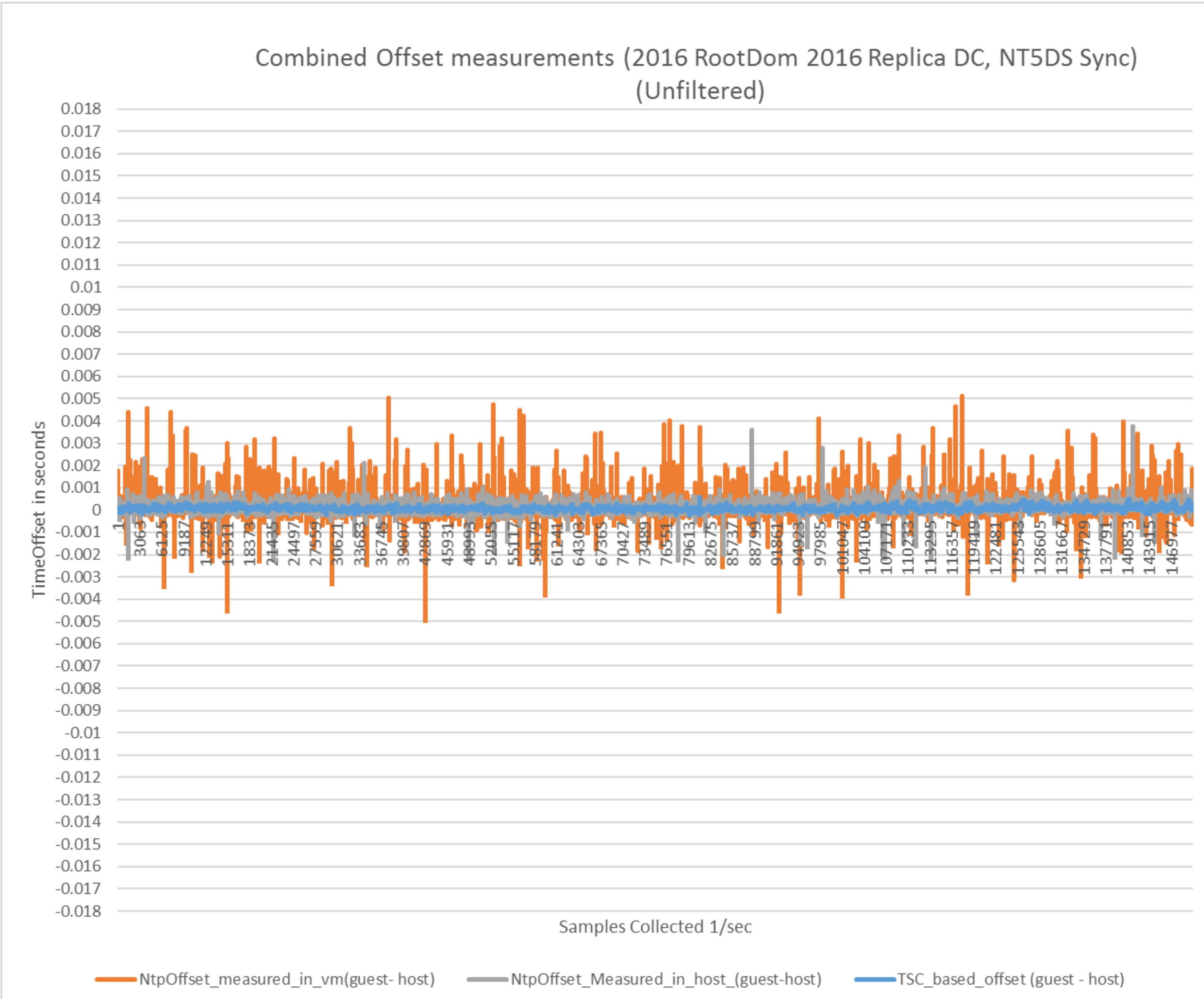


Figure 27 Offset Times - Server 2016 Root Dom 2016 Replica DC (VIMIC scenario)

Applying the Popcorn Filter results in the exclusion of 2% of the NTP ping samples recorded in the VM and 1% of the NTP ping samples recorded in the Hyper-V host.

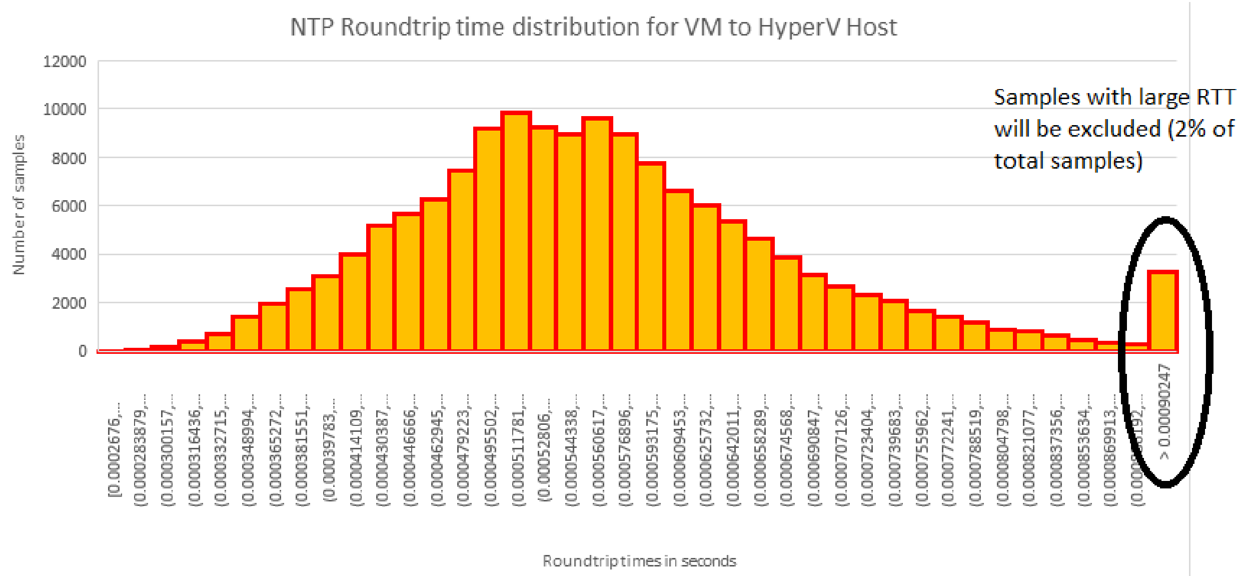


Figure 28 NTP Ping Roundtrip time distribution (VM to Hyper-V Host), Server 2016 Root Dom 2016 Repl DC (VMIC scenario)

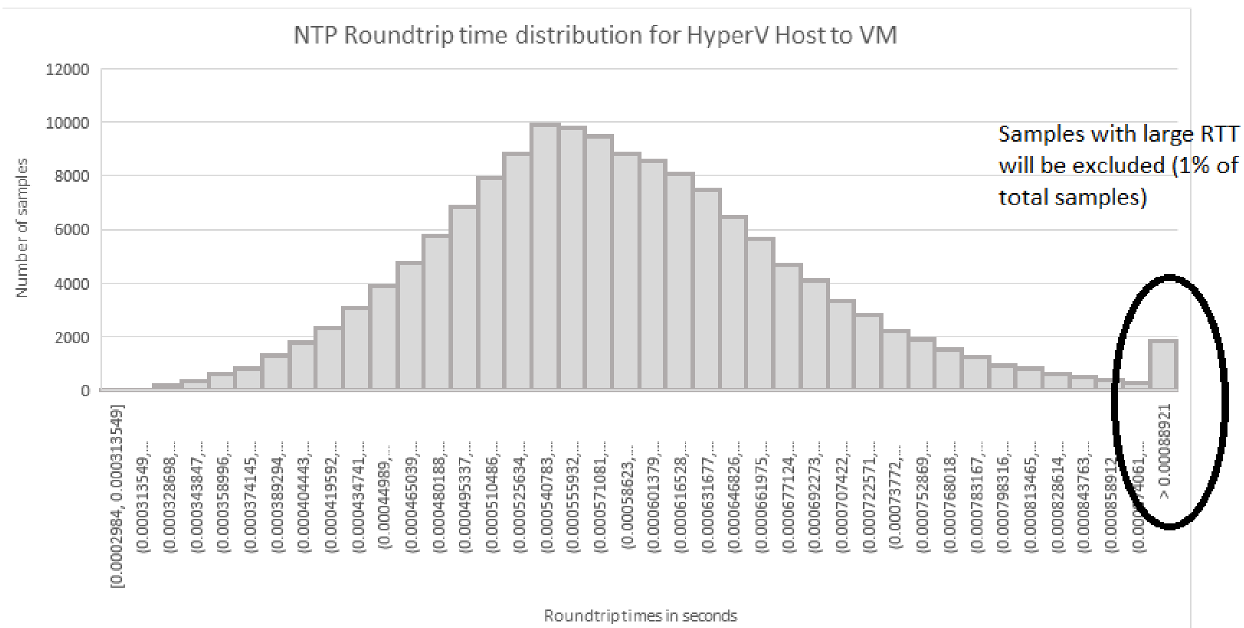


Figure 29 NTP Ping Roundtrip time distribution (Hyper-V Host to VM), Server 2016 Root Dom 2016 Repl DC (VMIC scenario)

After the filtering, the resulting chart shows the 3 data sets matching closely and staying below 1ms range.

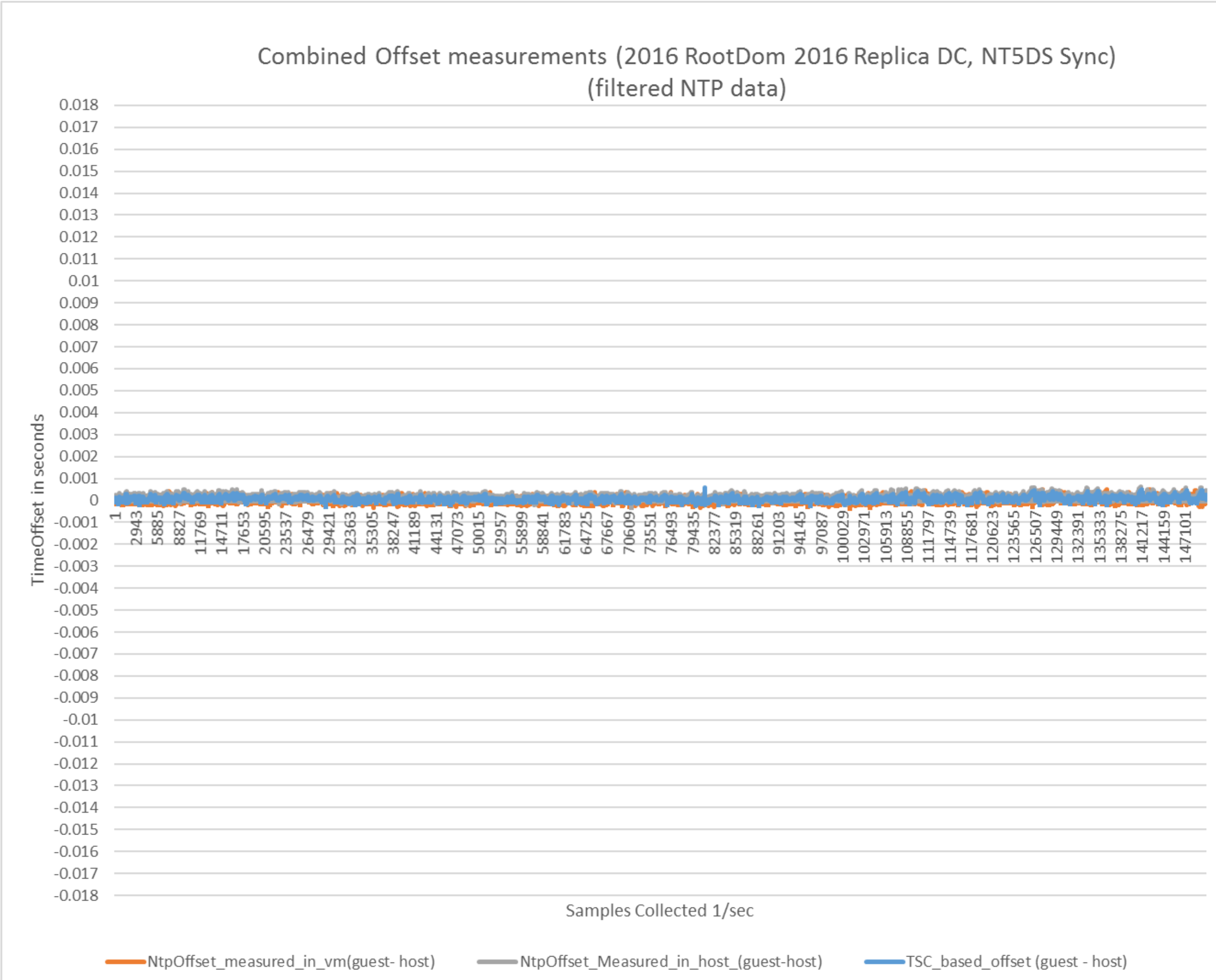


Figure 30 Filtered Offset Times - Server 2016 Root Dom 2016 Replica DC (VIMIC scenario)

This is a zoomed in view of previous chart, showing a time sync accuracy within 300 microseconds of zero.

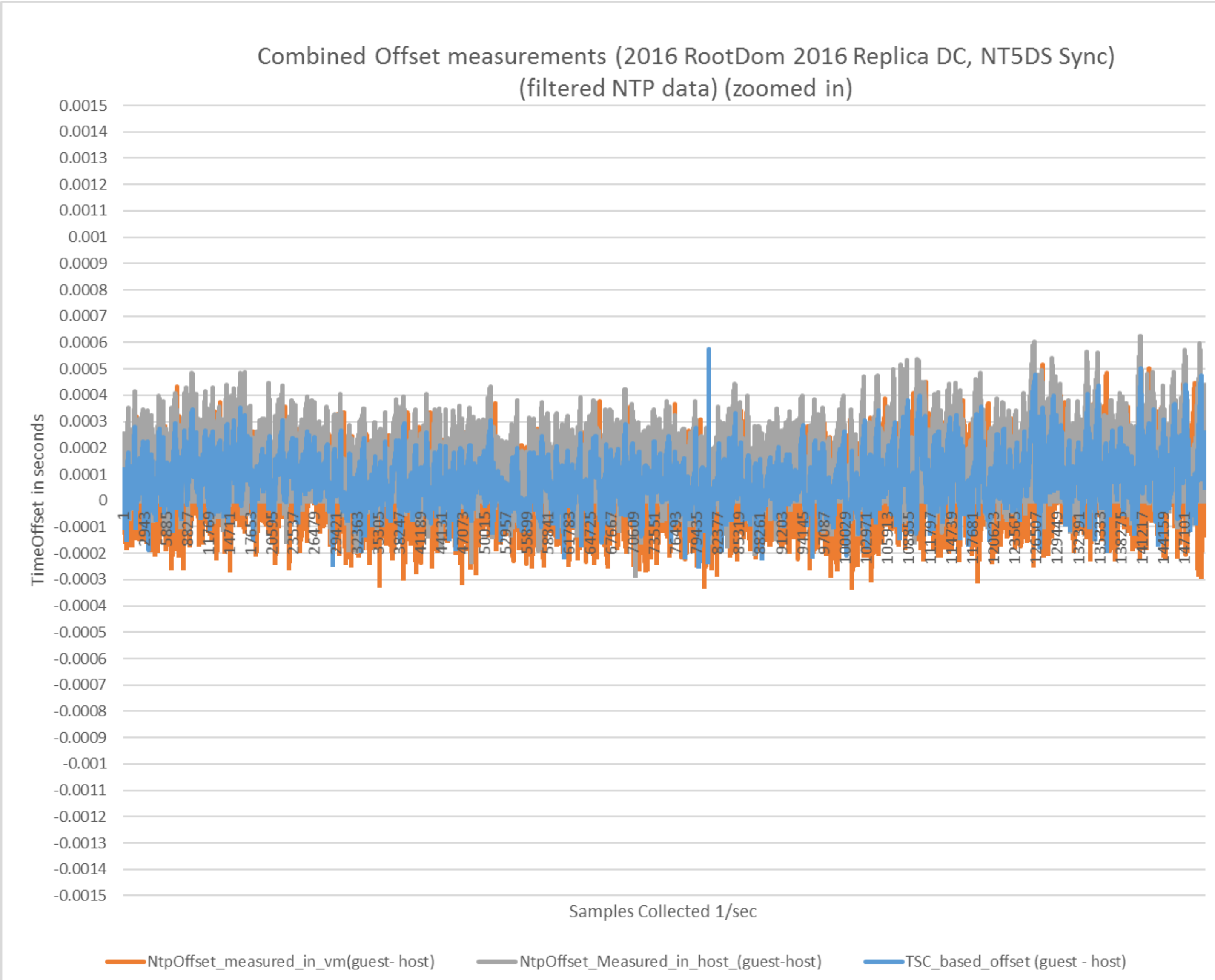


Figure 31 Filtered Offset Times - Server 2016 Root Dom 2016 Replica DC (VIMIC scenario) (Zoomed in)

Root Domain Client #1 (Server 2016)

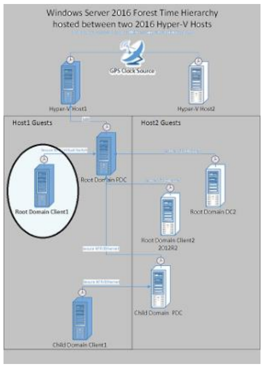


Figure 32 Topology Reference

A Server2016 domain client (a.k.a. member server) in the root domain is similar scenario when compared to the Replica DC. The main difference is that it syncs time over the virtual network instead of crossing the Ethernet.

(Note: In this particular case, one set of NTP ping measurements from Hyper-V host to the VM are not shown)

You would still observe that the two sets of data for this scenario closely match each other and they indicate that the time is kept within 1ms of accuracy.

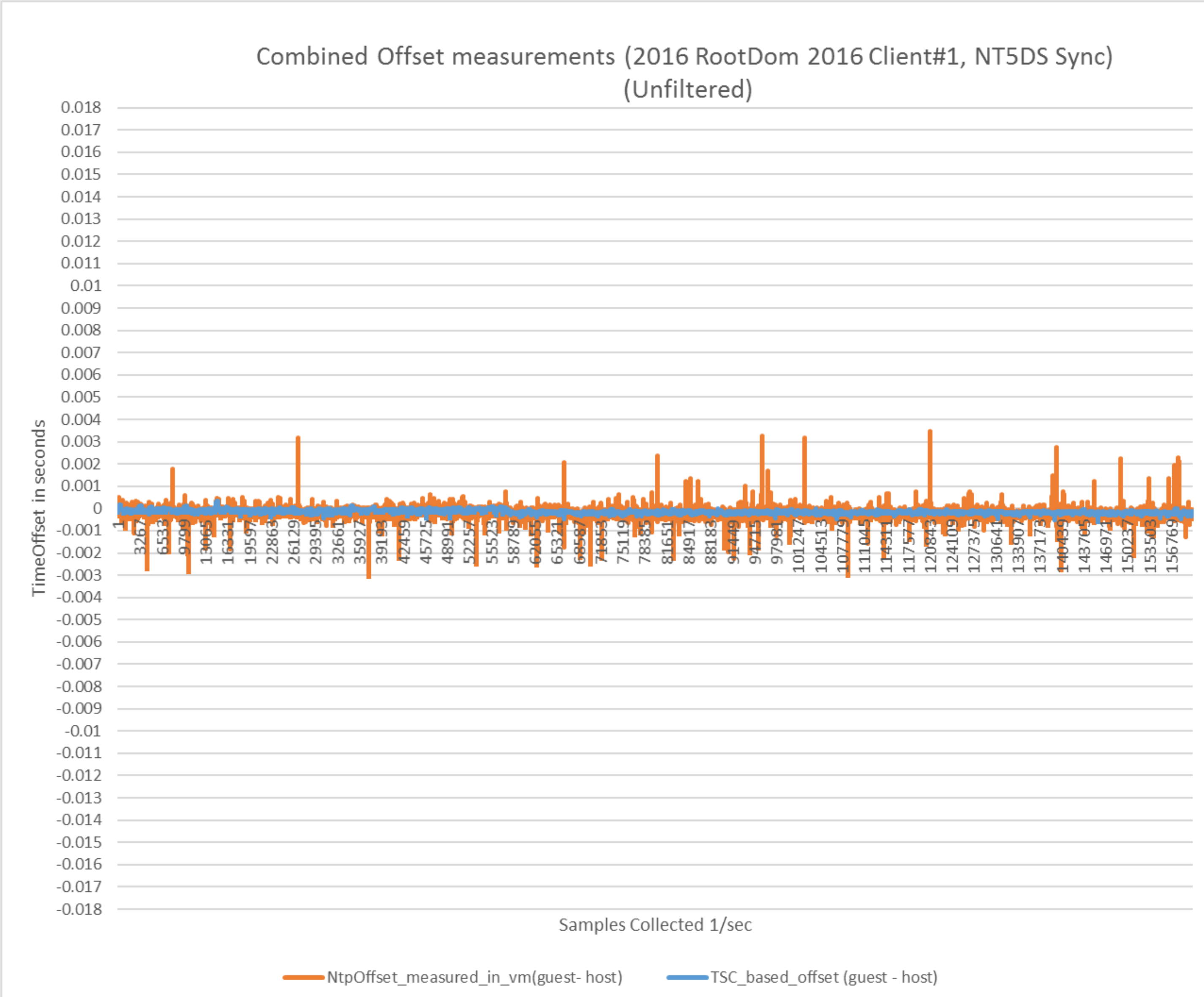


Figure 33 Offset Times - Server 2016 Root Dom 2016 Client (VIMIC scenario)

The Popcorn Filter catches 1% of the NTP ping samples measured inside the VM.

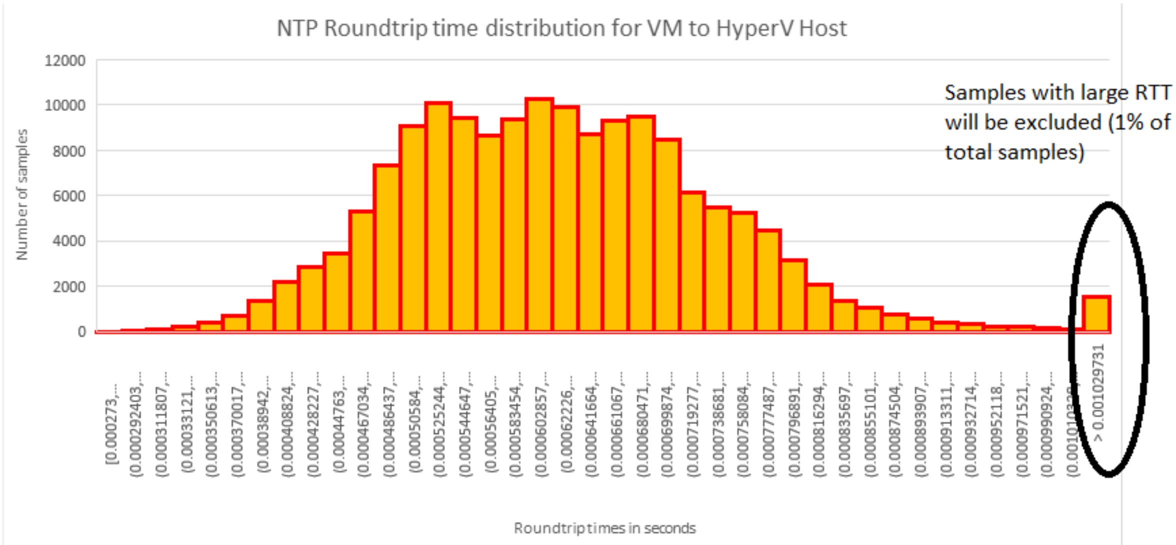


Figure 34 NTP Ping Roundtrip time distribution (VM to Hyper-V Host), Server 2016 Root Dom 2016 Client (VMIC scenario)

Here is the filtered data:

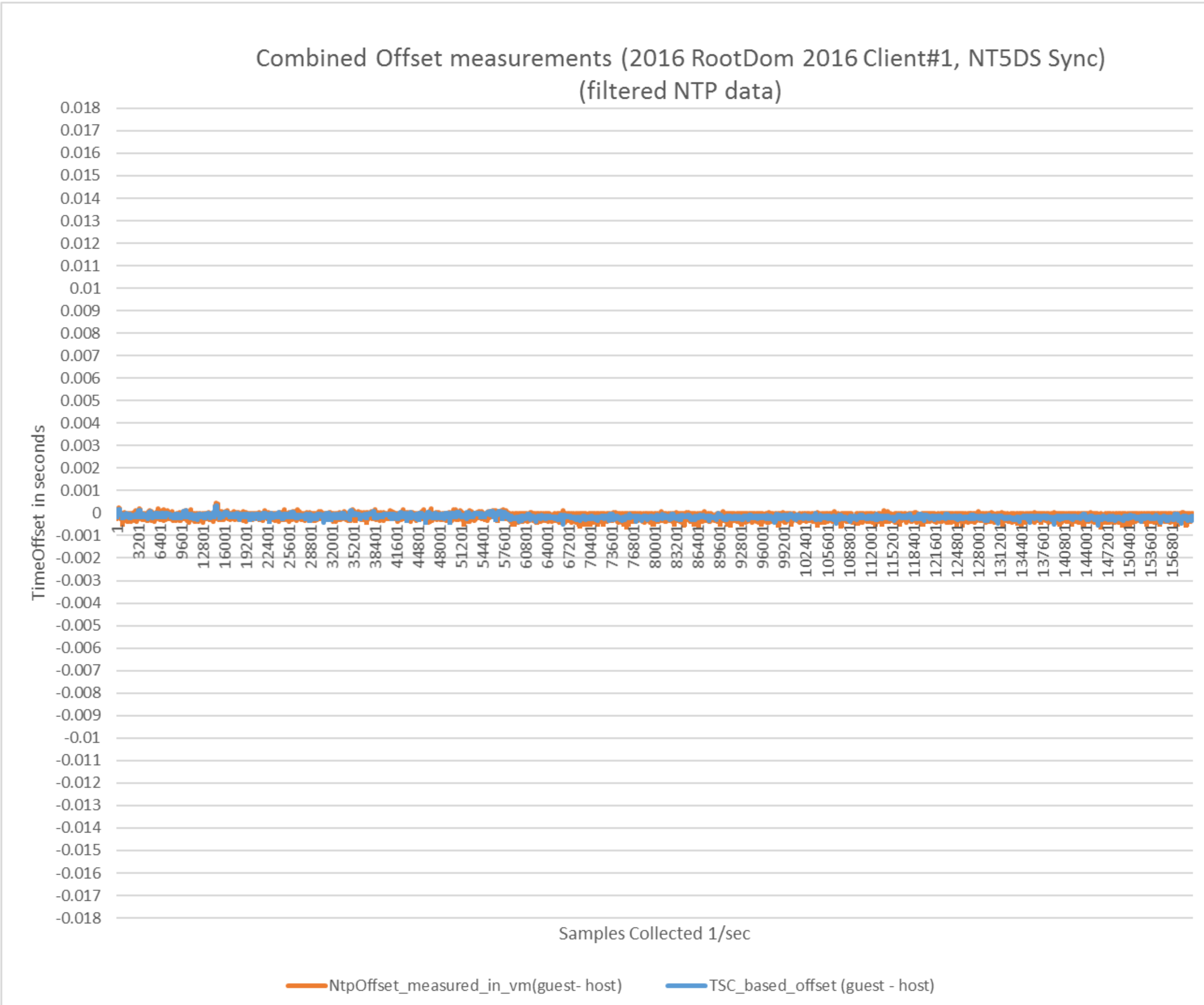


Figure 35 Filtered Offset Times - Server 2016 Root Dom 2016 Client (VIMIC scenario)

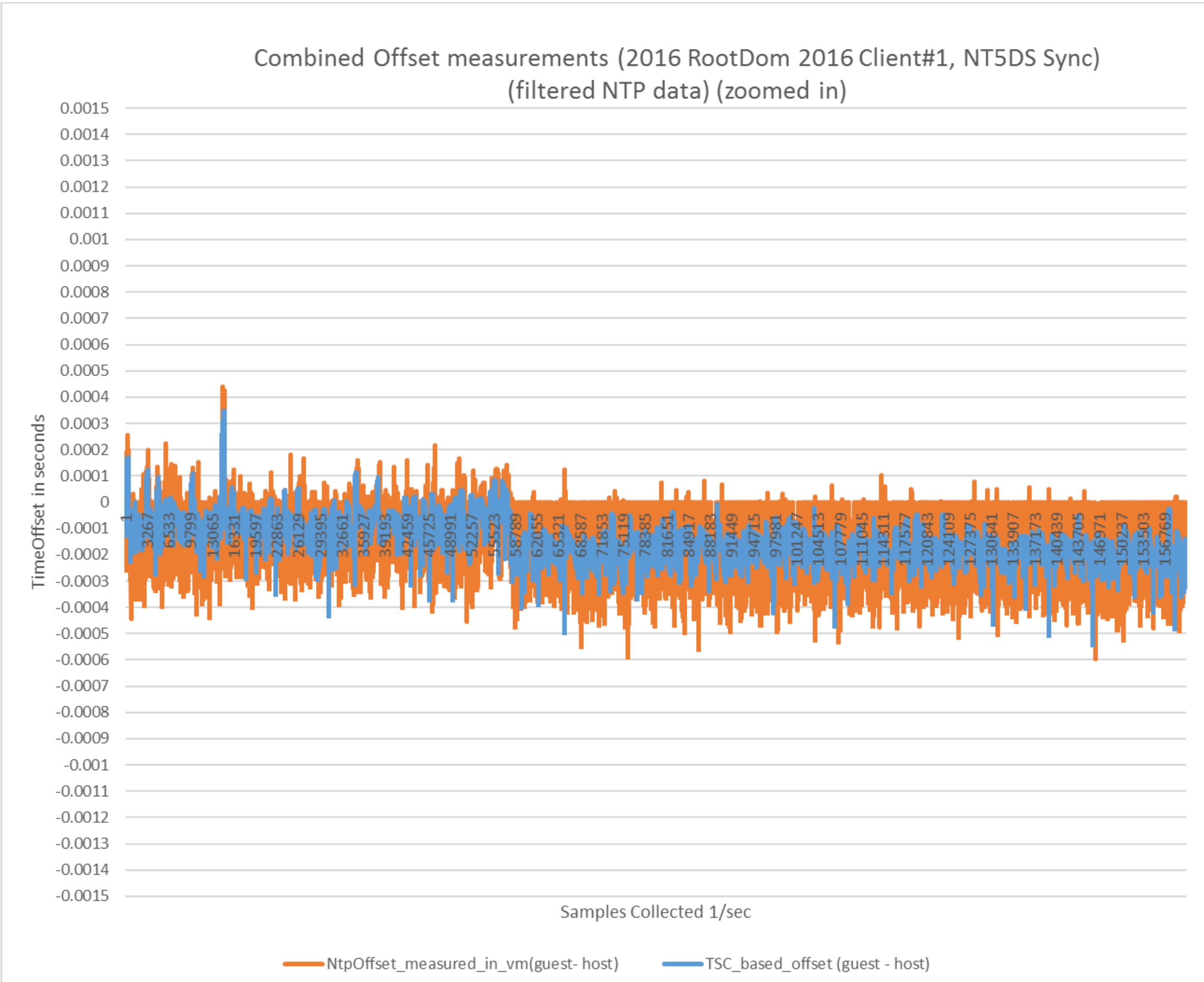


Figure 36 Filtered Offset Times - Server 2016 Root Dom 2016 Client (VIMIC scenario) (Zoomed in)

Root Domain Client #2 (Server 2012R2)

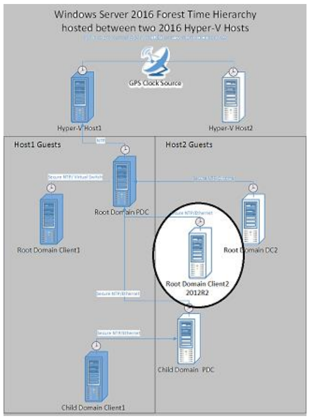


Figure 37 Topology Reference

The goal again here is to measure the time sync performance of a Server2012R2 client in a Server2016 forest environment. This time around, the client seems to keep its time with an offset of [-1, 3] milliseconds, but measuring it using NTP from the Hyper-V host is still off of this value by about 10 milliseconds. This discrepancy in NTP pings matches the Server2012R2 behavior observed in other scenarios as well. No noise filtering was applied in this case.

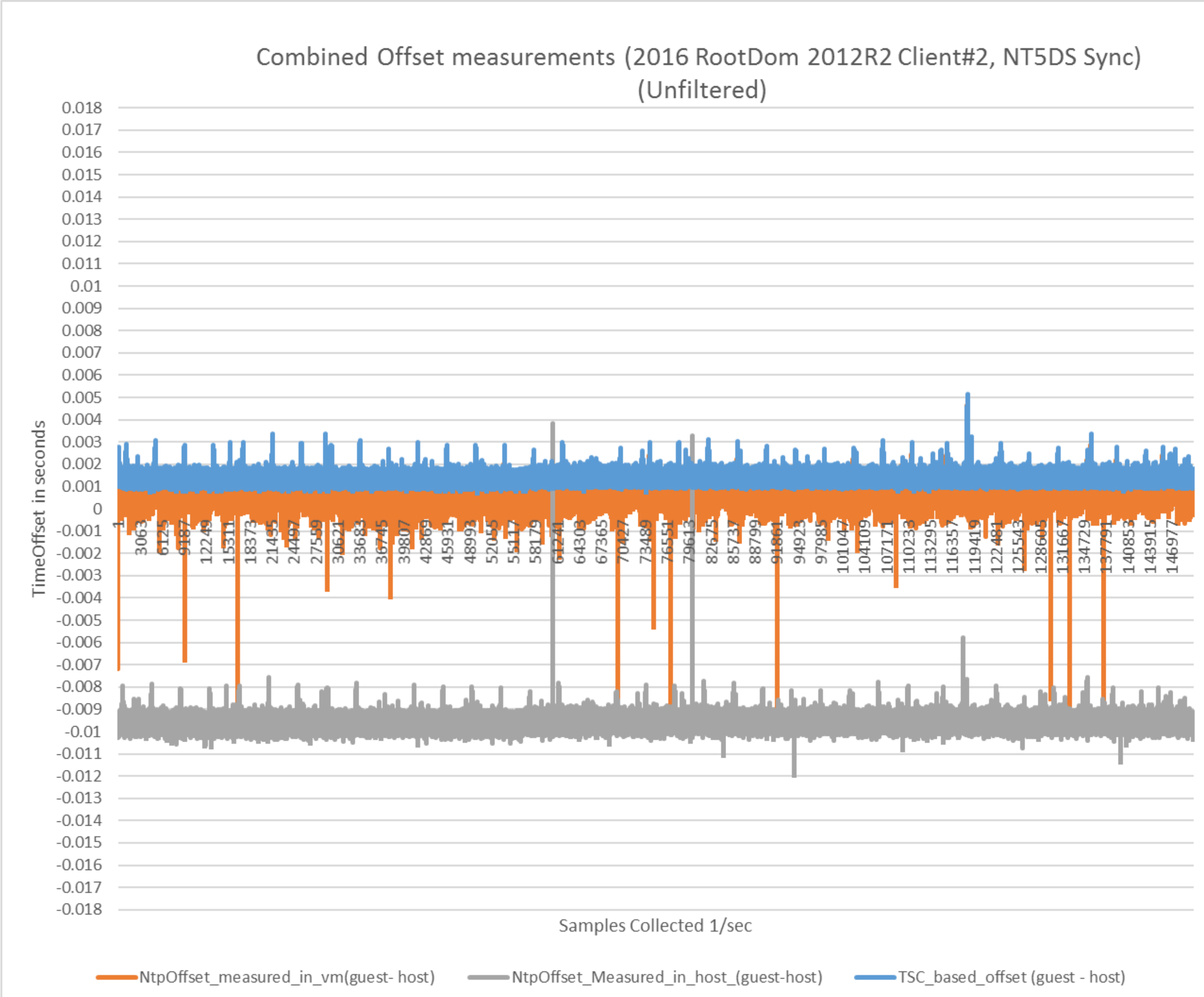


Figure 38 Offset Times - Server 2016 Root Dom 2012R2 Client (VIMIC scenario)

Child Domain PDC (Server 2016)

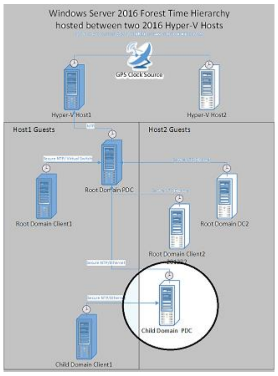


Figure 39 Topology Reference

The child domain PDC is almost an identical position as the root domain replica DC in terms of the stratum and distance from PDC. Its performance matches that of the replica DC.

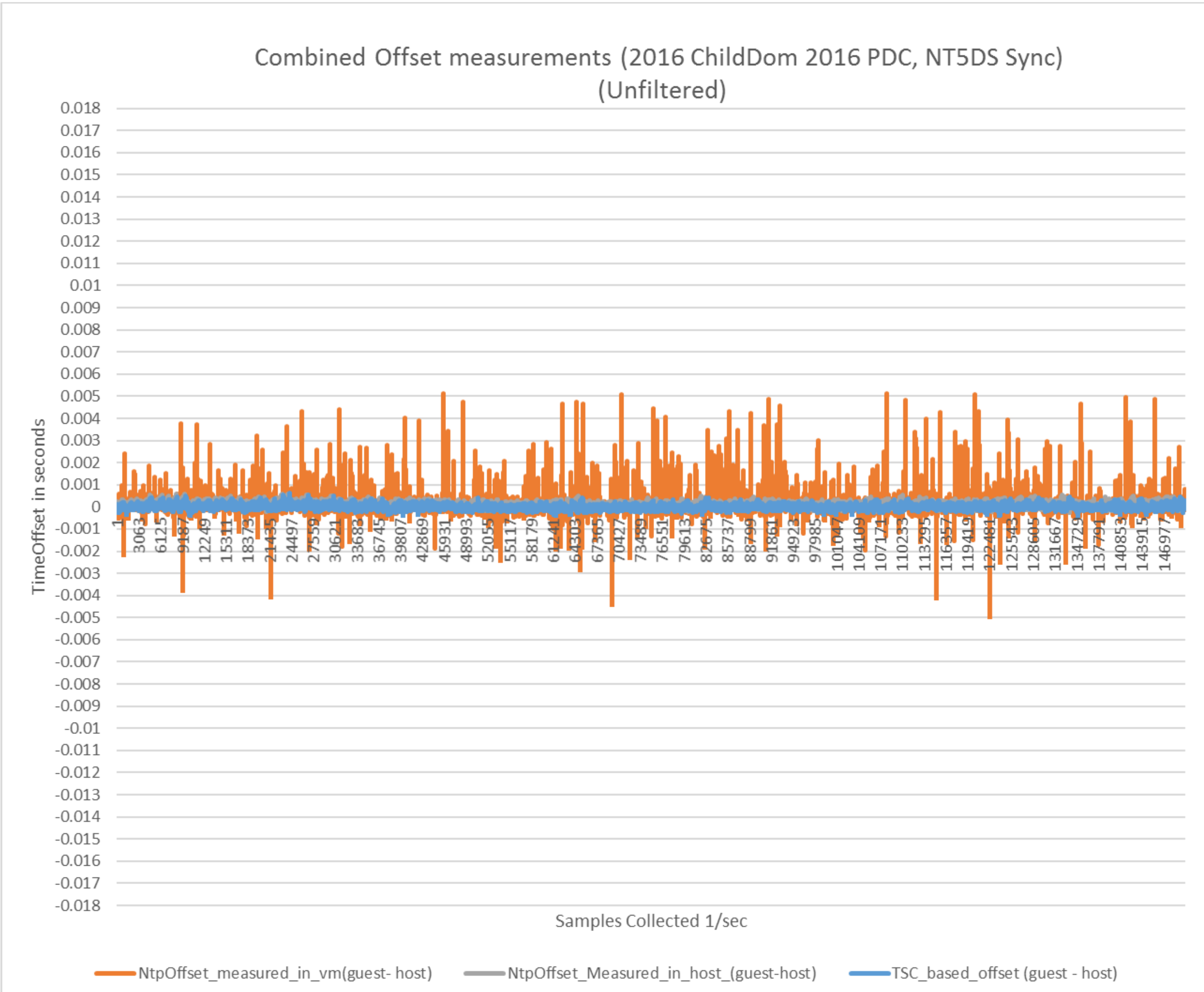


Figure 40 Offset Times - Server 2016 Child Dom 2016 PDC (VIMIC scenario)

The Popcorn Filter has caught 1.5% of the NTP pings from VM to Hyper-V host and 1% of the NTP pings from Hyper-V host to the VM.

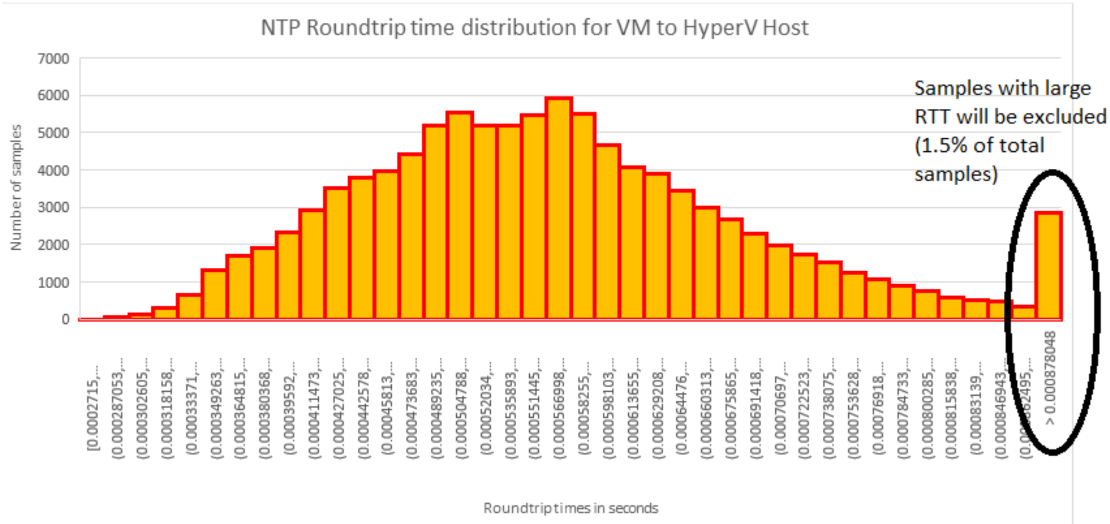


Figure 41 NTP Ping Roundtrip time distribution (VM to Hyper-V Host), Server 2016 Child Dom 2016 PDC (VMIC scenario)

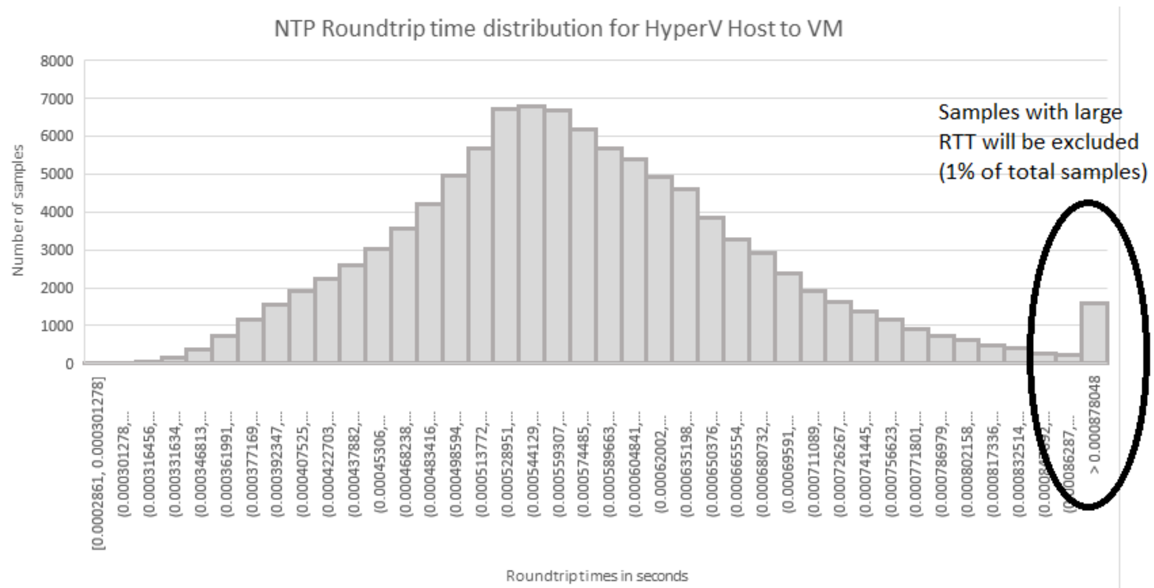


Figure 42 NTP Ping Roundtrip time distribution (Hyper-V Host to VM), Server 2016 Child Dom 2016 PDC (VMIC scenario)

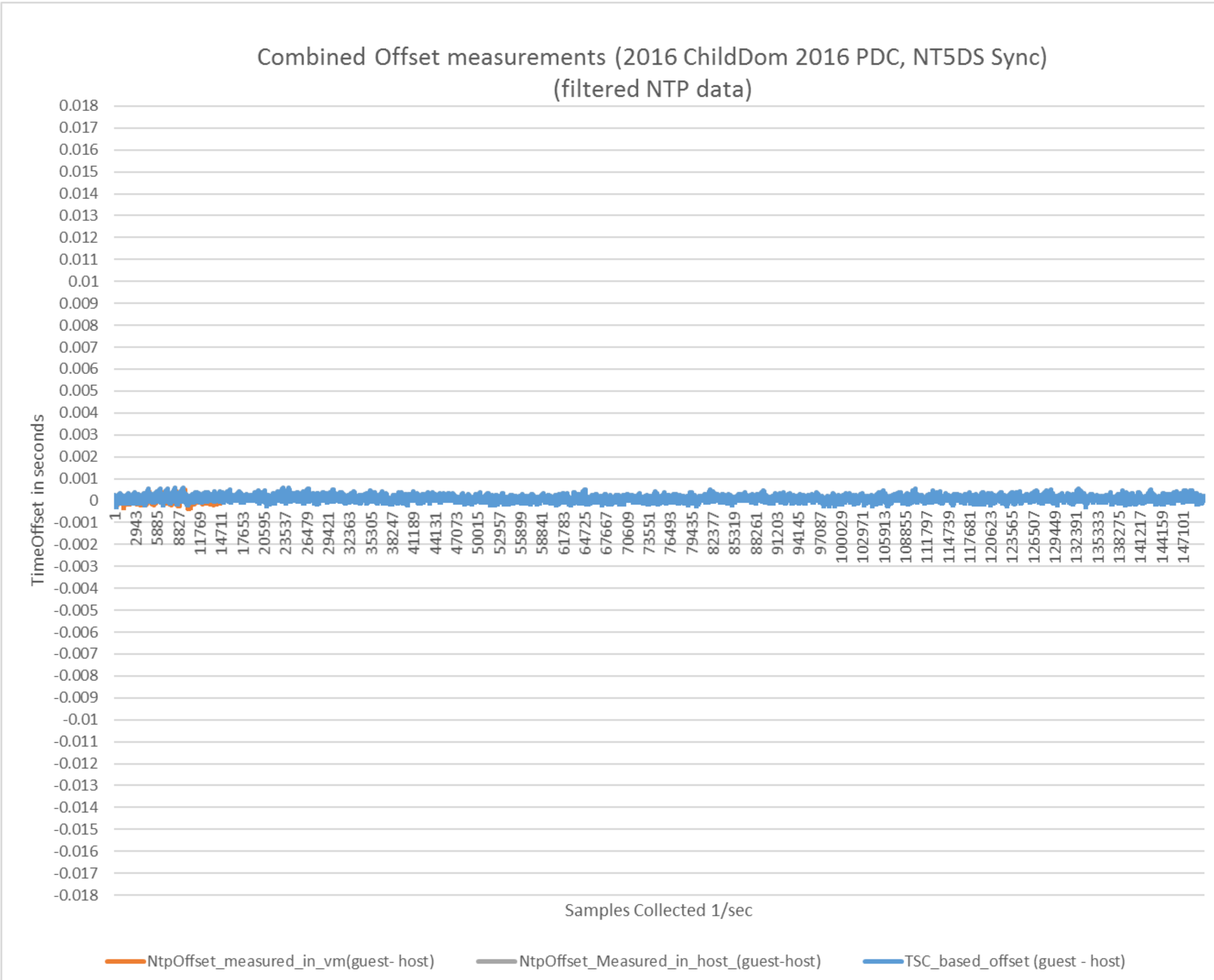


Figure 43 Filtered Offset Times - Server 2016 Child Dom 2016 PDC (VIMIC scenario)

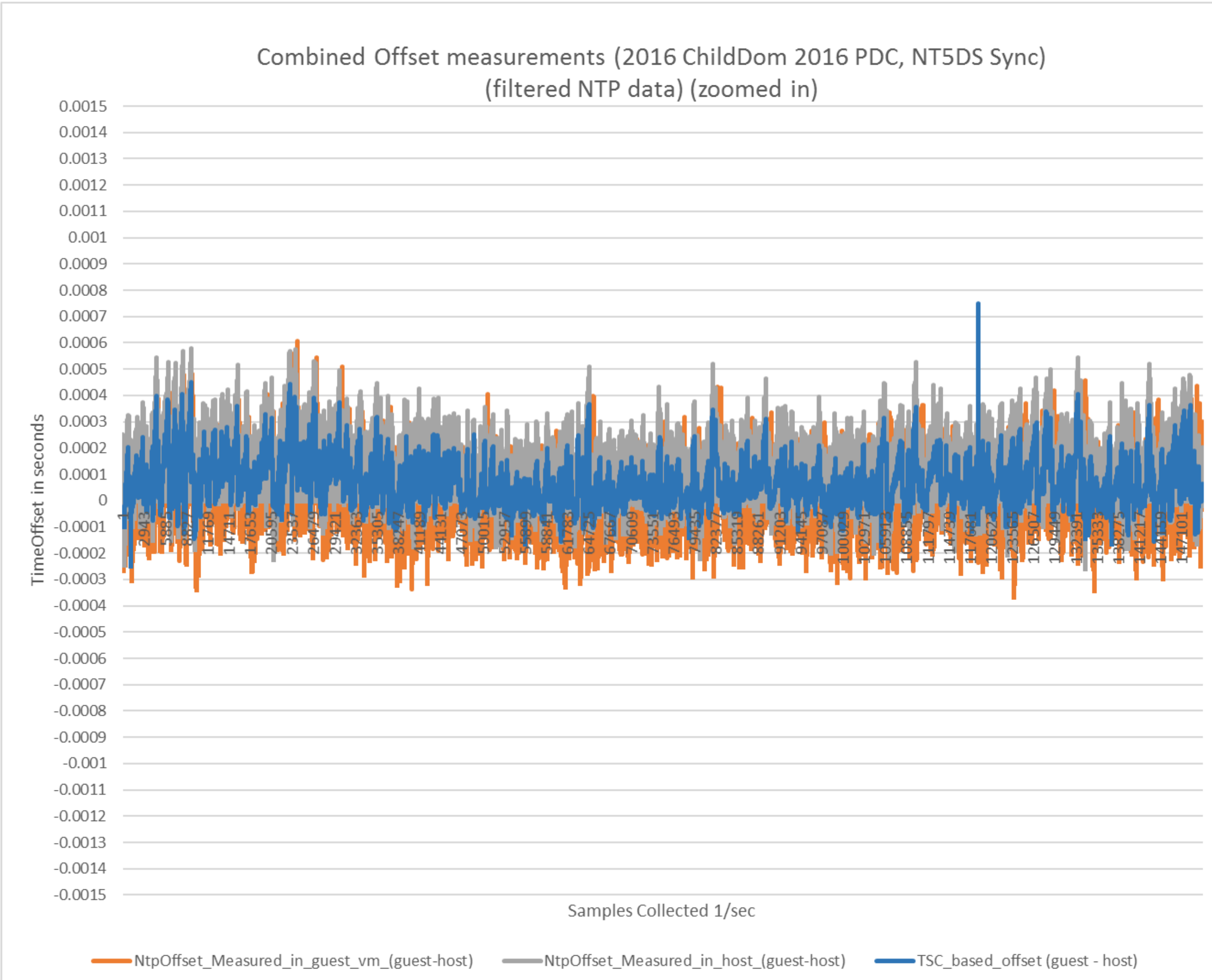


Figure 44 Filtered Offset Times - Server 2016 Child Dom 2016 PDC (VIMIC scenario) (Zoomed in)

Child Domain Client #1 (Server 2016)

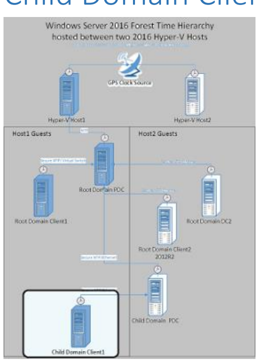


Figure 45 Topology Reference

This is a critical test to demonstrate the scalability of accurate time synchronization in an AD Forest. This client is 2 NTP strata away from the Root PDC and you can see here that it maintains its time within 1ms accuracy.

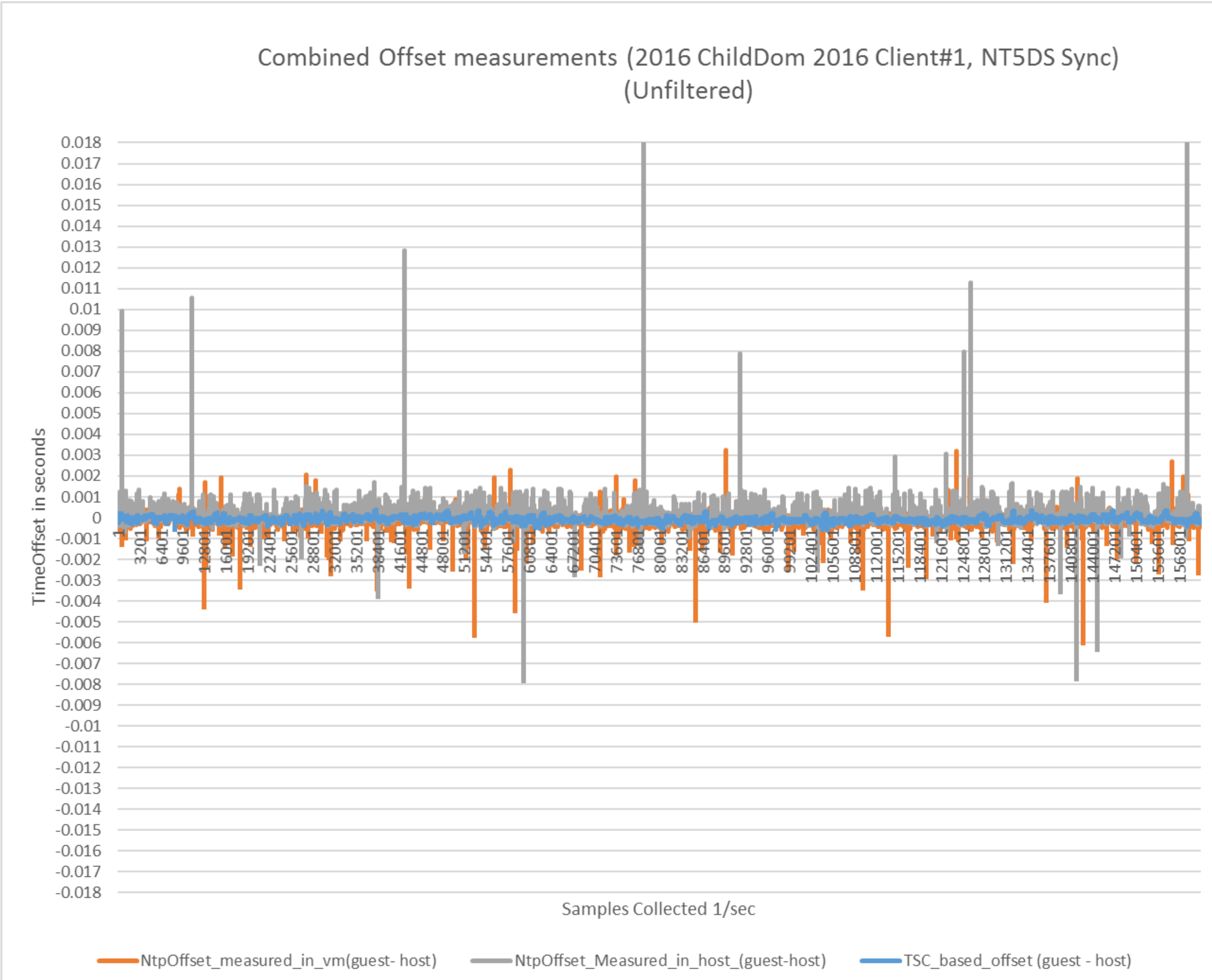


Figure 46 Filtered Offset Times - Server 2016 Child Dom 2016 Client (VIMIC scenario)

The Popcorn filter excluded 0.6% of NTP ping samples from both sets of measurements in this case.

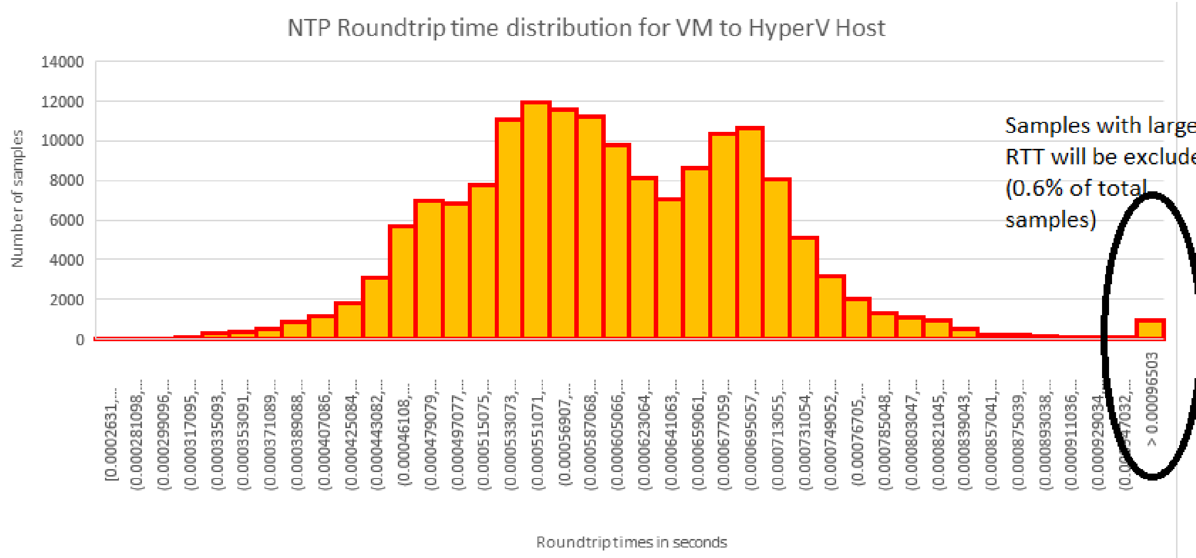


Figure 47 NTP Ping Roundtrip time distribution (VM to Hyper-V Host), Server 2016 Child Dom 2016 Client (VMIC scenario)

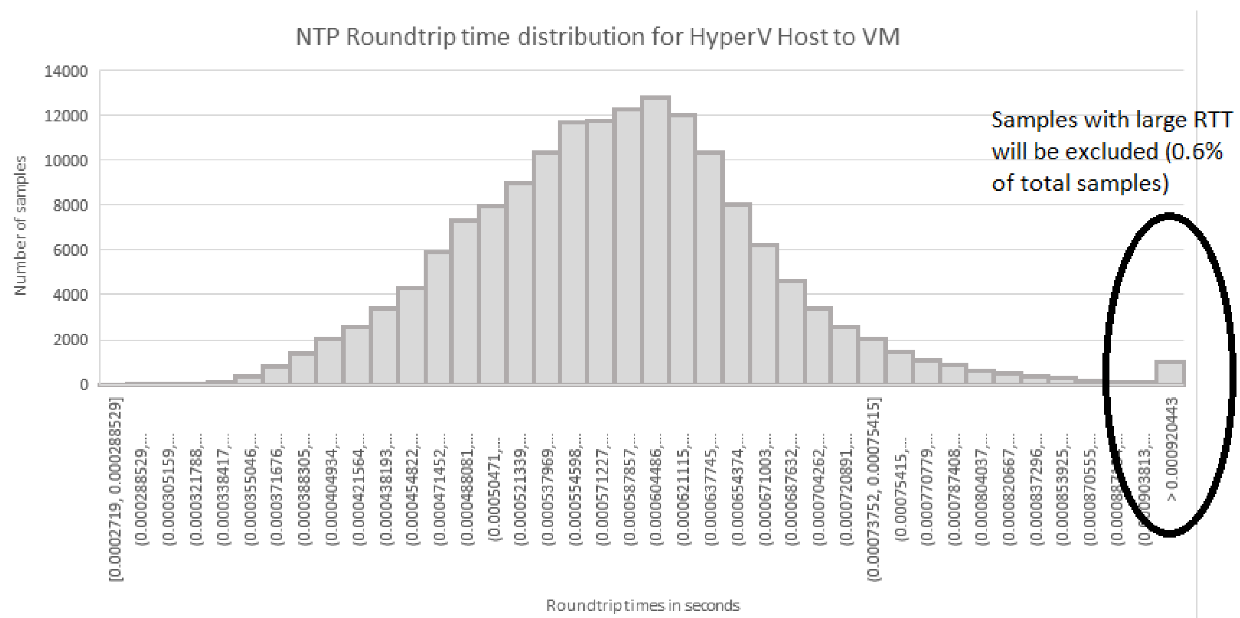


Figure 48 NTP Ping Roundtrip time distribution (Hyper-V Host to VM), Server 2016 Child Dom 2016 Client (VMIC scenario)

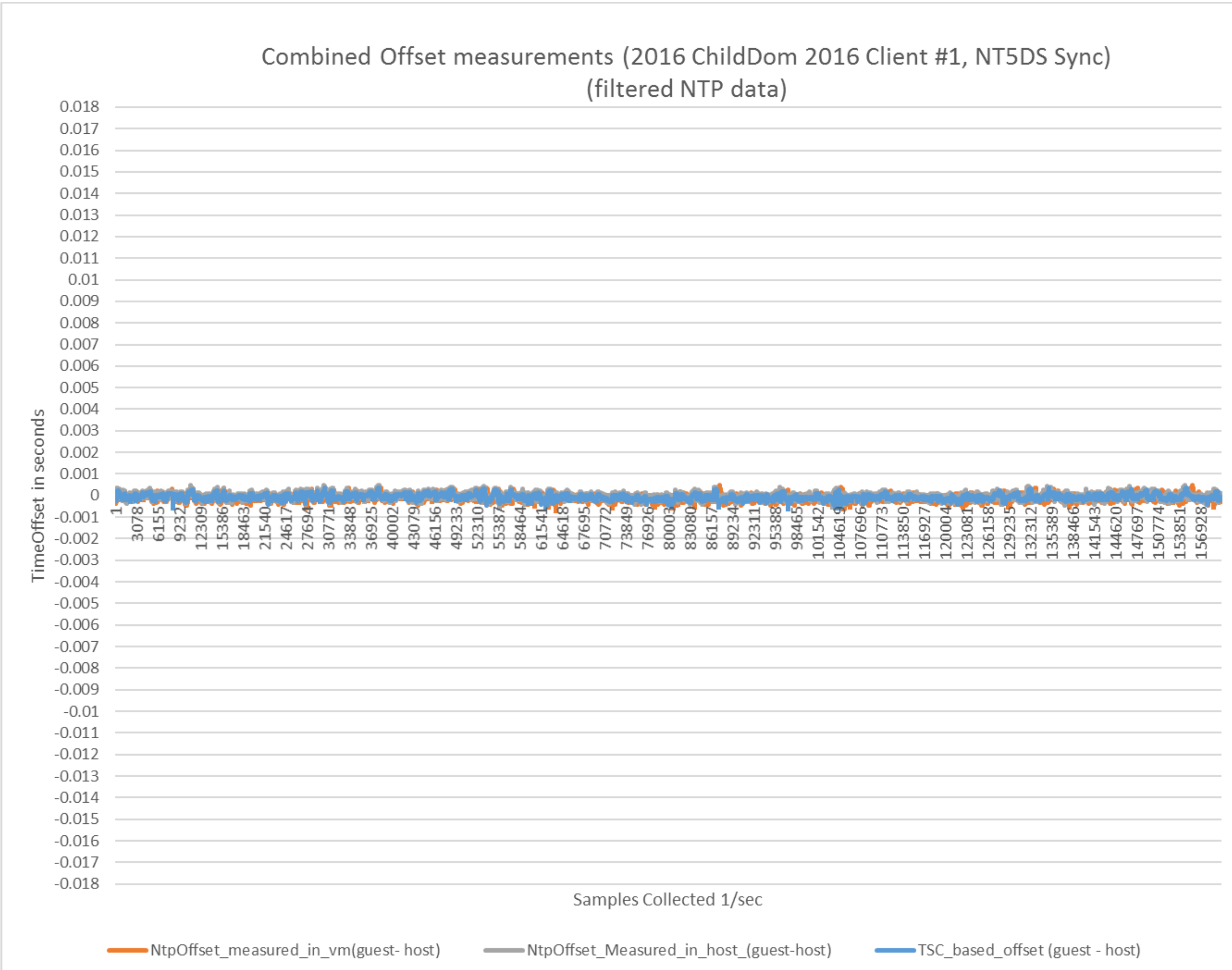


Figure 49 Filtered Offset Times - Server 2016 Child Dom 2016 Client (VIMIC scenario)

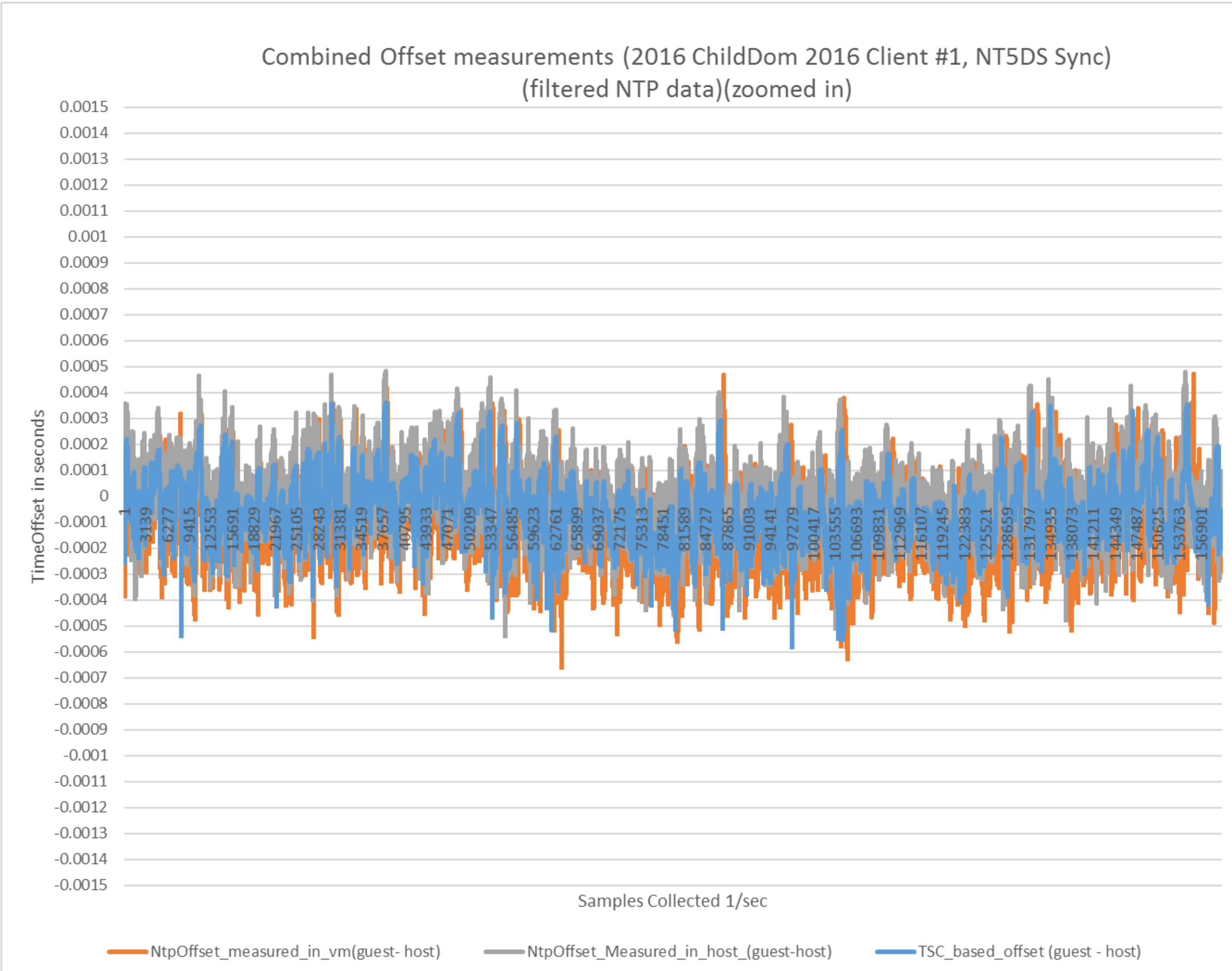


Figure 50 Filtered Offset Times - Server 2016 Child Dom 2016 Client (VIMIC scenario) (Zoomed in)

Server 2016 Forest using NTP Root time source (Scenario #2)

In this scenario, the Server 2016 Root PDC of the forest is configured manually to syncing its time using NTP from its Hyper-V host machine itself. This is essentially a manual configuration in W32time to use a simple (unsecured) NTP server, polling at an interval of 64 seconds. The host partition's clock is synchronized to a very high accuracy using Sync-n-Scale hardware.

Root PDC (Server 2016)

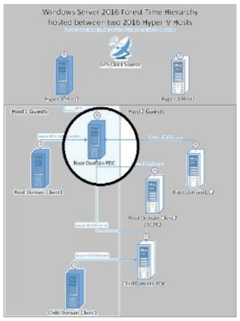


Figure 51 Topology Reference

This chart shows the 3 different sets of time offsets like in the previous scenario for the 2016 Root Domain PDC. The TSC offset shows the clock holding time within 1ms offset. The NTP data indicates larger offsets at times and needs to be put through the Popcorn filter.

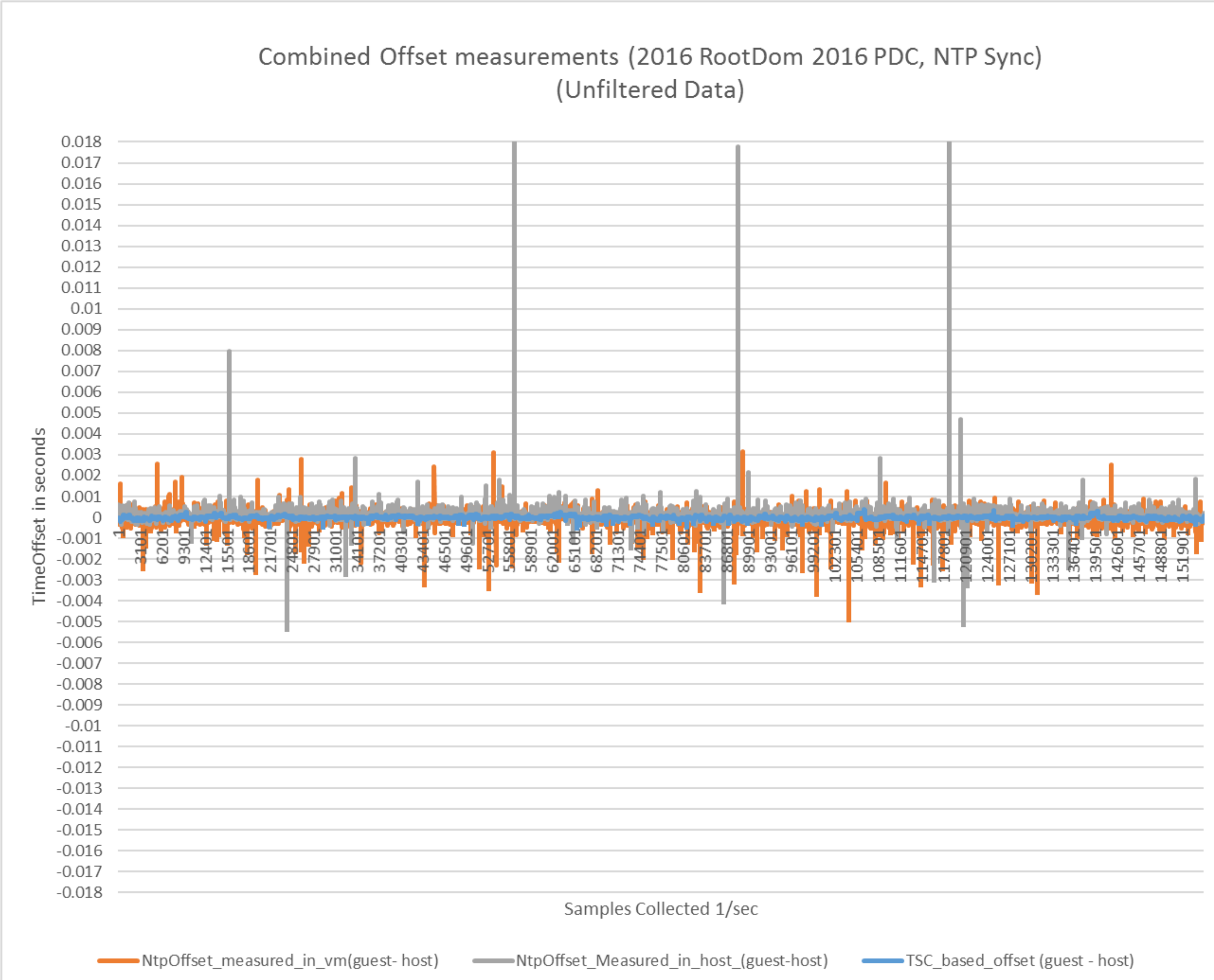


Figure 52 Offset Measurements - Server 2016 Root Dom PDC (NTP scenario)

Note: The performance observed here is a characteristic of W32time in Server 2016 when using a simple NTP server that is in close network proximity. Please refer to the "NTP Quarter Mile" test case for details on time sync performance when using a simple NTP server that is a few network hops away.

The Popcorn filter has caught 1.1% of the samples from both sets of NTP data.

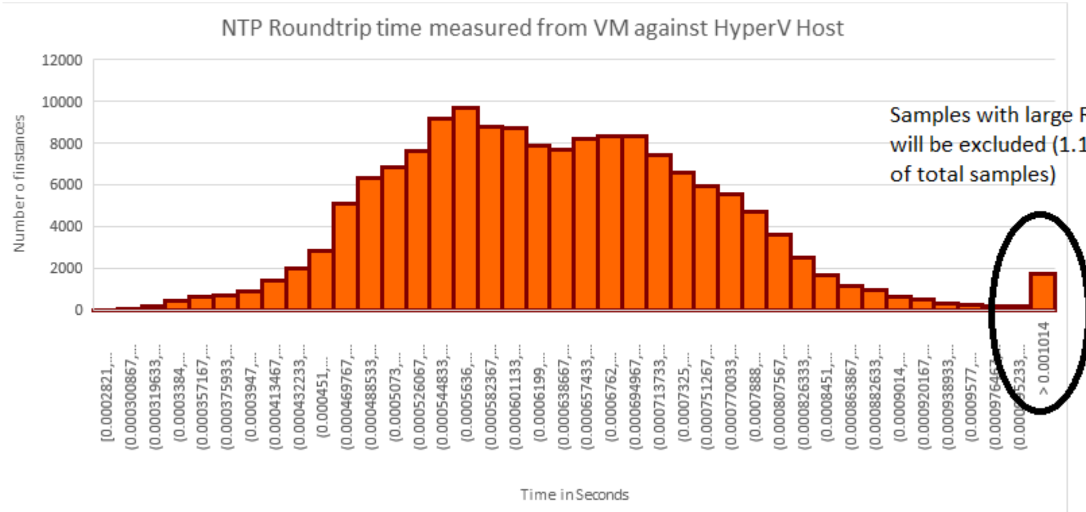
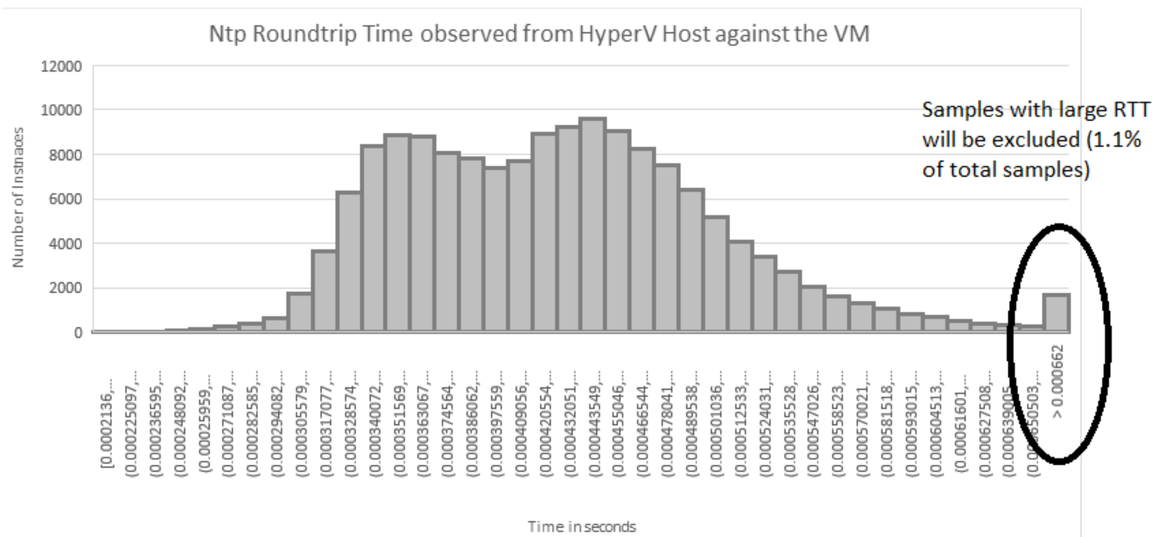


Figure 53 NTP Ping Roundtrip time distribution (VM to Hyper-V Host), Server 2016 Root Dom PDC (NTP scenario)



This data shows that once we exclude the potentially noisy NTP data, the rest of it agrees with the TSC data. All of the data stays well within 1ms, demonstrating the sub-ms time sync accuracy.

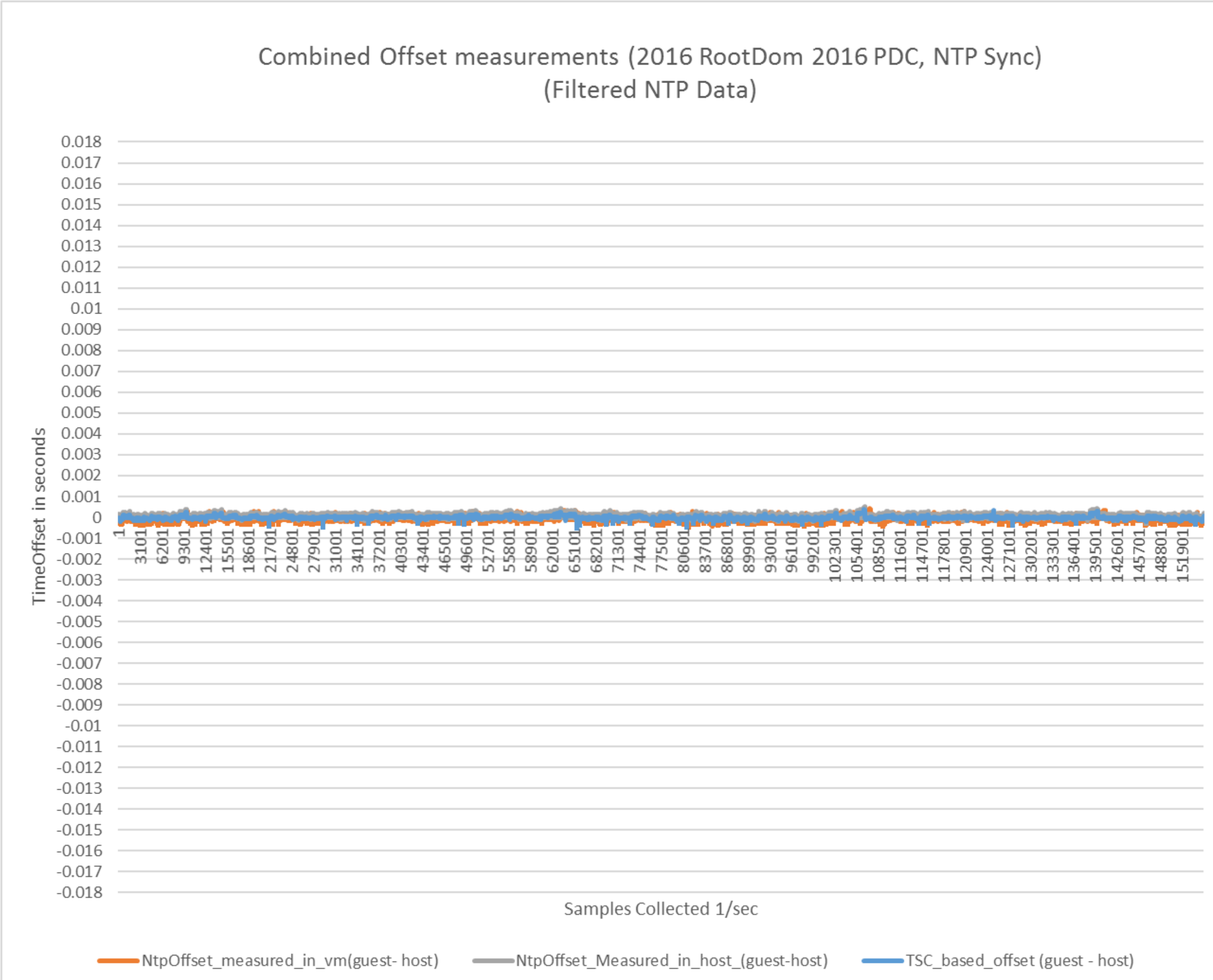


Figure 54 Filtered Offset Times - Server 2016 Root Dom PDC (NTP scenario)

The chart below zooms into the above data to show the details. The NTP data measured from the Hyper-V Host is a little less noisy than the data measured other way around. Most importantly, the three streams of data are in agreement.

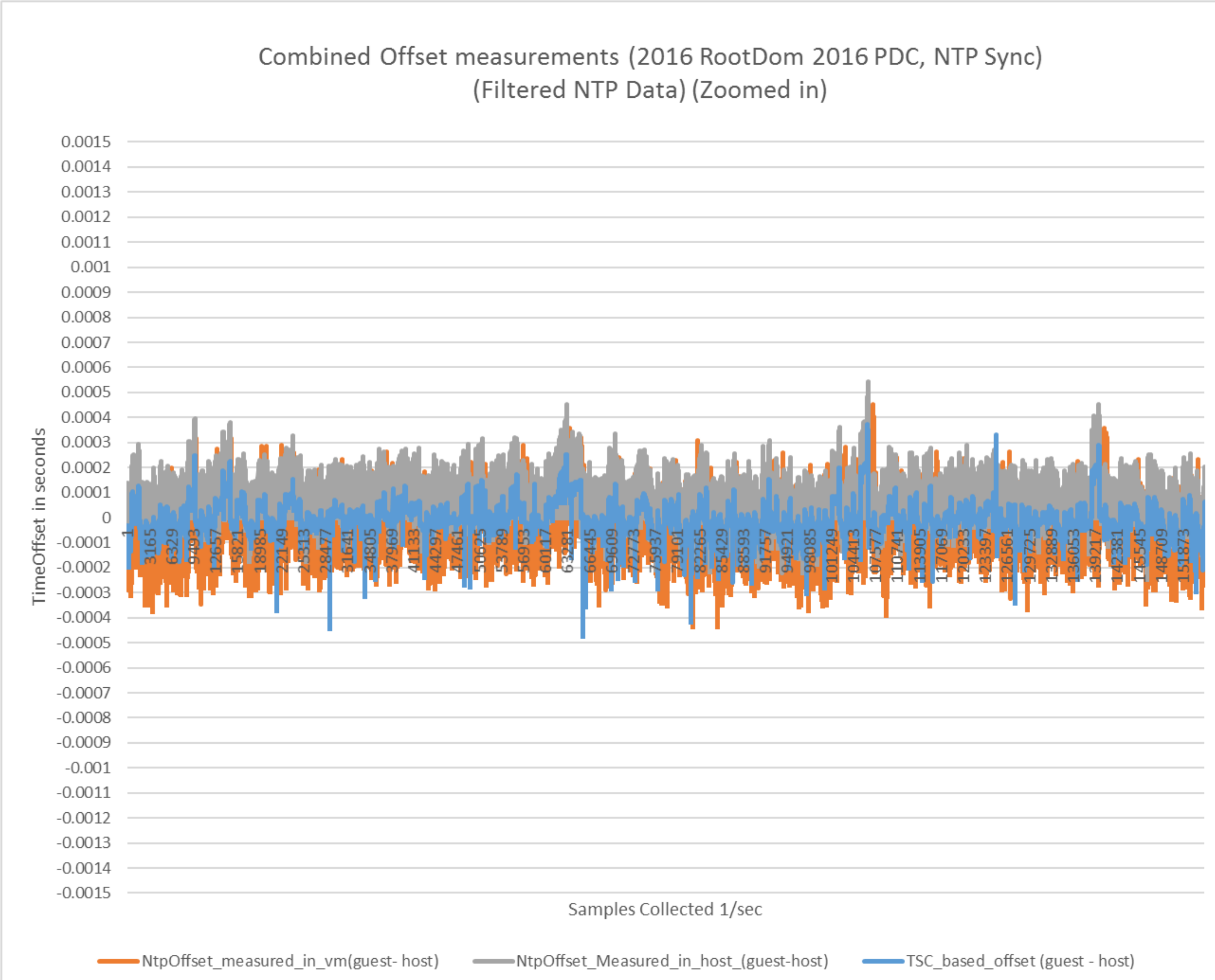


Figure 55 Filtered Offset Times - Server 2016 Root Dom PDC (NTP scenario) (Zoomed in)

Root Domain Replica DC (Server 2016)

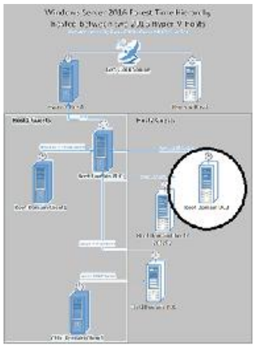


Figure 56 Topology Reference

TSC data here shows the time on the replica DC in the root domain is being maintained within 1ms. The NTP data needs to be put through the Popcorn filter for better clarity.

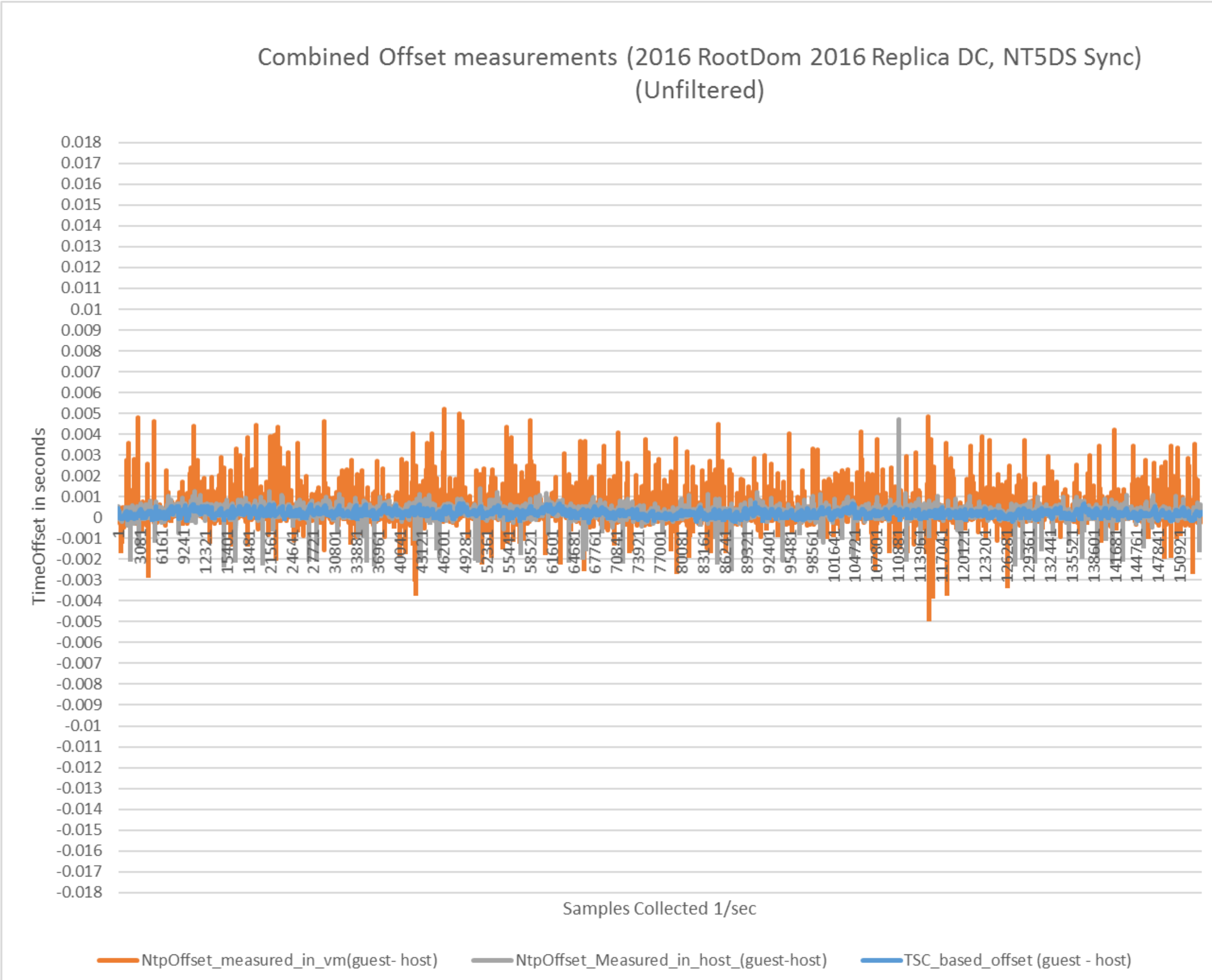


Figure 57 Offset Measurements - Server 2016 Root Dom Replica DC (NTP scenario)

The filter excluded 4% of the NTP data measured in the VM and 1% of the NTP data measured in the host.

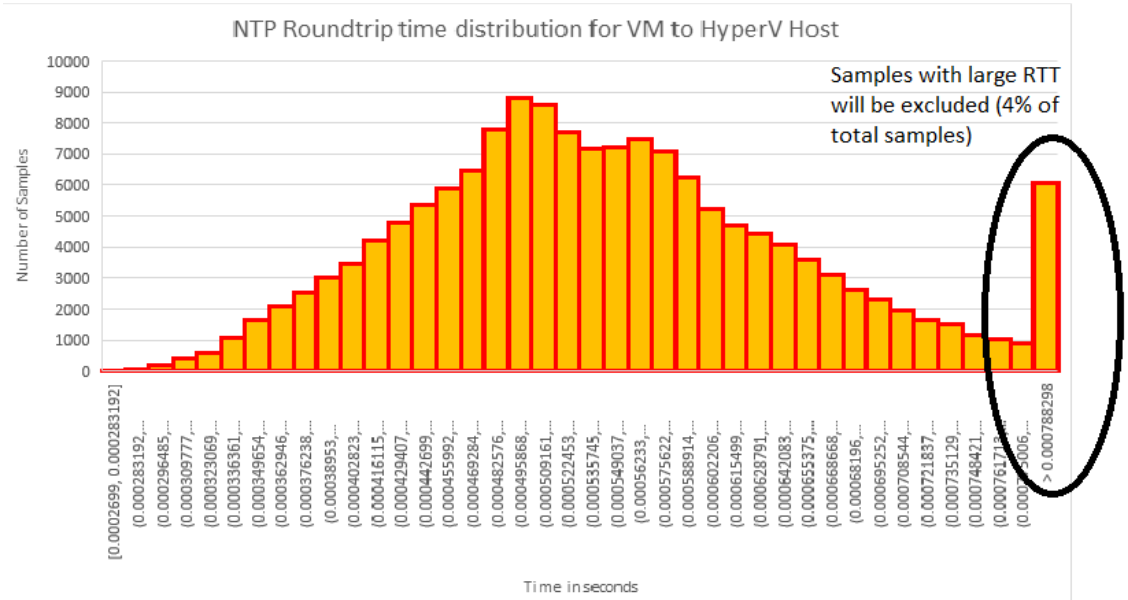


Figure 58 NTP Ping Roundtrip time distribution (VM to Hyper-V Host), Server 2016 Root Dom Replica DC (NTP scenario)

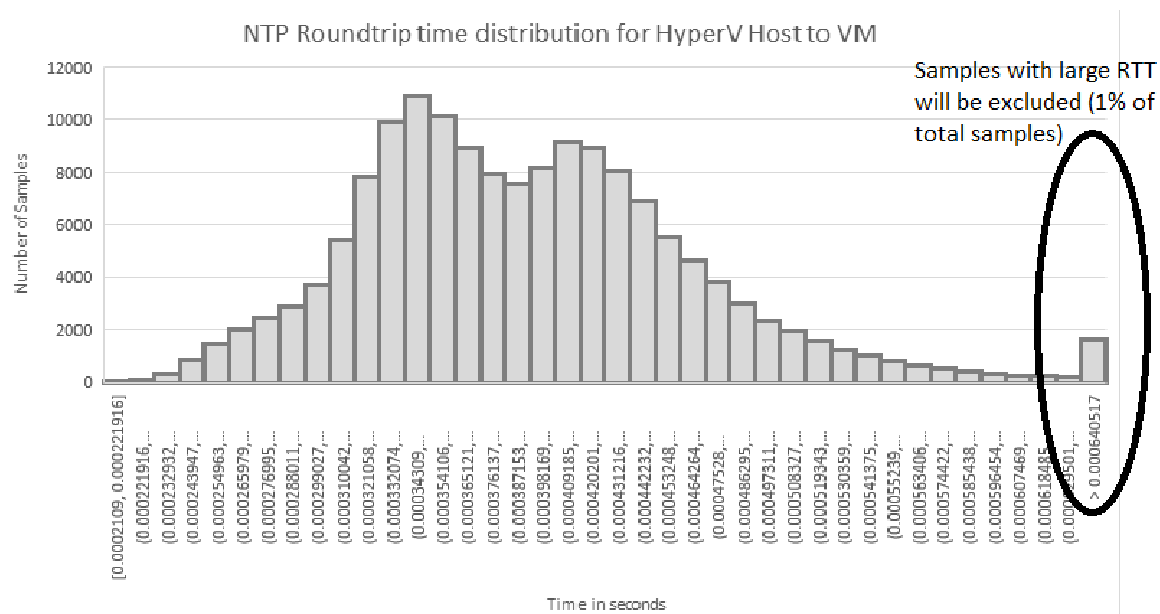


Figure 59 NTP Ping Roundtrip time distribution (Hyper-V Host to VM), Server 2016 Root Dom Replica DC (NTP scenario)

The filtered data shows a bit more noise than the Root PDC case, but that is expected and the time is maintained within 1ms error.

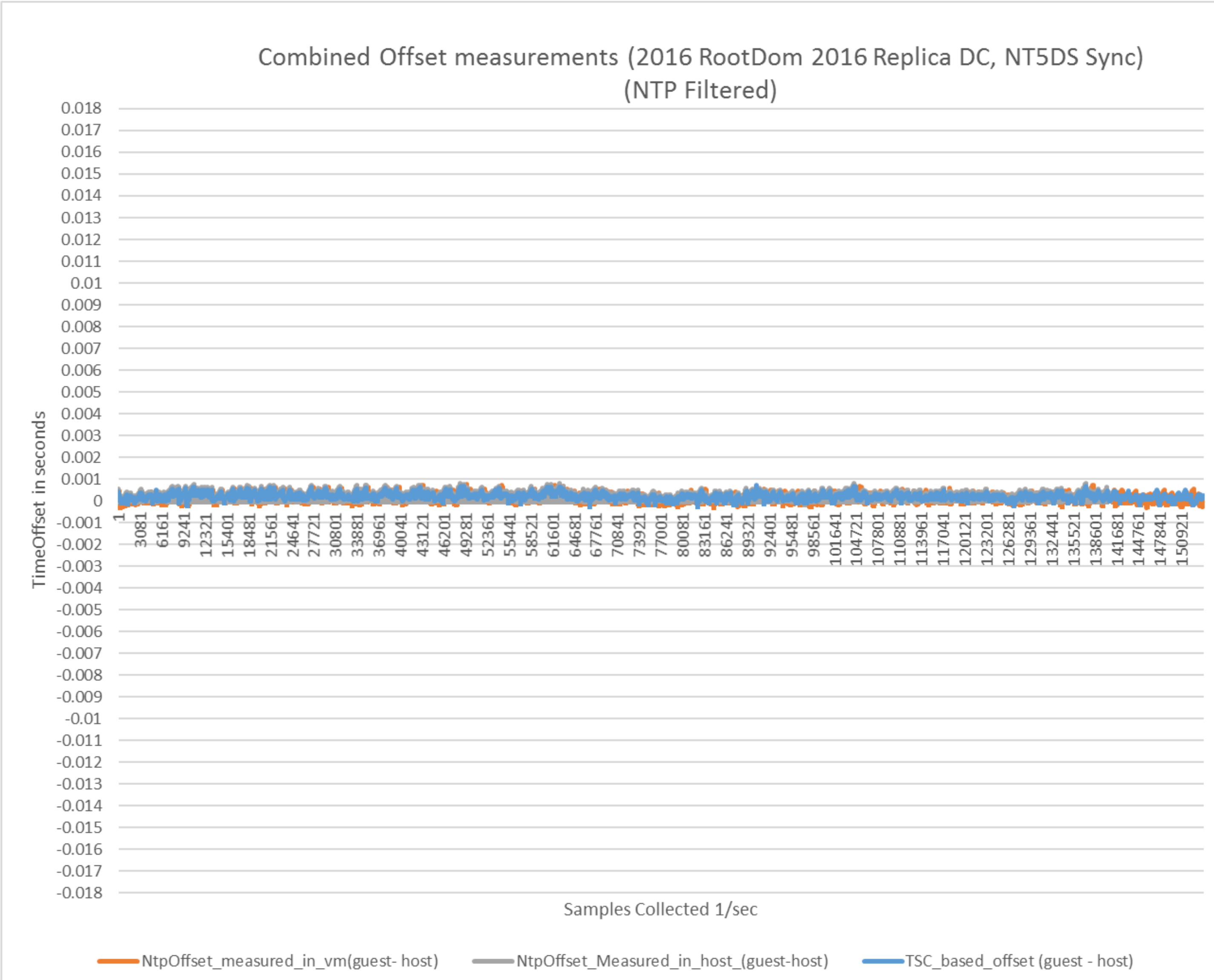


Figure 60 Filtered Offset Times - Server 2016 Root Dom Replica DC (NTP scenario)

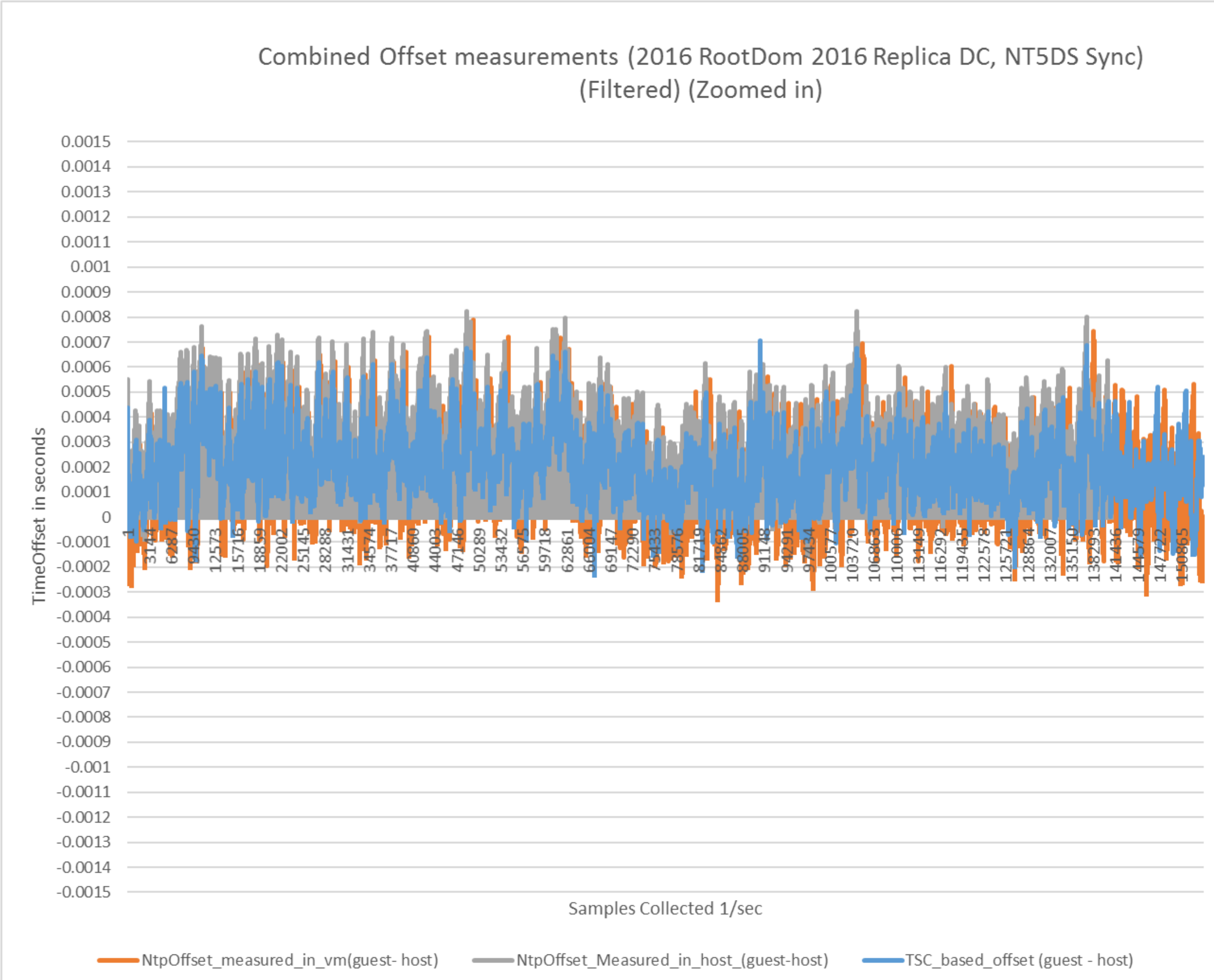


Figure 61 Filtered Offset Times - Server 2016 Root Dom Replica DC (NTP scenario) (Zoomed in)

Root Domain Client #1 (Server 2016)

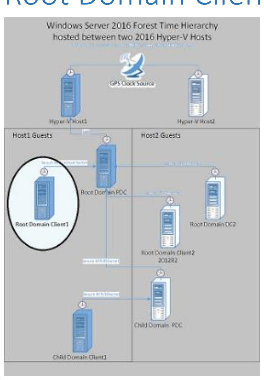


Figure 62 Topology Reference

This machine is one stratum away from the root PDC similar to the replica DC in the root domain. The main difference between the two machines is that this machine is not going over Ethernet to obtain time from its time server. However, the time sync performance on both machines are very much comparable to each other.

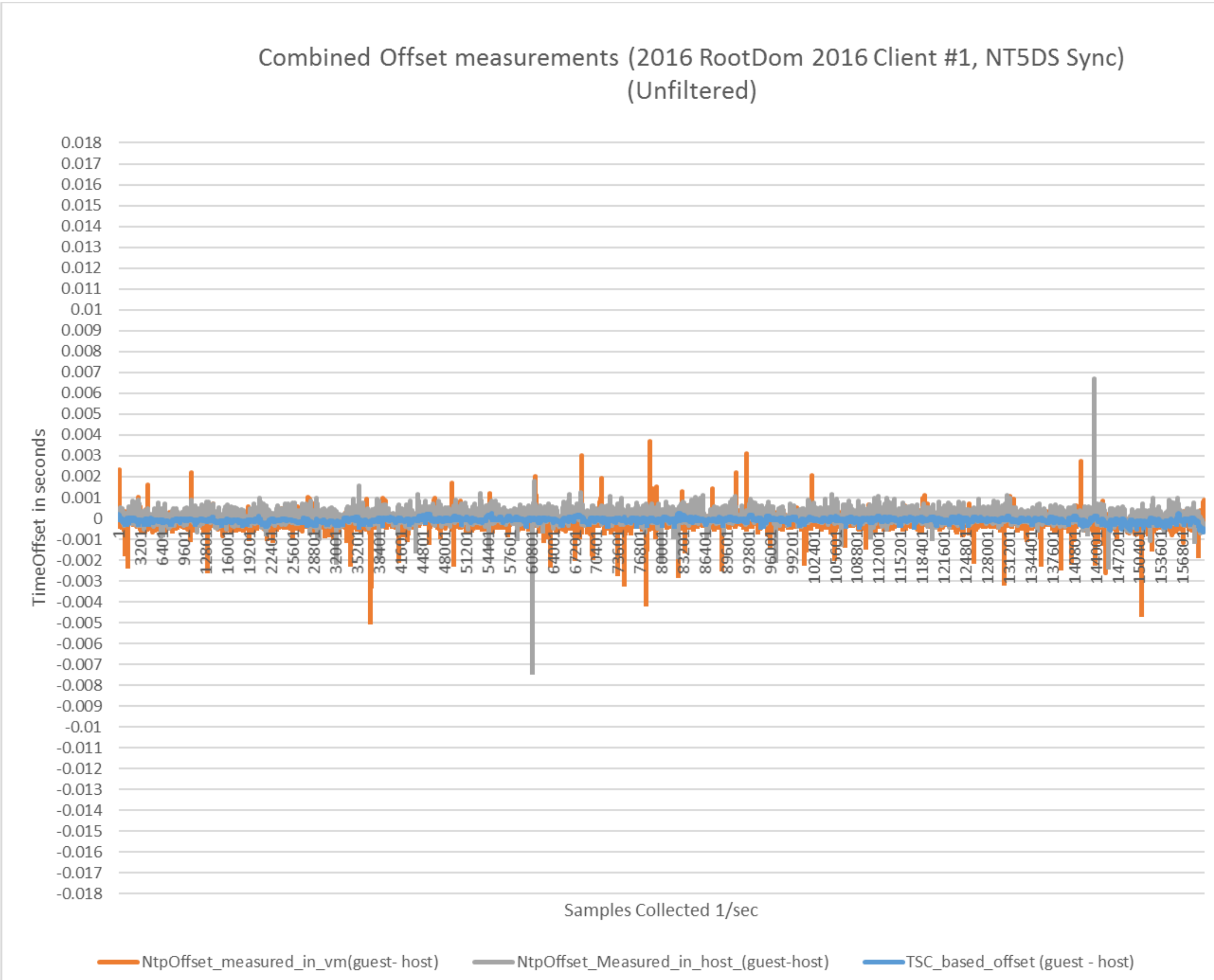


Figure 63 Offset Times - Server 2016 Root Dom 2016 Client (NTP scenario)

The Popcorn Filter excludes ~1.5% of the NTP data measured from within the VM and ~1% of the NTP data measured from the Hyper-V host.

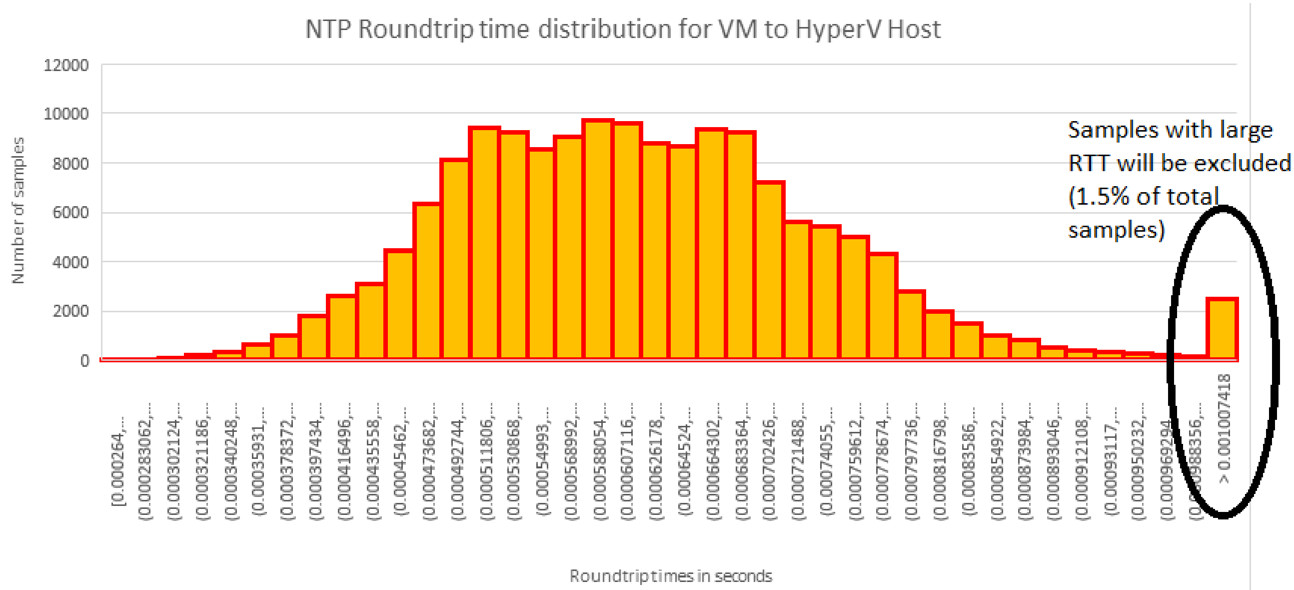


Figure 64 NTP Ping Roundtrip time distribution (VM to Hyper-V Host), Server 2016 Root Dom 2016 Client (NTP scenario)

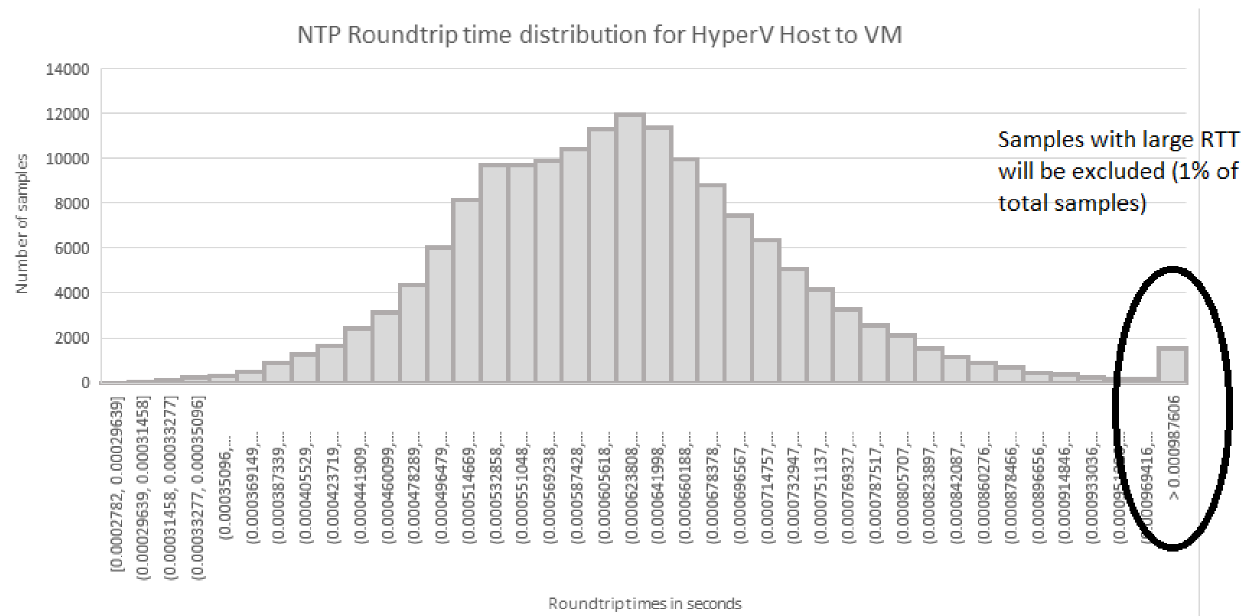


Figure 65 NTP Ping Roundtrip time distribution (Hyper-V Host to VM), Server 2016 Root Dom 2016 Client (NTP scenario)

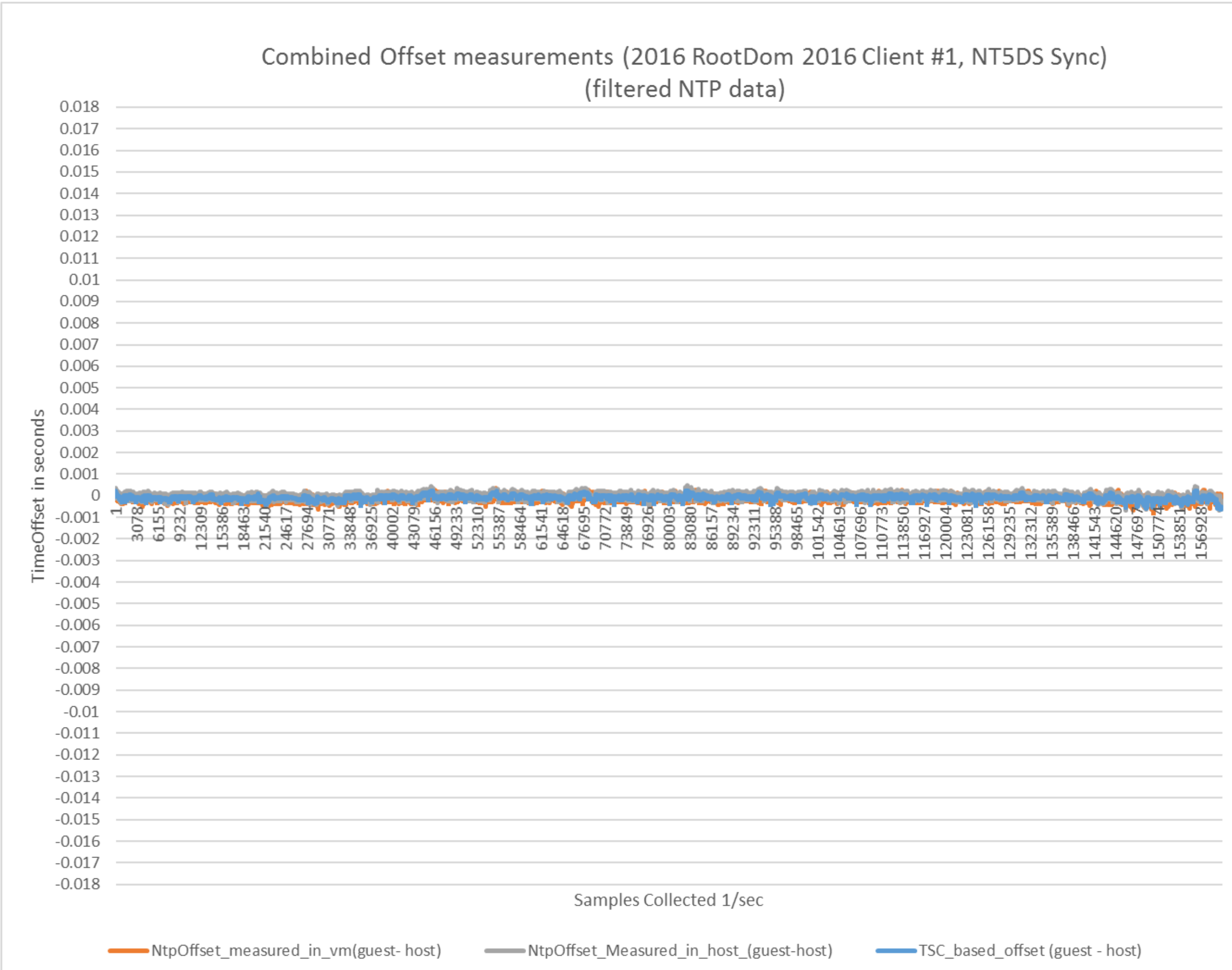


Figure 66 Filtered Offset Times - Server 2016 Root Dom 2016 Client (NTP scenario)

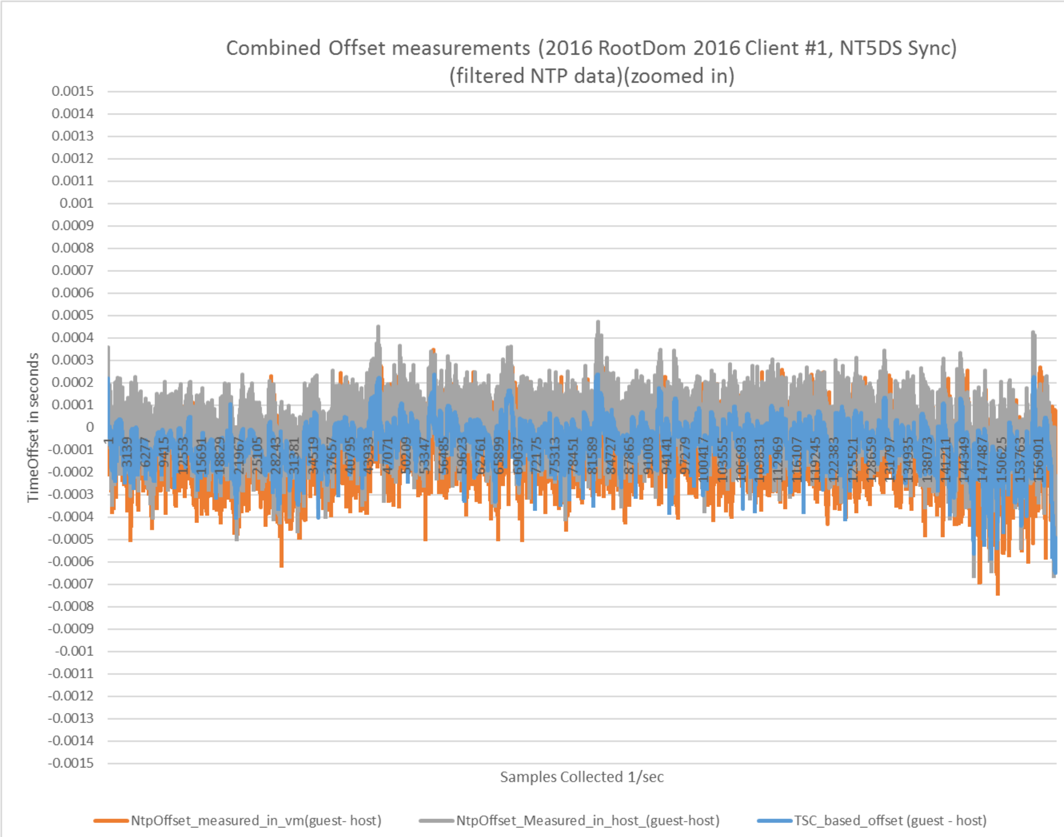


Figure 67 Filtered Offset Times - Server 2016 Root Dom 2016 Client (NTP scenario) (Zoomed in)

Root Domain Client #2 (Server 2012R2)

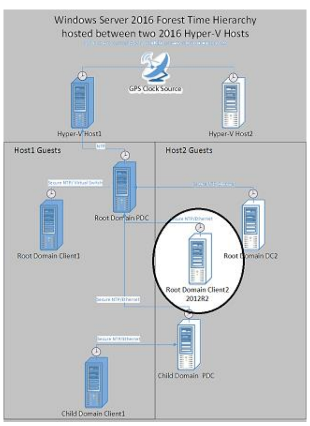


Figure 68 Topology Reference

This is another data set that shows different NTP ping measurements in Server2012 R2, depending on where the measurement is made from. This server is similar to the Root PDC in scenario #2 and is keeping a very steady time compared to a typical Server 2012R2 machine. However, just like other Server2012 R2 machines, this the time returned by this NTP Server is off of the system time by about 10ms. No filtering is necessary to present this data.

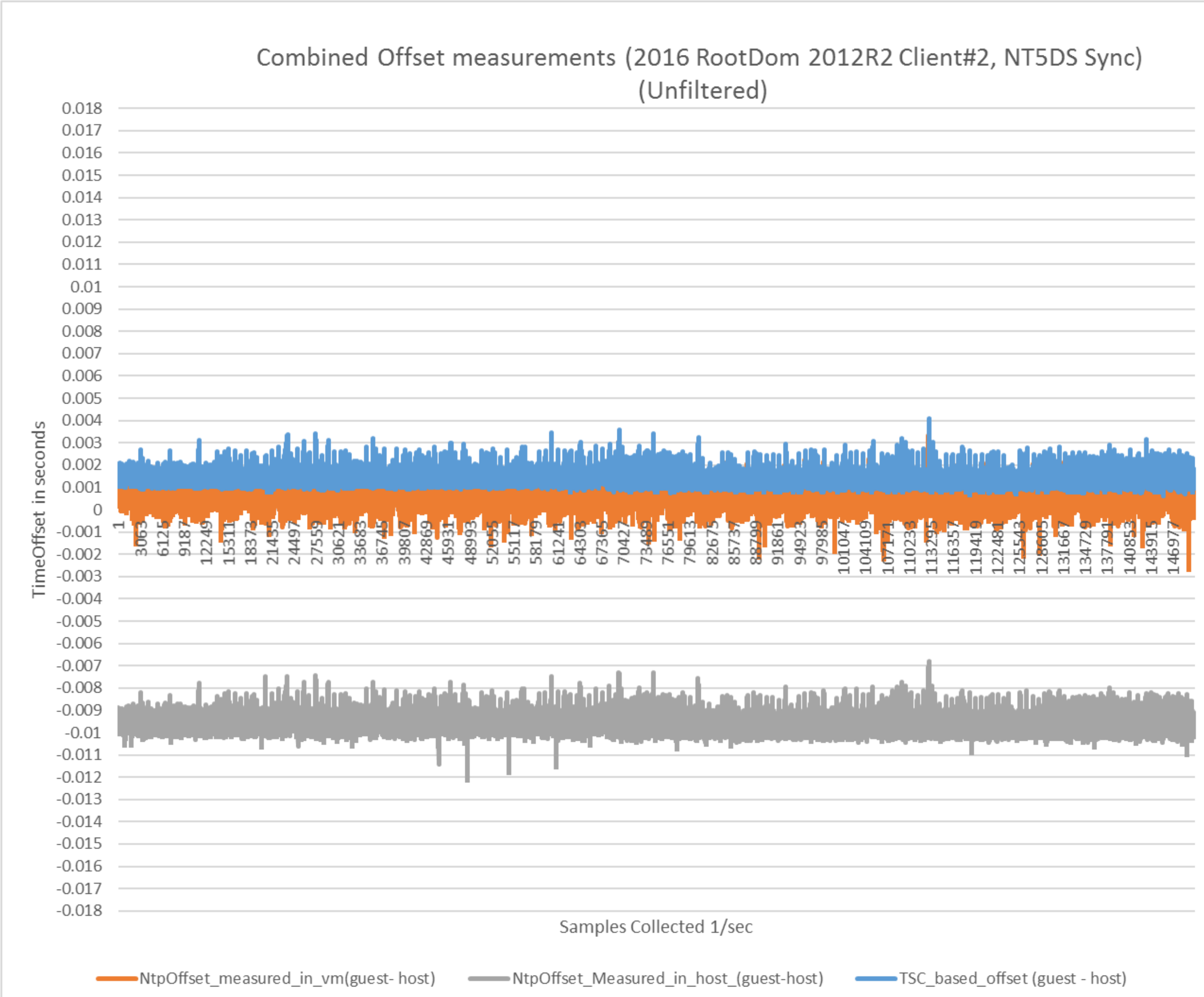


Figure 69 Offset Times - Server 2016 Root Dom 2012R2 Client (NTP scenario)

Child Domain PDC (Server 2016)

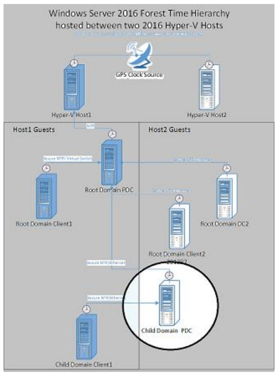


Figure 70 Topology Reference

This is in a near identical position as the replica DC in this topology. The time still stays within 1ms error.

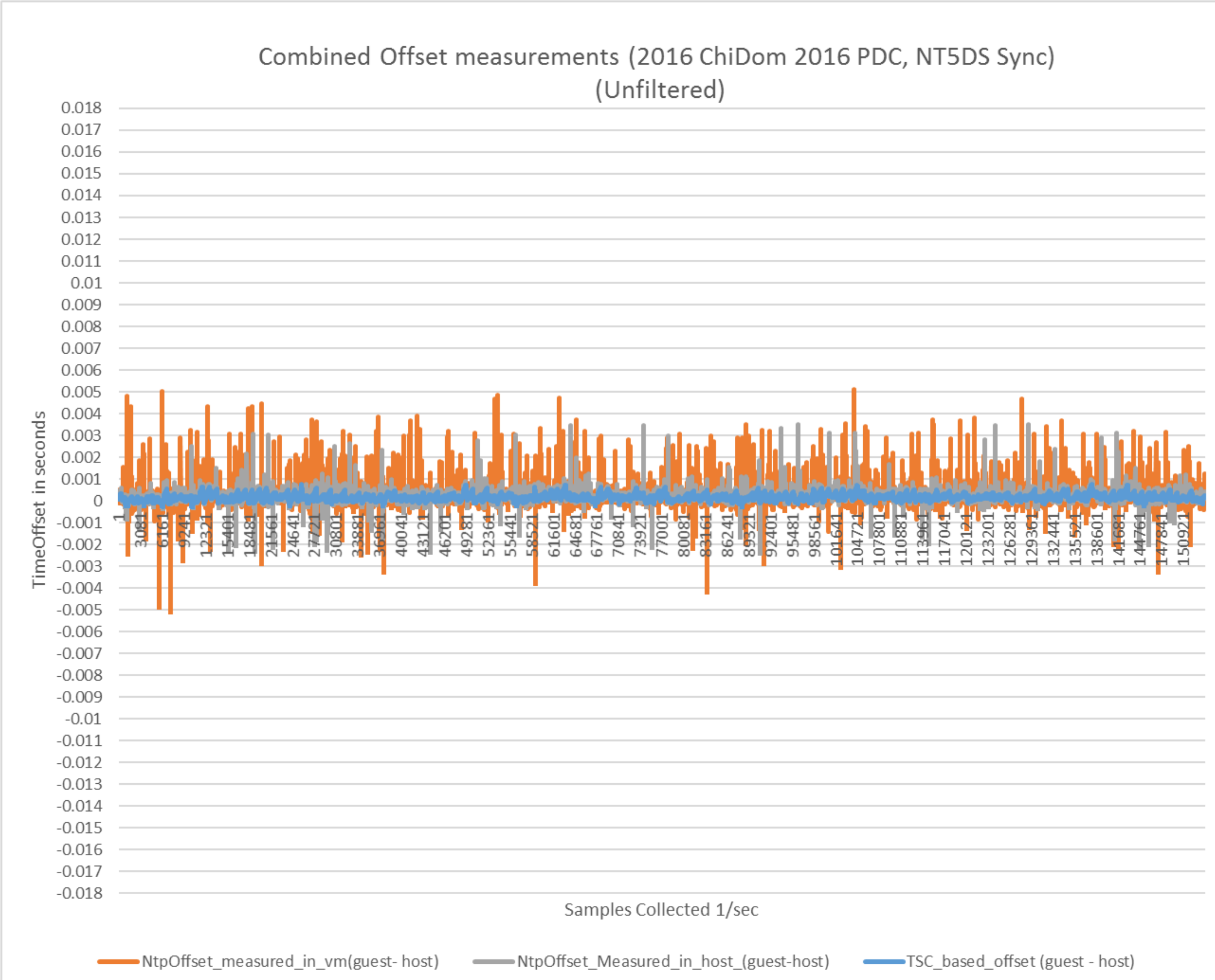


Figure 71 Offset Times - Server 2016 Child Dom 2016 PDC (NTP scenario)

The Popcorn filter catches 2.6% of VM NTP samples and 0.3% of host NTP samples in this case.

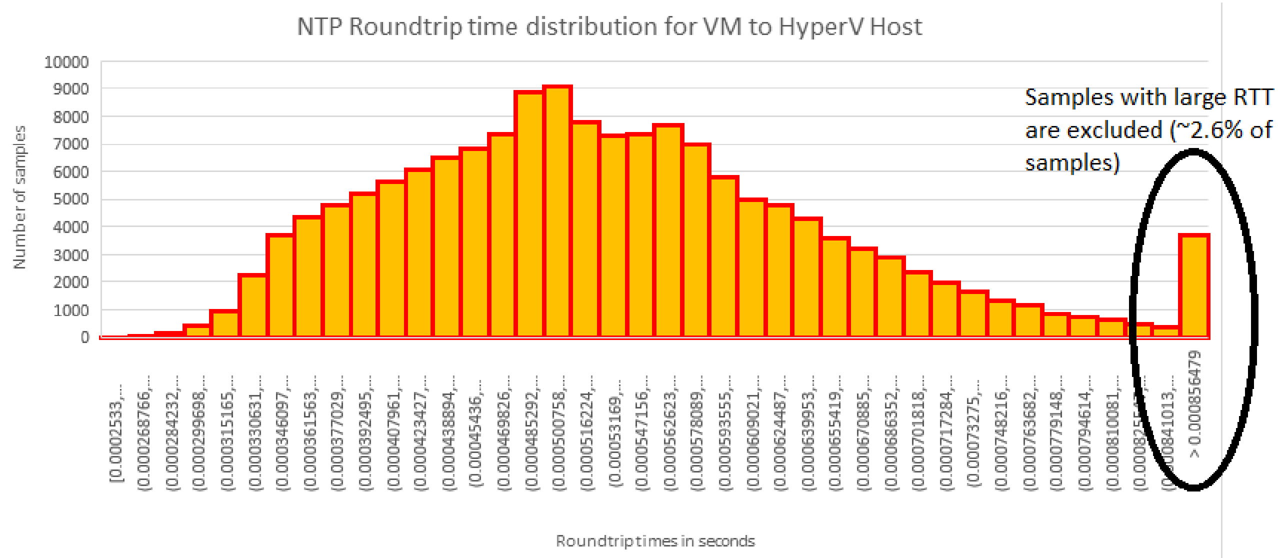


Figure 72 NTP Ping Roundtrip time distribution (VM to Hyper-V Host), Server 2016 Child Dom 2016 PDC (NTP scenario)

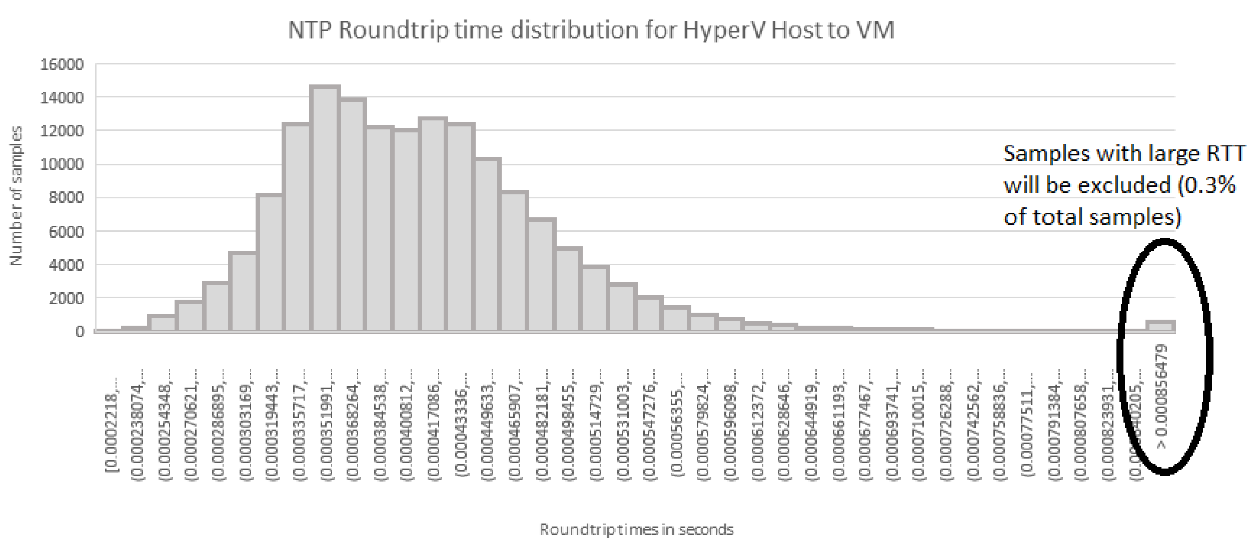


Figure 73 NTP Ping Roundtrip time distribution (Hyper-V Host to VM), Server 2016 Child Dom 2016 PDC (NTP scenario)

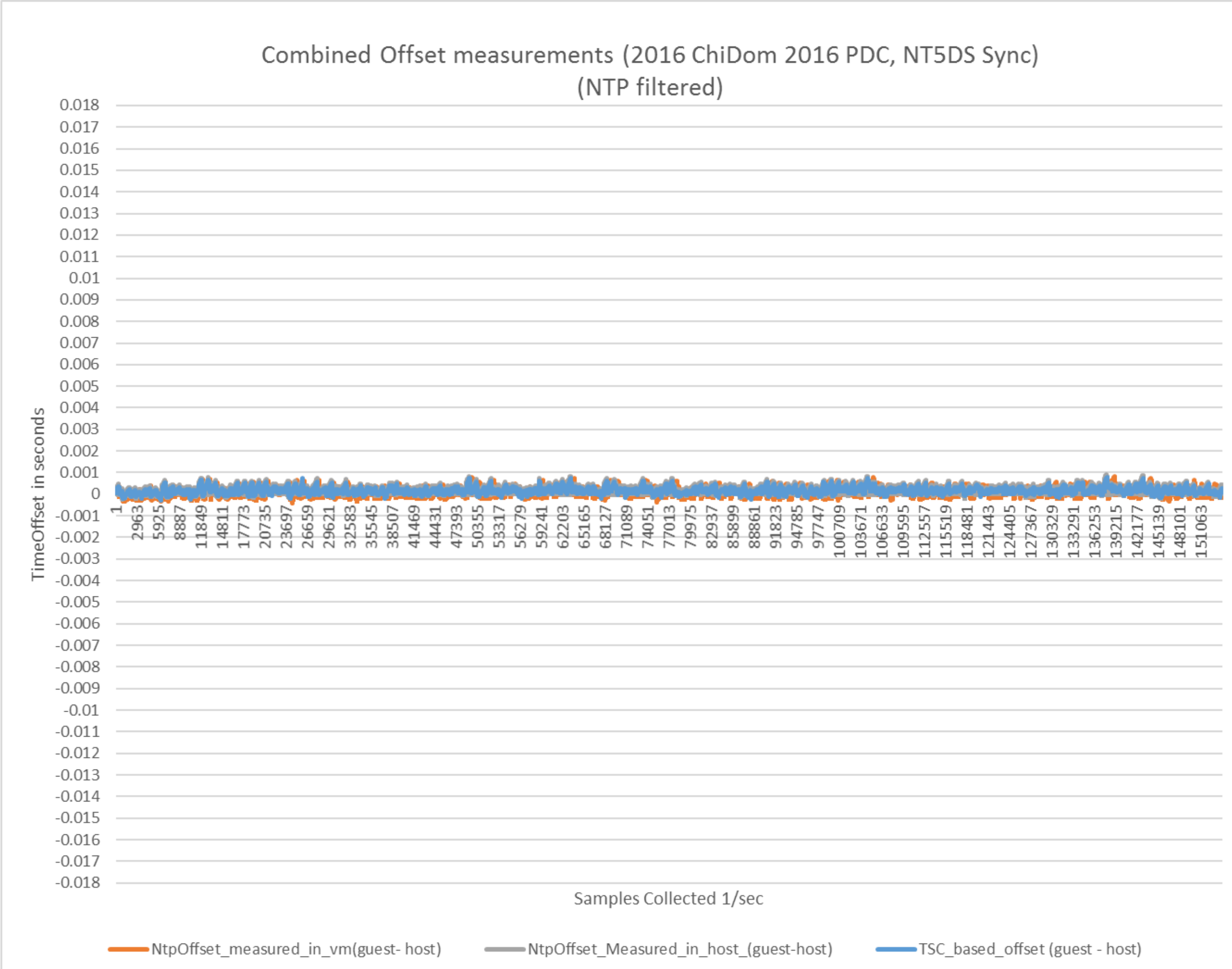


Figure 74 Filtered offset Times - Server 2016 Child Dom 2016 PDC (NTP scenario)

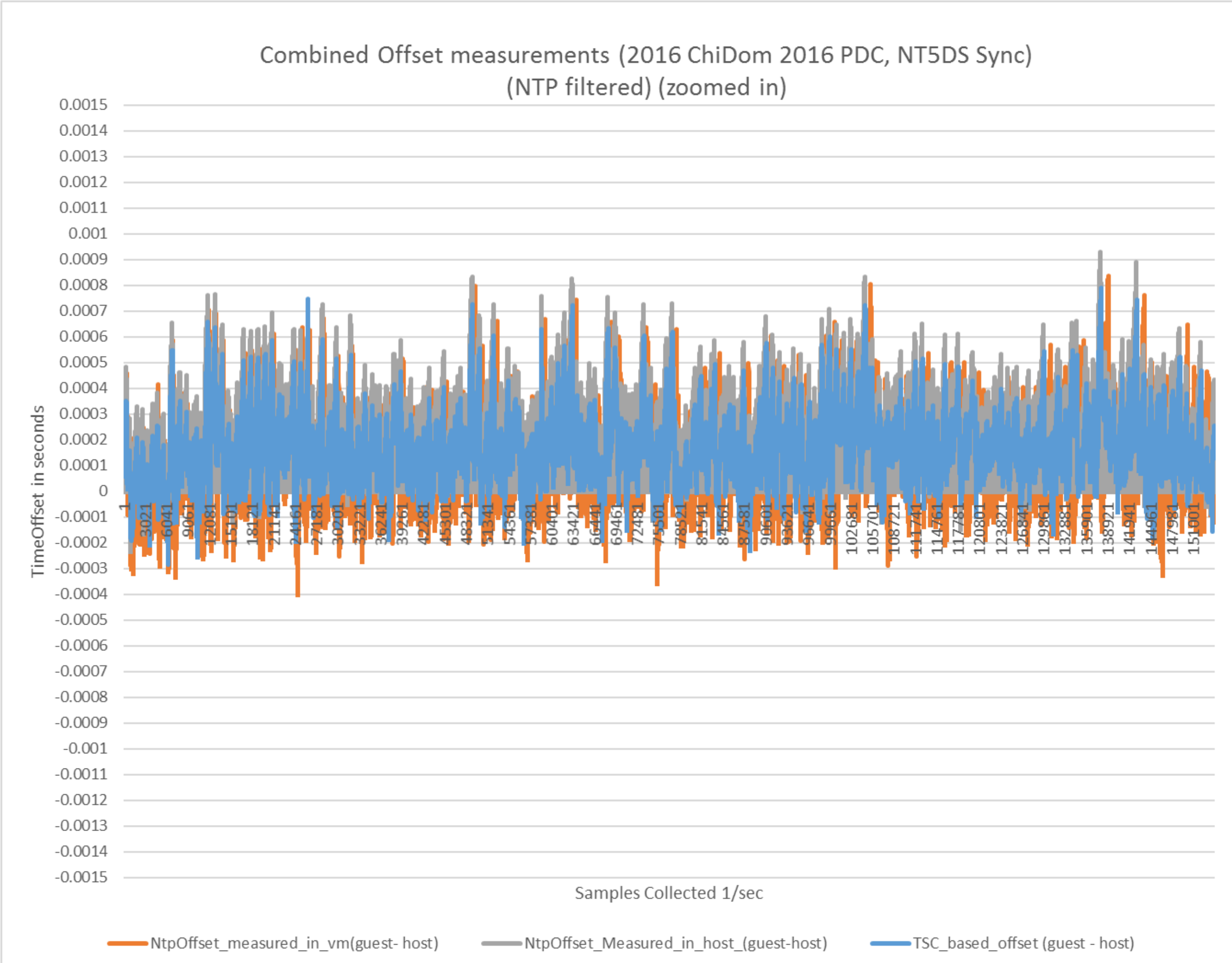


Figure 75 Filtered offset Times - Server 2016 Child Dom 2016 PDC (NTP scenario) (Zoomed in)

Child Domain Client #1 (Server 2016)

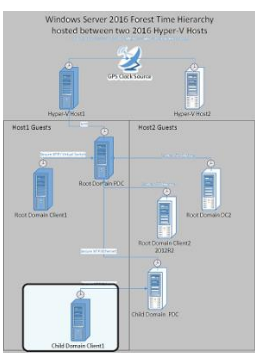


Figure 76 Topology Reference

This machine is 2 strata away from Root PDC and 3 from the accurate NTP source. It is critical because it demonstrates the scalability of the high accuracy service. The accuracy is very likely to cross 1ms boundary in another 1 or 2 hops after this.

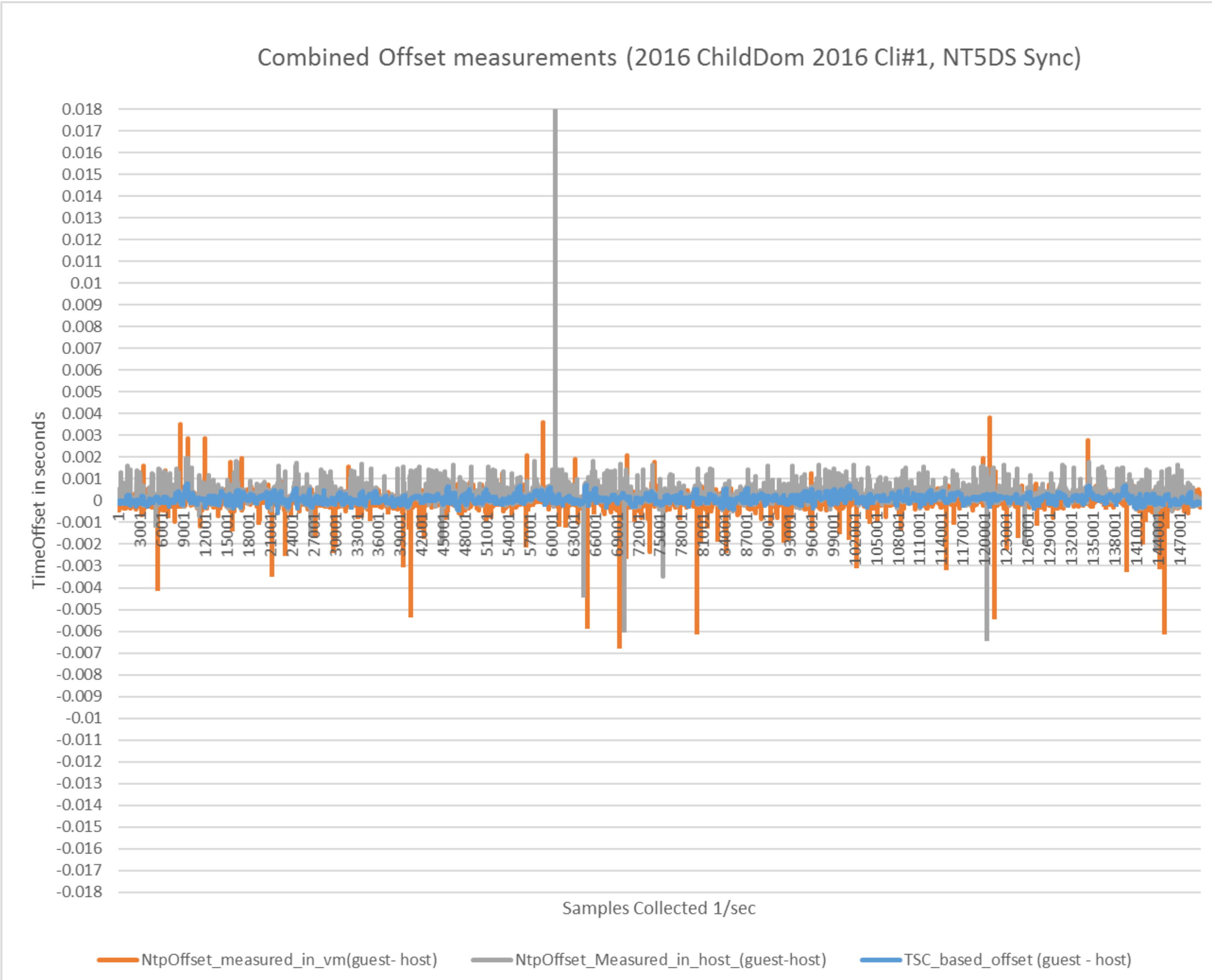


Figure 77 Offset Times - Server 2016 Child Dom 2016 Client (NTP scenario)

The popcorn filter has caught 3.1% of the NTP samples in the VM and 0.5% of the samples taken in the Host.

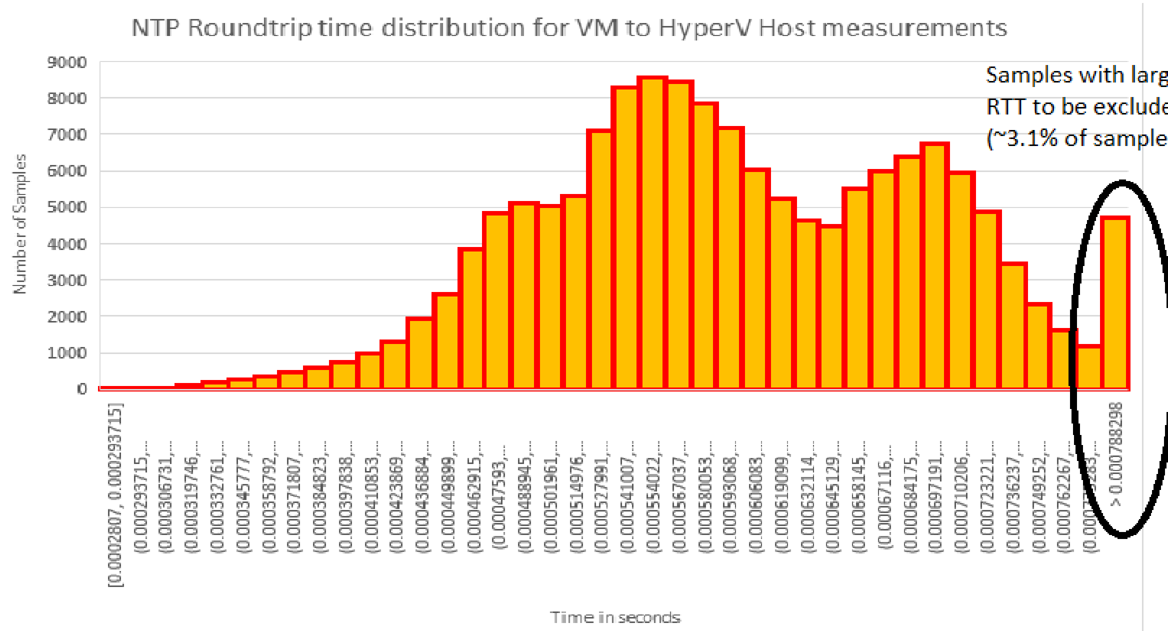


Figure 78 NTP Ping Roundtrip time distribution (VM to Hyper-V Host), Server 2016 Child Dom 2016 Client (NTP scenario)

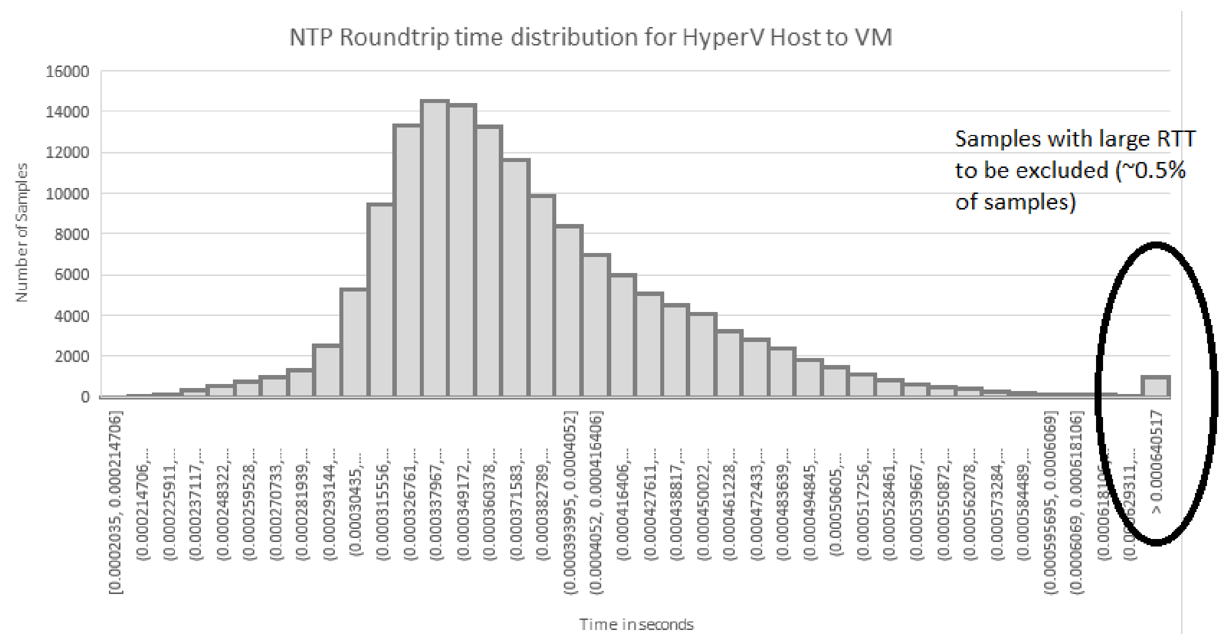


Figure 79 NTP Ping Roundtrip time distribution (Hyper-V Host to VM), Server 2016 Child Dom 2016 Client (NTP scenario)

A closer look at the filtered NTP data shows that it is noisy at times when compared to the data from the Root PDC, which is due to the cumulative effects of noise after 2 strata. However, the time on this node stays within 1ms as well.

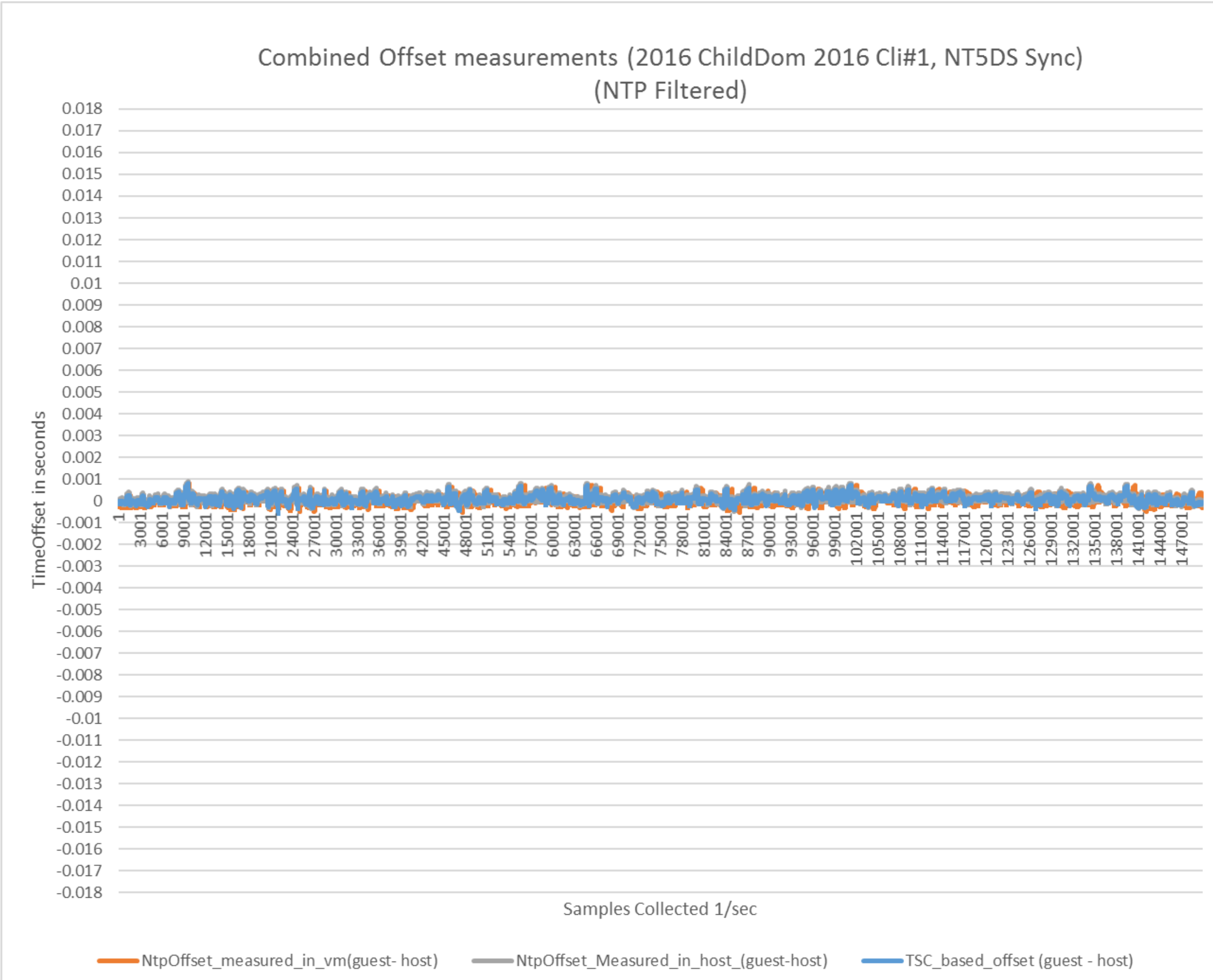


Figure 80 Filtered Offset Times - Server 2016 Child Dom 2016 Client (NTP scenario)

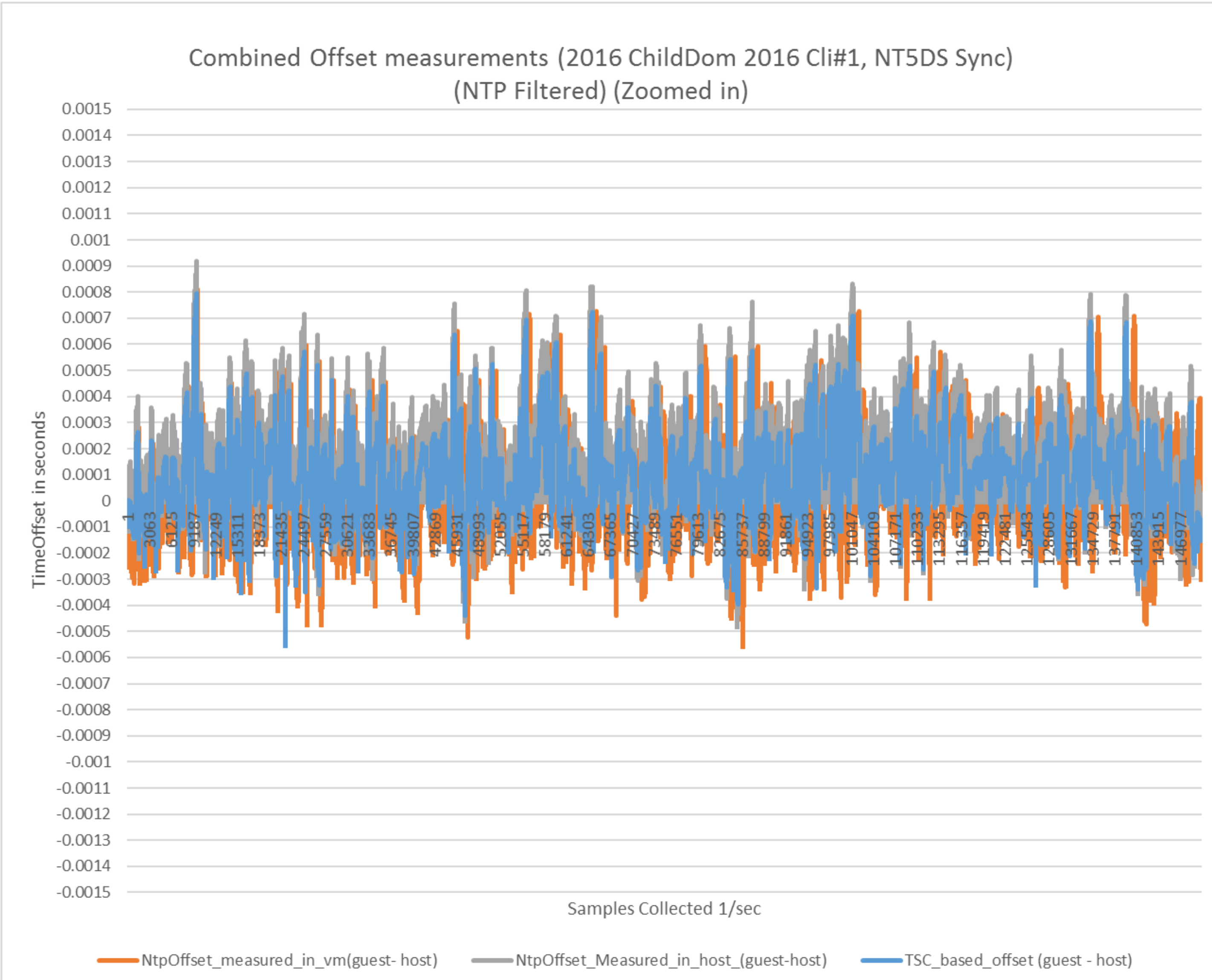


Figure 81 Filtered Offset Times - Server 2016 Child Dom 2016 Client (NTP scenario) (Zoomed in)

Server 2012 R2 Forest using VMIC Root Time Source (Scenario #3)

Root PDC (Server2012 R2)

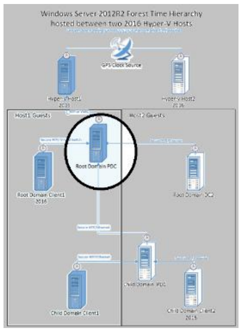


Figure 82 Topology Reference

Using the VMIC provider in Server2012R2 results in a sawtooth pattern of noise, peaking at ~15ms. The time offset measured from within the VM follows the offset computed using TSC data. However, the time offset of the VM measured remotely seems very accurate. This is a false reading observed when obtaining time from this server remotely, as you will observe in the data in this doc. The value read depends greatly on timing and if we get the timing just right, this is what is observed.

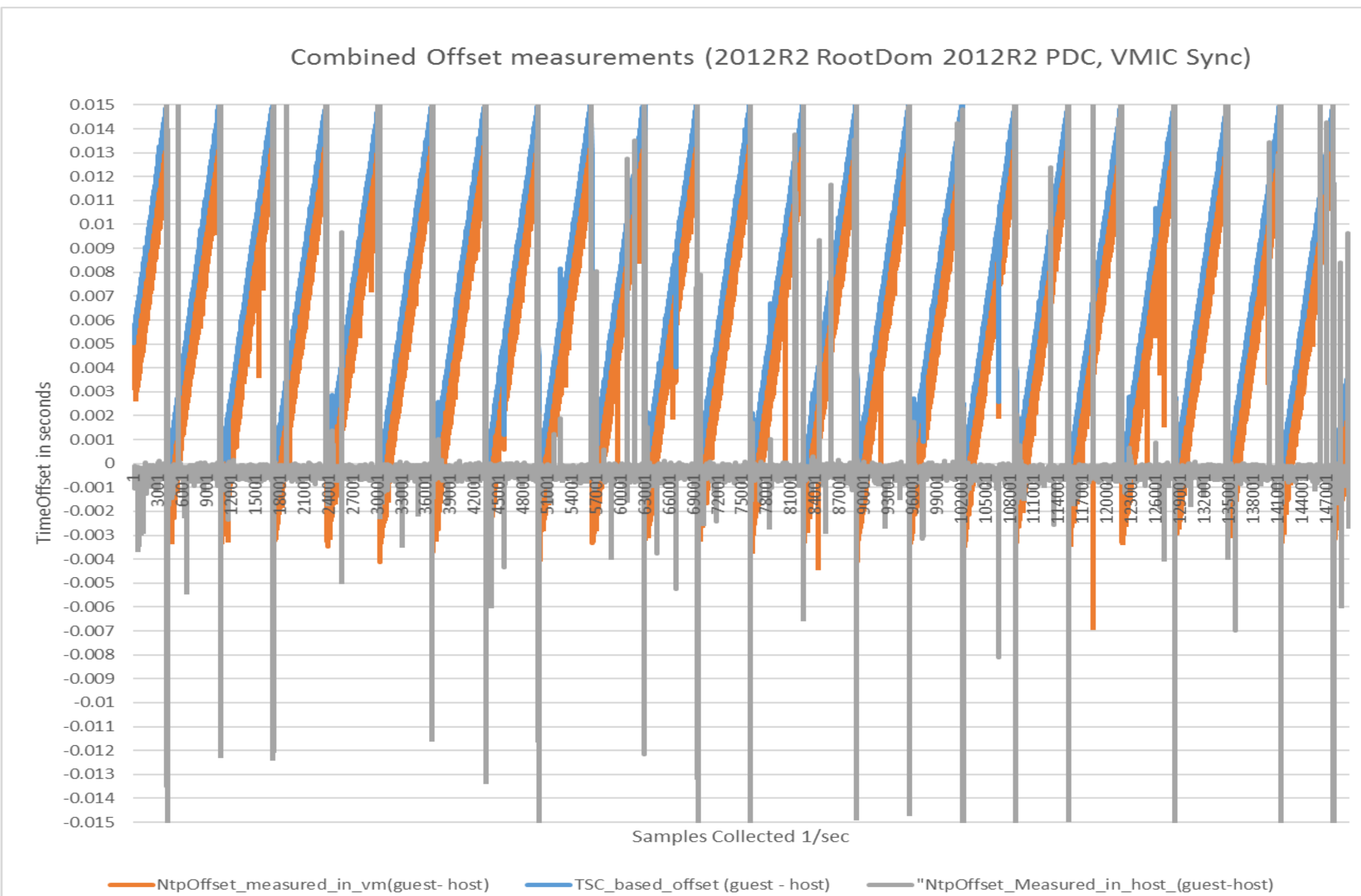


Figure 83 Offset Measurements - Server 2012 R2 RootDom PDC (VMIC scenario)

Root Domain Replica DC (Server 2012 R2)

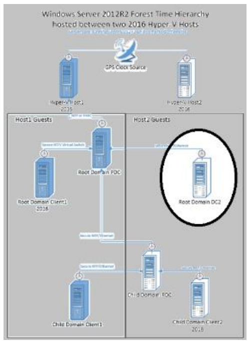


Figure 84 Topology Reference

This DC gets its time from the root PDC over Ethernet/virtualNIC. You can observe that additional noise is added to noisy time kept by root PDC and the offset is over 15ms. Stripchart (NTP) data from collected from within the VM against external server closely tracks the data from TSC Offset, but the W32tm "stripchart" (NTP) data collected externally, against this VM is off by several milliseconds. (As seen by the blue peeking out behind the grey in the chart below).

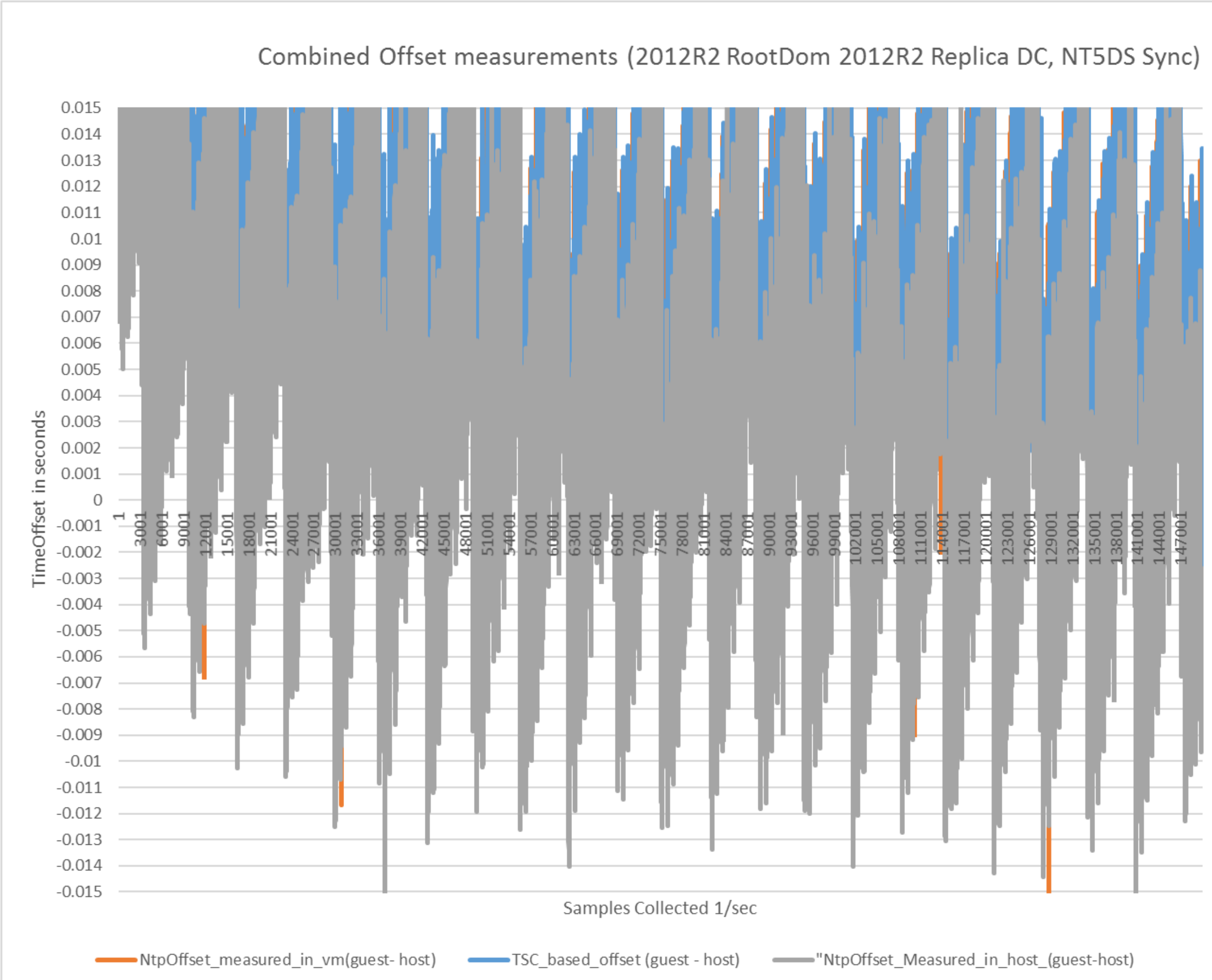


Figure 85 Offset Measurements - Server 2012 R2 RootDom Replica DC (VMIC scenario)

Root Domain Client #1 (Server 2016)

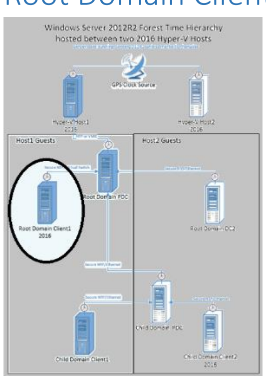


Figure 86 Topology Reference

This 2016 client is syncing its time from the 2012R2 Root PDC over the virtual switch. Here you can observe the effects of a bad upstream time server on an accurate NTP client. The W32tm "stripchart" (NTP) data from within and outside the VM seem to closely track the TSC offset data here. There is noise in this data, but considerably smaller than the offsets we are seeing and no filtering is required.

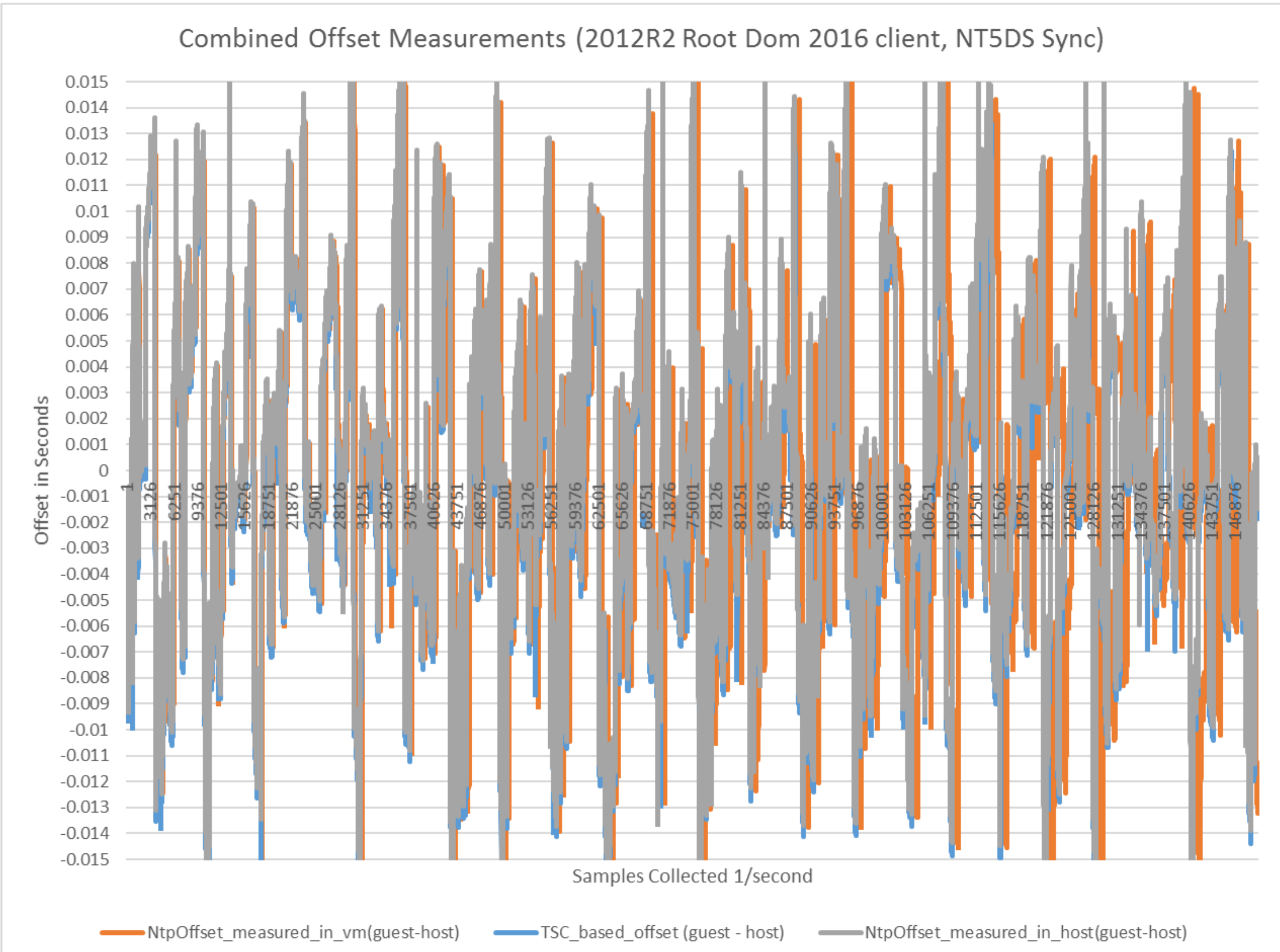


Figure 87 Offset Measurements - Server 2012 R2 Root Dom 2016 Client (VMIC scenario)

Child Domain PDC (Server2012 R2)

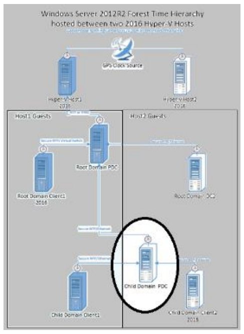


Figure 88 Topology Reference

The Child Domain PDC is in a comparable position to the replica DC above. You will notice that this has produced slightly different results from that computer. The W32tm “stripchart” (NTP) data in both measurements seems to track the TSC measurements, but it is more of a coincidence that the W32tm “stripchart” (NTP) data measured from outside the VM matches the other two measurements. We cannot guarantee that these results are exactly repeatable. In any case, the measured offset is in a wide range.

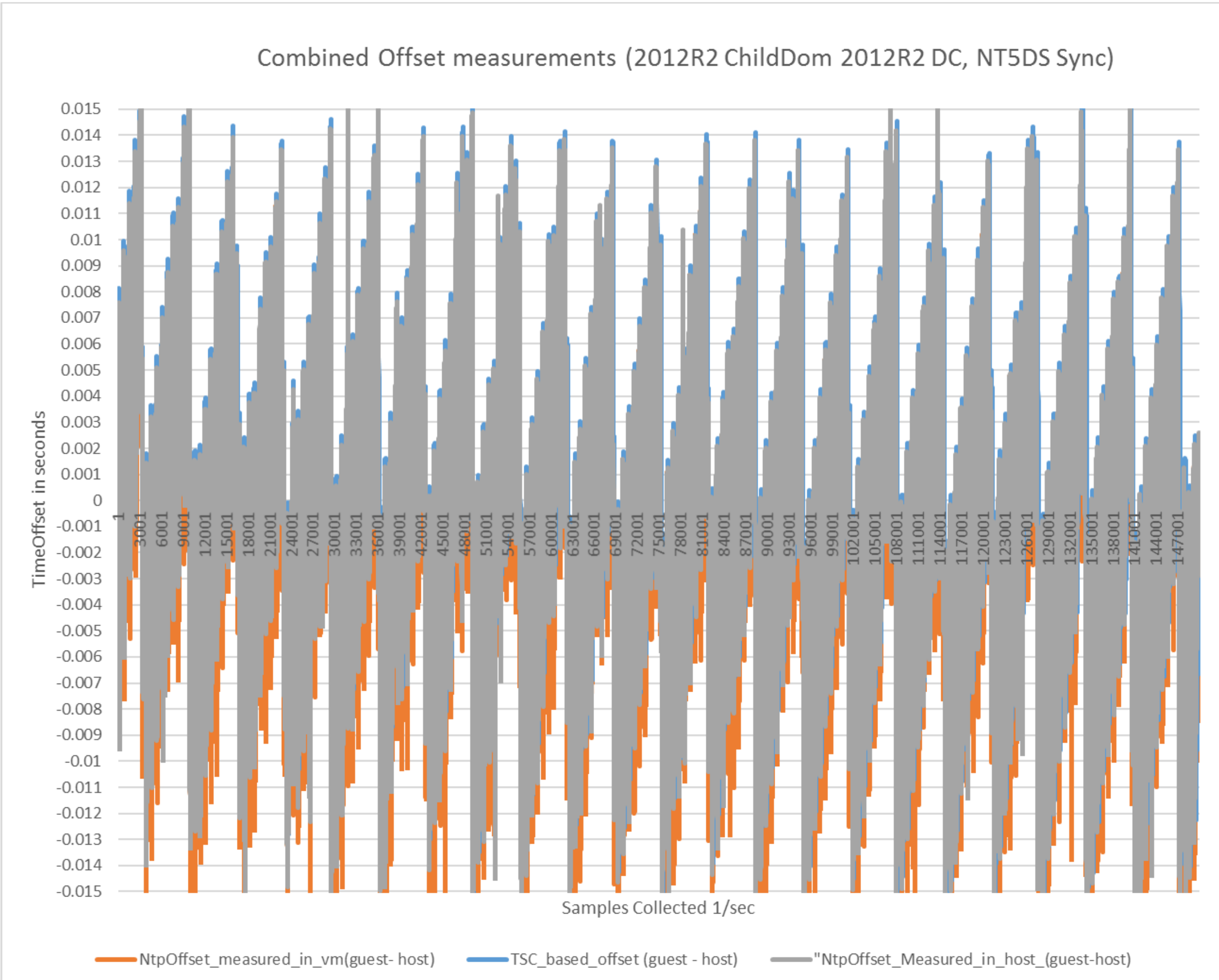


Figure 89 Offset Measurements - Server 2012 R2 Child Dom PDC (VMIC scenario)

Child Domain Client #1 (Server2012 R2)

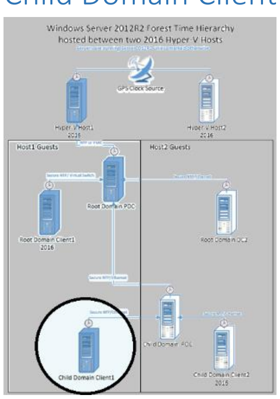


Figure 90 Topology Reference

These measurements again show that the W32tm “stripchart” (NTP) data measured from outside the VM can vary widely and cannot be relied upon for accuracies less than 15-20ms. The W32tm “stripchart” (NTP) data measured from within the VM seems to track the TSC offset data fairly closely.

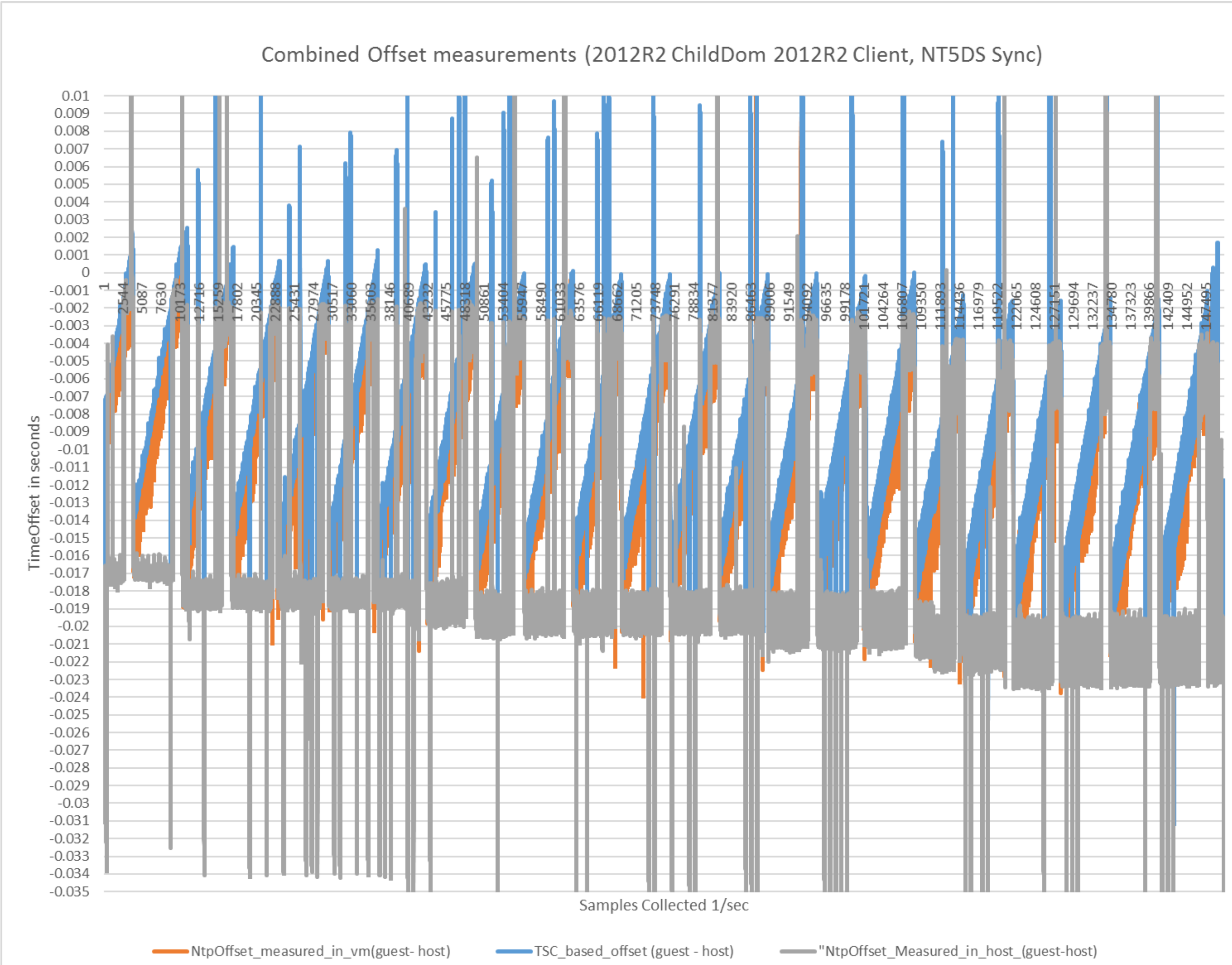


Figure 91 Offset Measurements - Server 2012 R2 Child Dom 2012R2 Client (VMIC scenario)

Child Domain Client #2 (Server 2016)

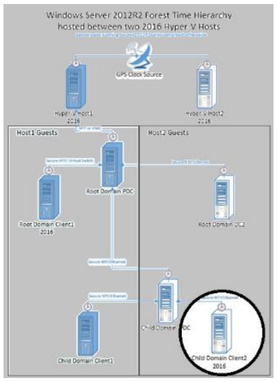


Figure 92 Topology Reference

This Server2016 client was added to demonstrate the effects of syncing time over 2 strata of Server2012R2 DCs. You can see that the measurements closely track each other, but the time can be off by ~30ms due to the inaccurate time servers in the way.

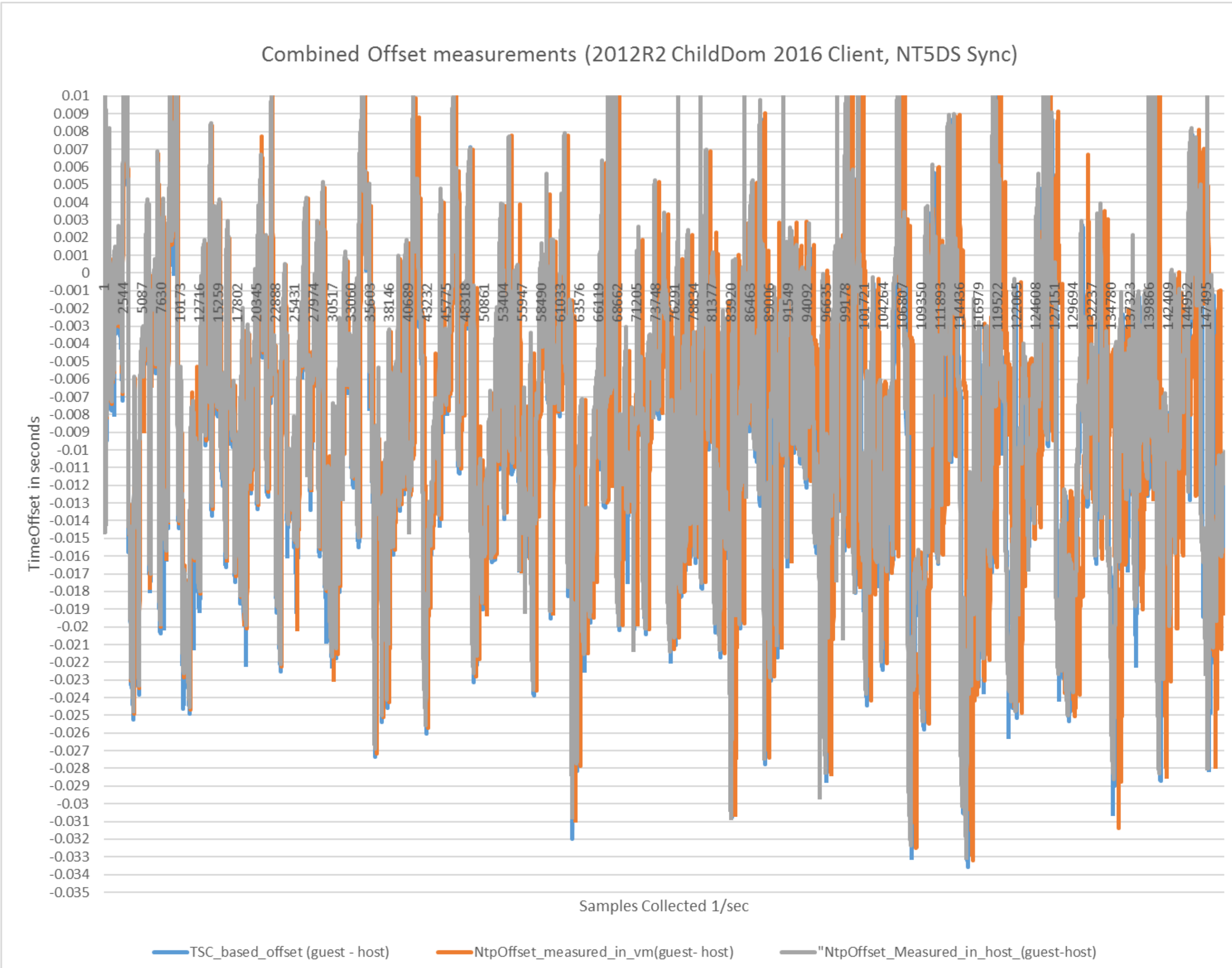


Figure 93 Offset Measurements - Server 2012 R2 Child Dom 2016 Client (VMIC scenario)

Server 2012 R2 Forest using NTP Root Time Source (Scenario #4)

We usually recommend our customers to use an accurate NTP Server as their root time source for their AD Forests and this is an interesting to see how this performs compared to other scenarios.

Root PDC (Server2012 R2)

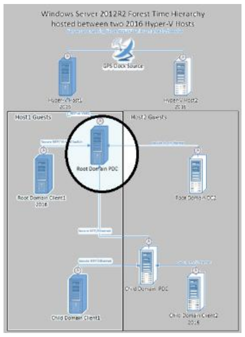


Figure 94 Topology Reference

As indicated by the TSC offset data in the chart below, the accuracy on this 2012R2 machine looks much better than any machine in Scenario#3, although it is not in the sub-millisecond range. Stripchart (NTP) data measured from within the VM against the Hyper-V host shows an error of ~4ms from the local time. Stripchart (NTP) data measured from the Hyper-V host against this VM shows a saw tooth pattern with ~17ms peak. Essentially the data returned by the NTPServer in the VM is inaccurate up to ~17ms and the saw tooth pattern is a coincidental. However, this is the data that downstream clients will receive from this server and they will inevitably be more inaccurate than what we are seeing here.

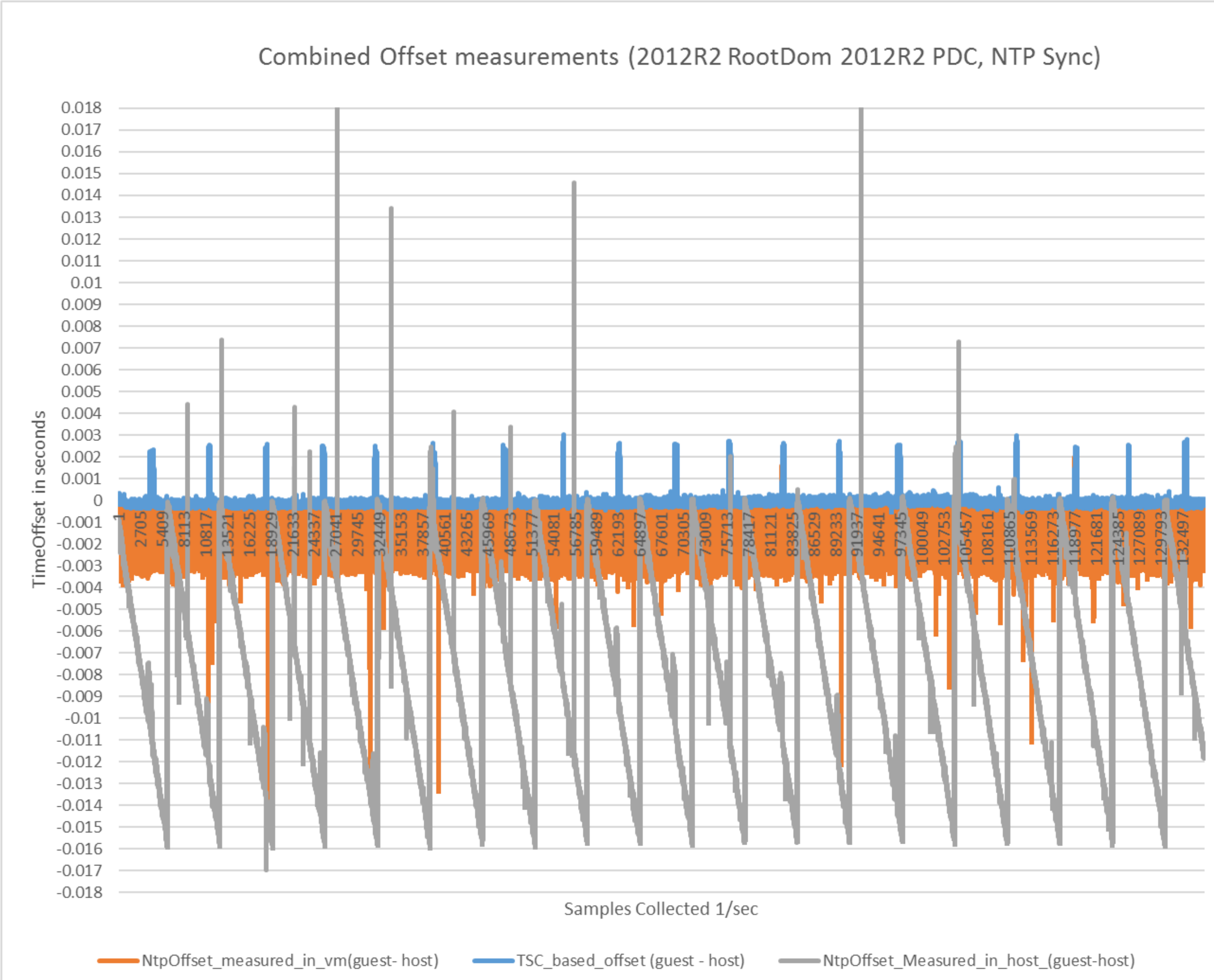


Figure 95 Offset Measurements - Server 2012 R2 Roo tDom PDC (NTP scenario)

Root Domain Replica DC (Server 2012 R2)

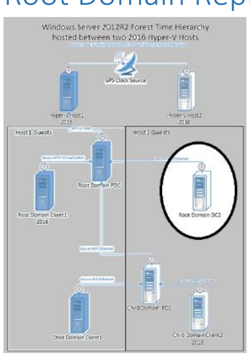


Figure 96 Topology Reference

This DC's time sync state is better than the same DC in scenario #1, but the overall spread is ~12ms peak-to-peak. As expected, a few milliseconds of additional noise has creeped in. The W32tm "stripchart" (NTP) measurements between the Hyper-V host VM doesn't show a saw-tooth pattern, but still doesn't follow closely the actual time inside the VM.

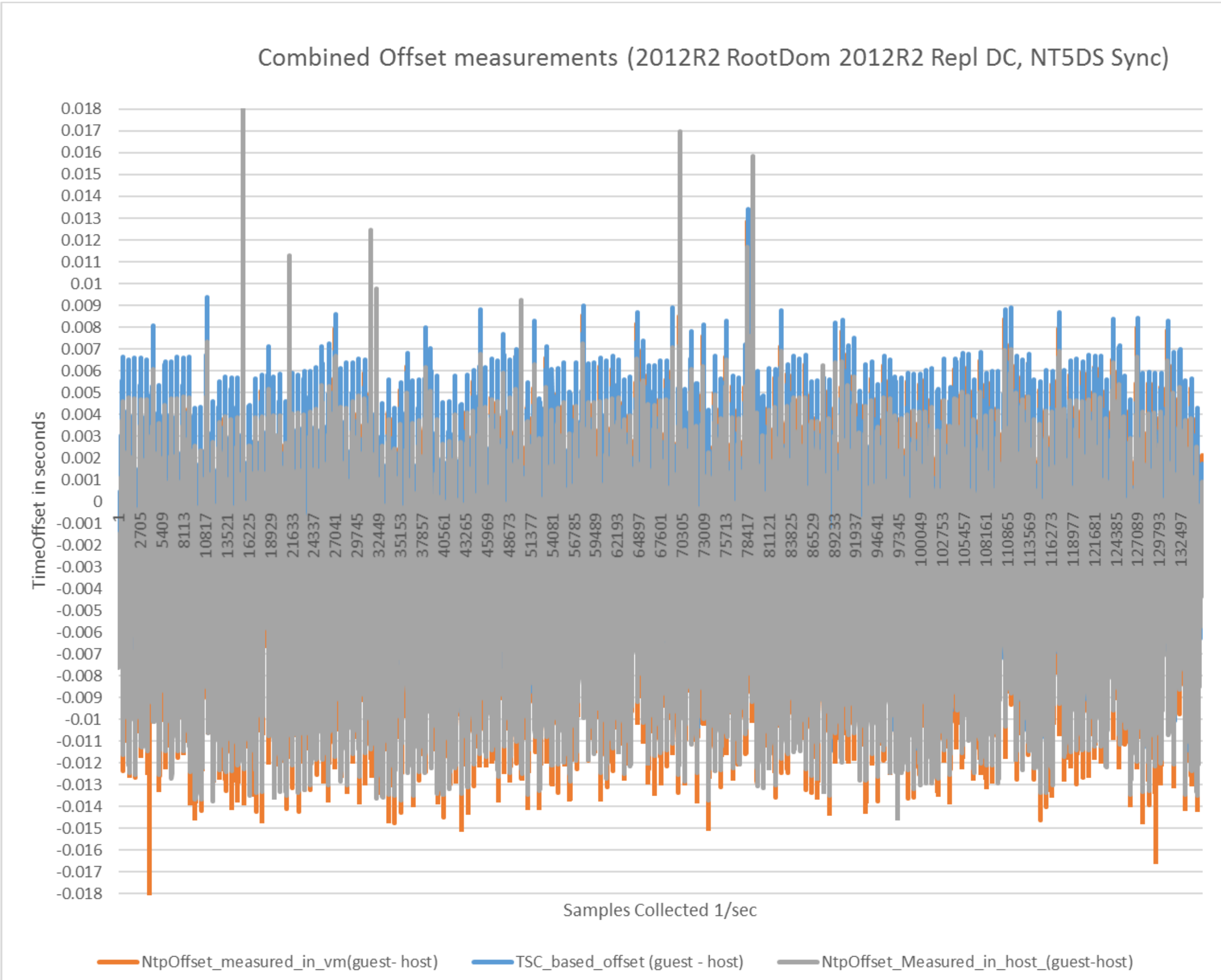


Figure 97 Offset Measurements - Server 2012 R2 Root Dom Replica DC (NTP scenario)

Root Domain Client #1 (Server 2016)

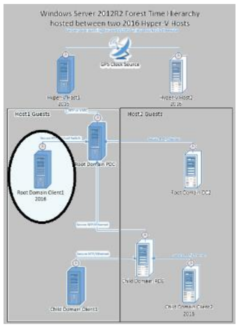


Figure 98 Topology Reference

This client is syncing its time from the Root PDC over the virtual switch. Although the Root PDC is keeping good time, it is unable to pass that on downstream and you can observe the effects on an accurate NTP client. The overall noise is less on this machine when we compare this the same machine in Scenario #1. The W32tm “stripchart” (NTP) data from within and outside the VM seem to closely track the TSC offset data here. There is noise in this data, but considerably smaller than the offsets we are seeing and we can ignore that for practical purposes.

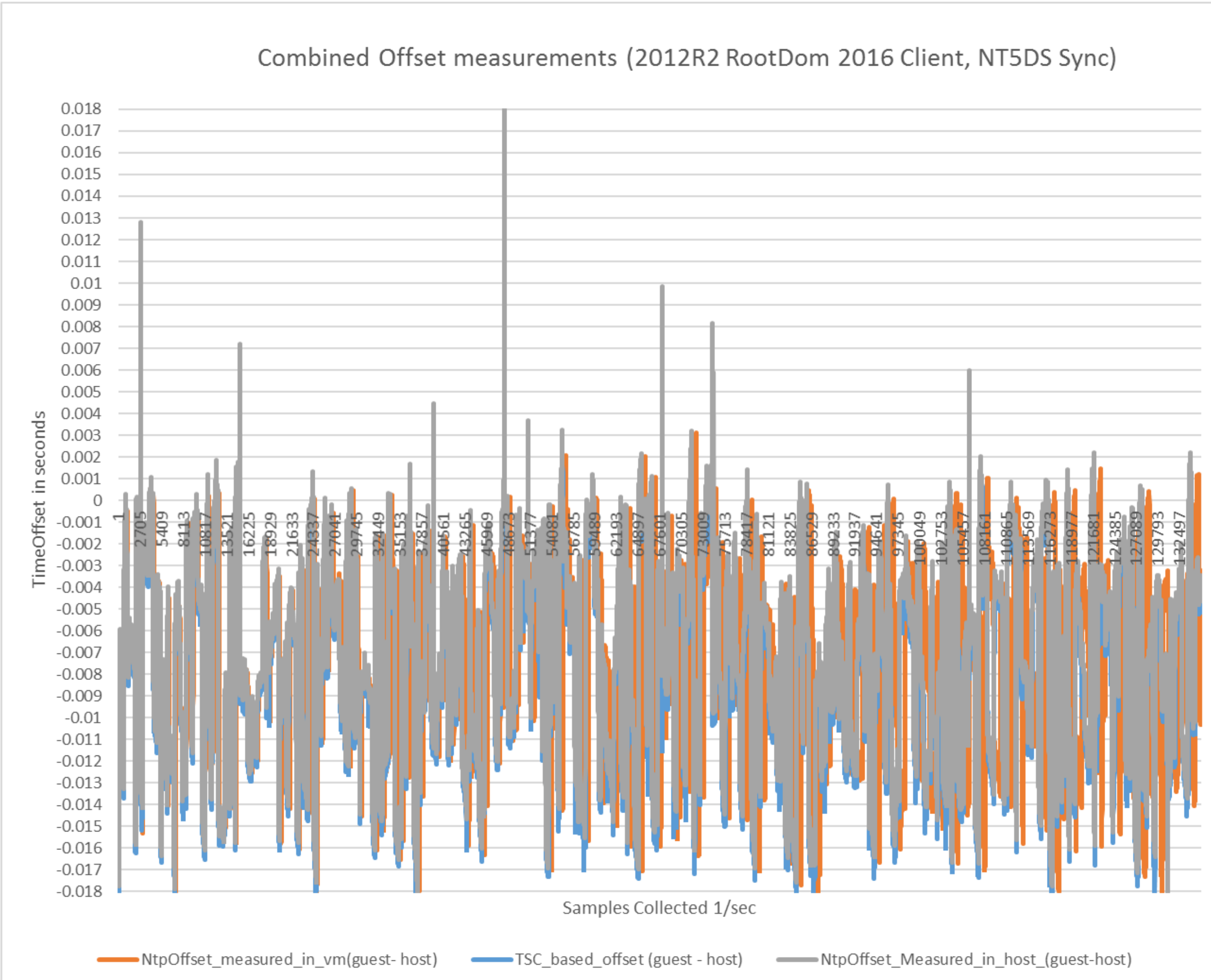


Figure 99 Offset Measurements - Server 2012 R2 Root Dom 2016 Client (NTP scenario)

Child Domain PDC (Server2012 R2)

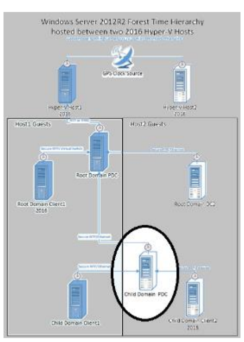


Figure 100 Topology Reference

The Child Domain PDC is in a comparable position to the replica DC above. You will notice that this has produced slightly different results from that computer. The W32tm “stripchart” (NTP) data in both measurements seems to track the TSC measurements, but the W32tm “stripchart” (NTP) data measured from outside the VM is off of the other two measurements by almost a constant value. Another data point to show that the NTPServer data in Server2012R2 can vary widely compared to the underlying system time.

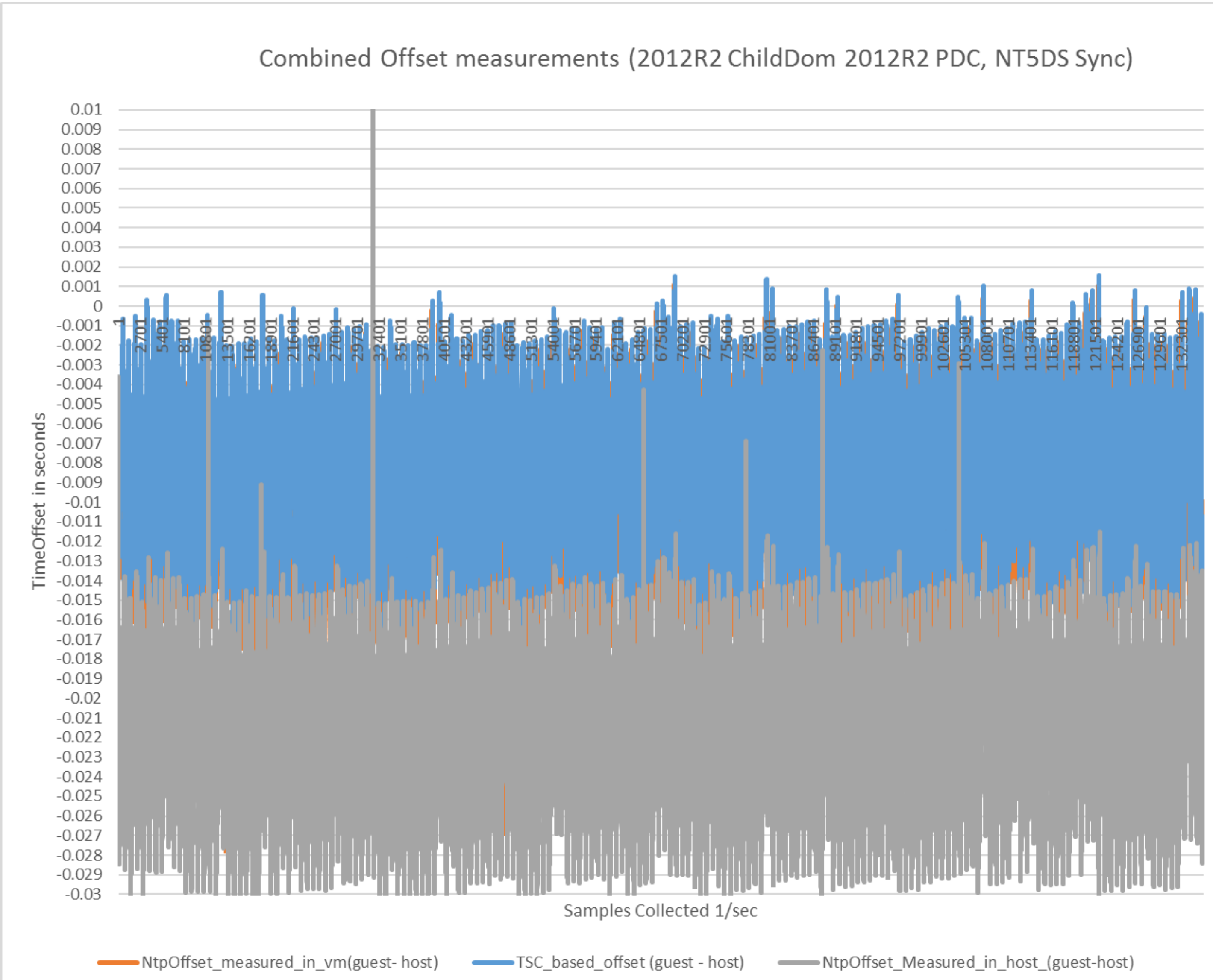


Figure 101 Offset Measurements - Server 2012 R2 Child Dom 2012R2 PDC (NTP scenario)

Child Domain Client #1 (Server2012 R2)

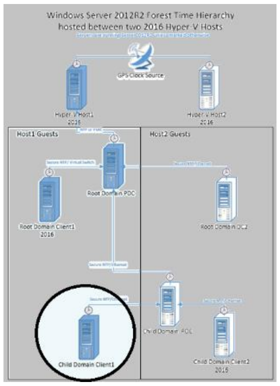


Figure 102 Topology Reference

After two hops, the time on this machine is off by almost a constant ~15 ms. These measurements again show that the W32tm “stripchart” (NTP) data measured from outside the VM can vary widely and cannot be relied upon for accuracies less than 15-20ms. The W32tm “stripchart” (NTP) data measured from within the VM seems to track the TSC offset data within ~3ms offset.

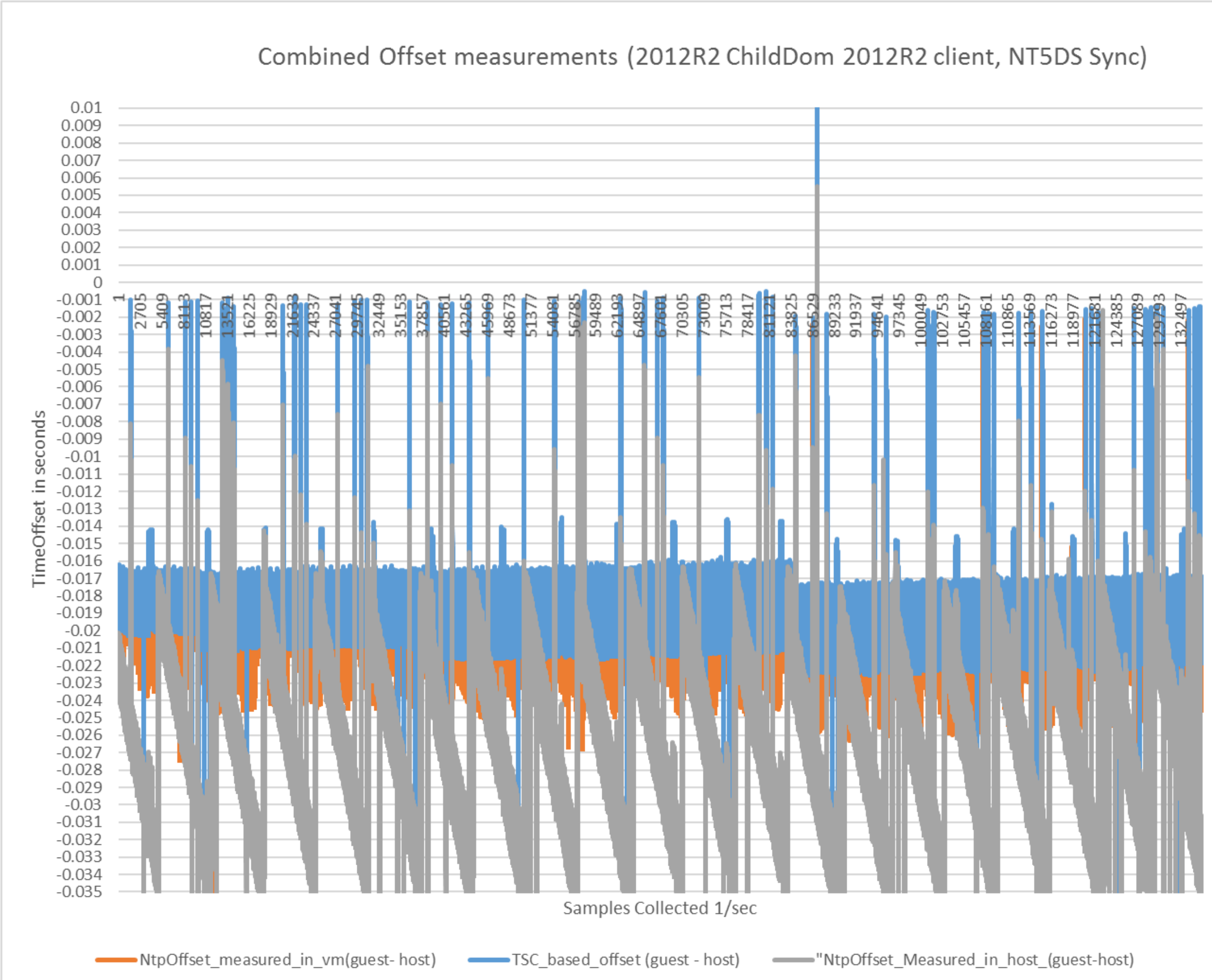


Figure 103 Offset Measurements - Server 2012 R2 Child Dom 2012 R2 Client (NTP scenario)

Child Domain Client #2 (Server 2016)

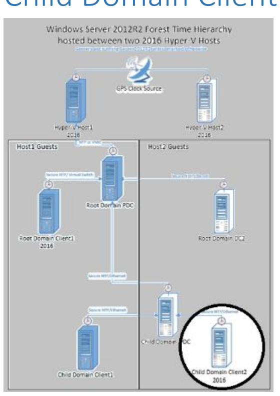


Figure 104 Topology Reference

This Server2016 client was added to demonstrate the effects of syncing time over 2 strata of Server2012R2 DCs. You can see that the measurements closely track each other, but the time can be off by ~30ms due to the inaccurate time servers in the way.

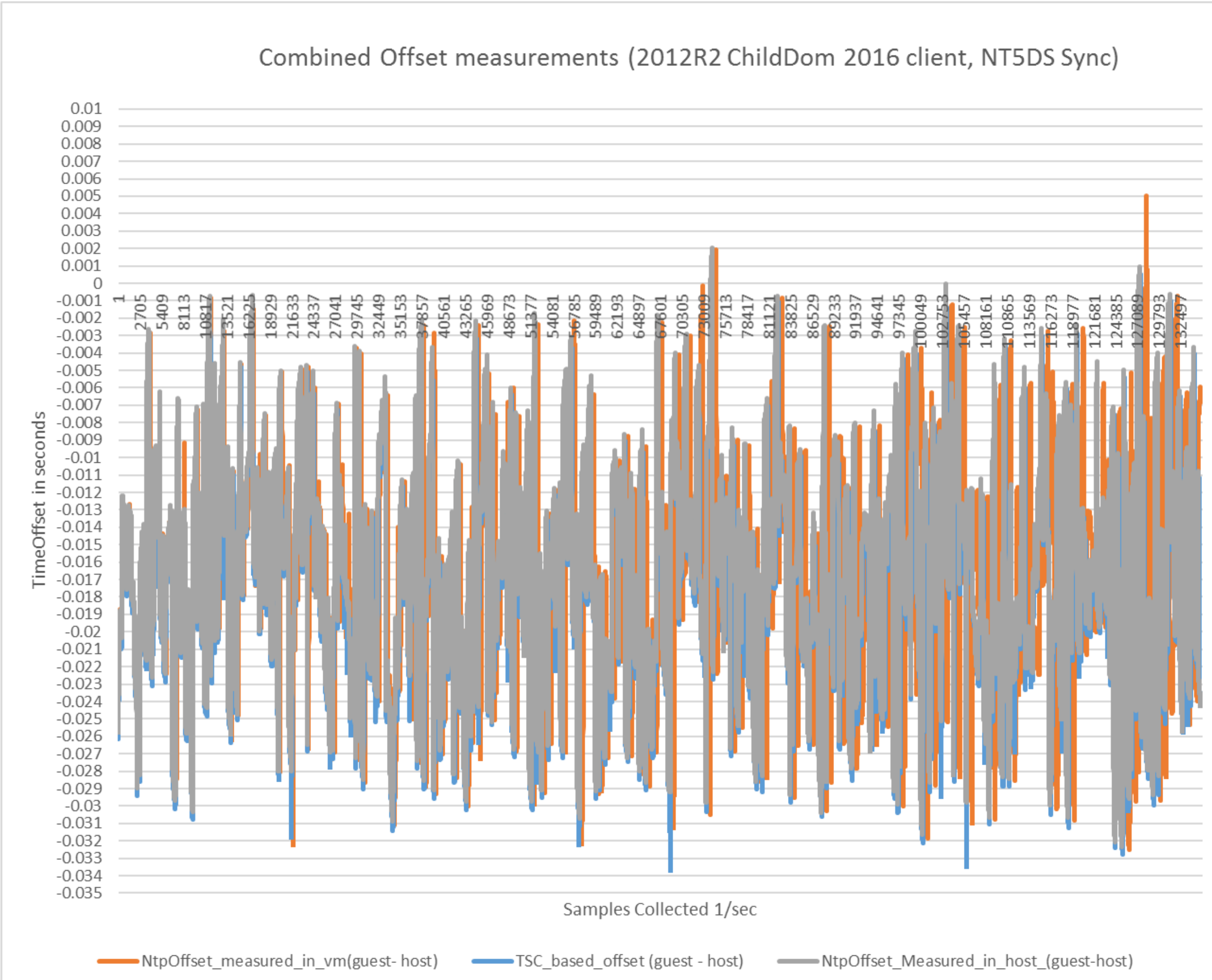


Figure 105 Offset Measurements - Server 2012 R2 Child Dom 2016 Client (NTP scenario)

Server 2016 NTP "Quarter mile" Test

Unsecured/Simple NTP gives better time sync performance than the secure NTP (NT5DS mode) used in AD Domains due to the additional security overhead in the latter case. (We can safely assume this to be the case for any secure time sync protocol.)

This test intends to show the limits of using Simple NTP to sync time in Server 2016 from an accurate time source.

The machine used for testing was a physical machine sitting in a Microsoft office and the accurate time server was located in a different Microsoft building about a quarter mile away. Hence the name "Quarter mile test". There are 6 generic network switches between the test machine and the accurate time source. The test machine is configured with the accurate time source as its manual peer and with the rest of the settings being the same as the recommended high accuracy time settings.

Here is the raw NTP offset data as measured in the machine against the accurate time source.

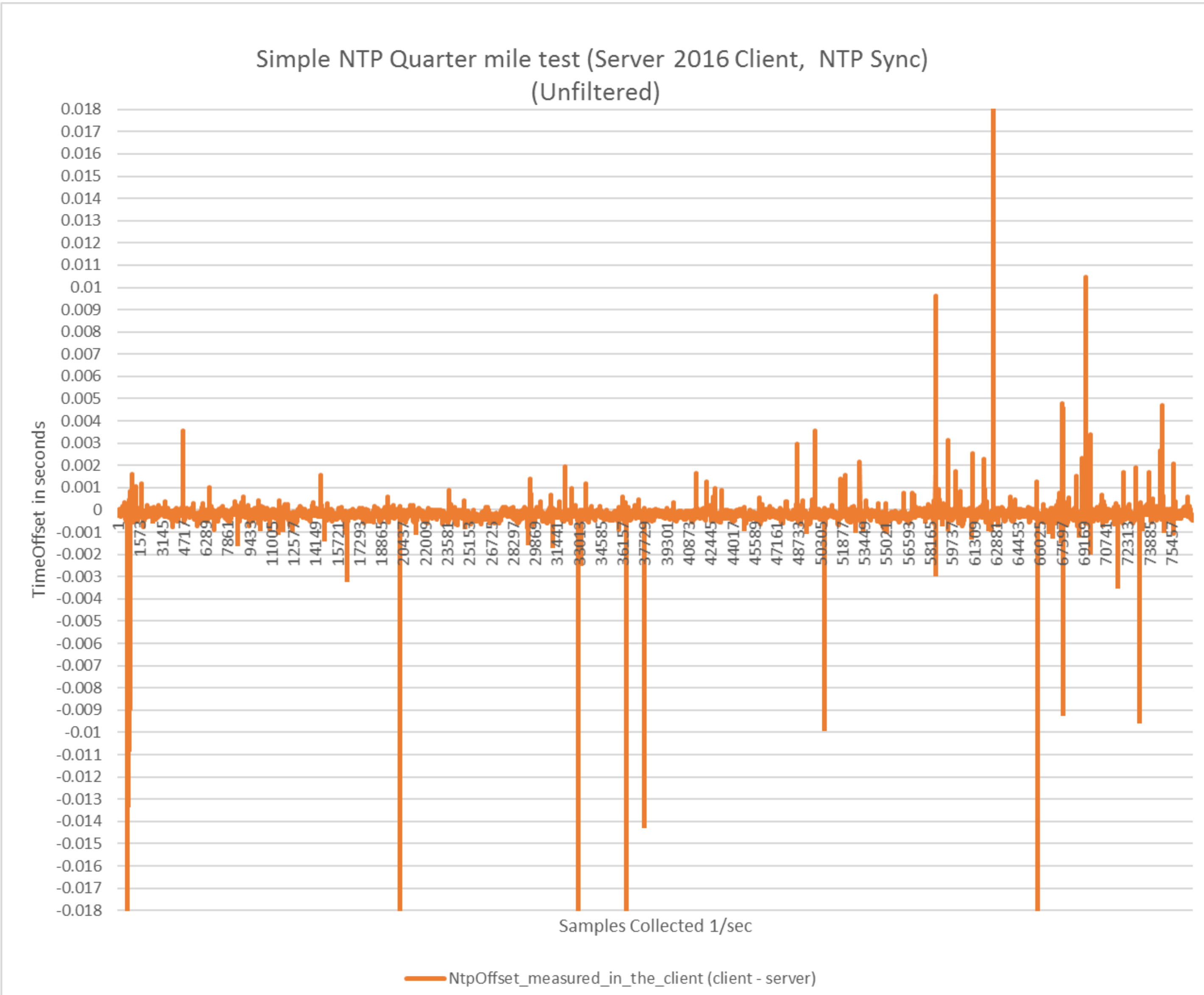


Figure 106 NTP Offset measured from the test machine against accurate NTP Server (Quarter mile test) (Unfiltered)

Here is the distribution of NTP roundtrip times and the samples that would be excluded by the Popcorn filter.

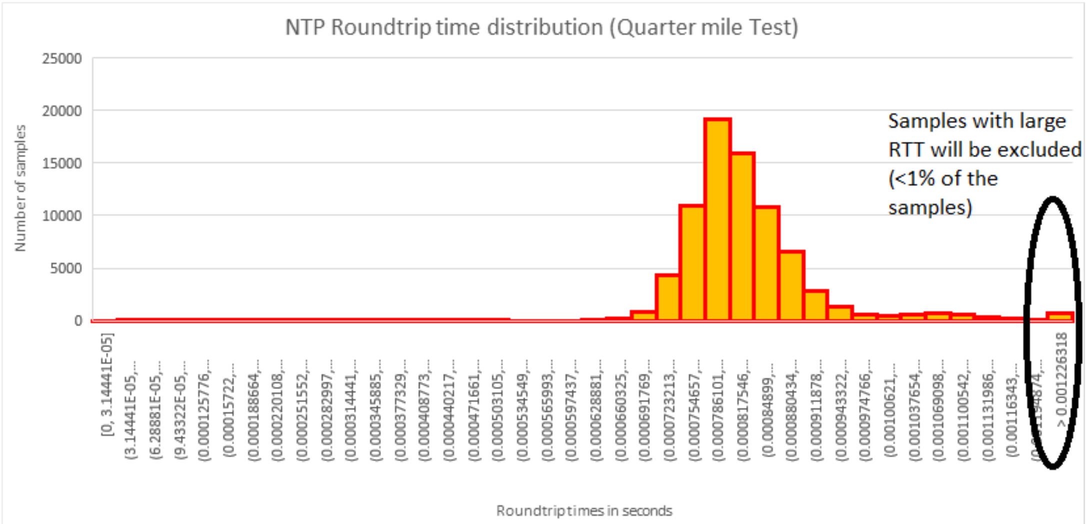


Figure 107 Distribution of NTP roundtrip times (Quarter Mile Test)

This the NTP offset of samples that have passed through the Popcorn filter.

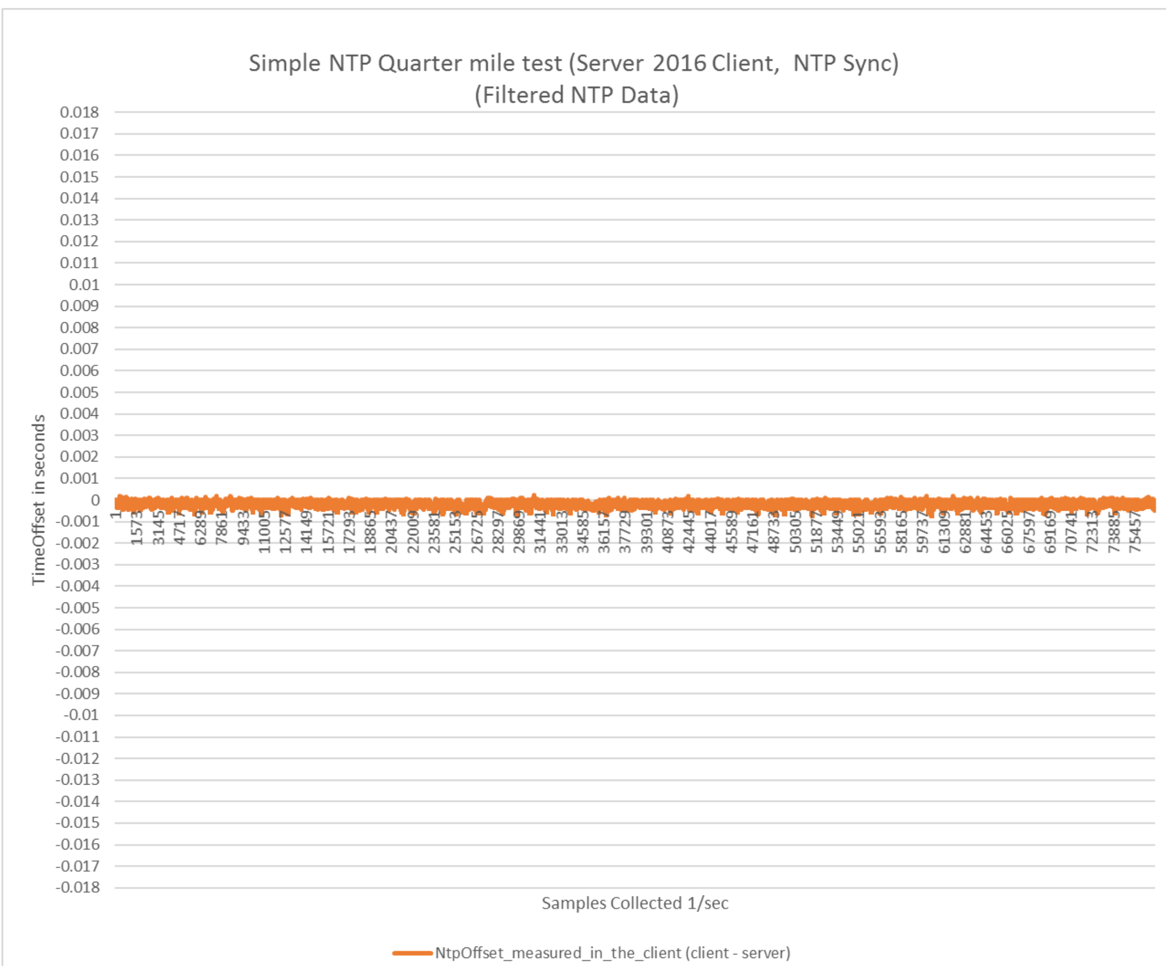


Figure 108 Filtered NTP offset (Quarter mile test)

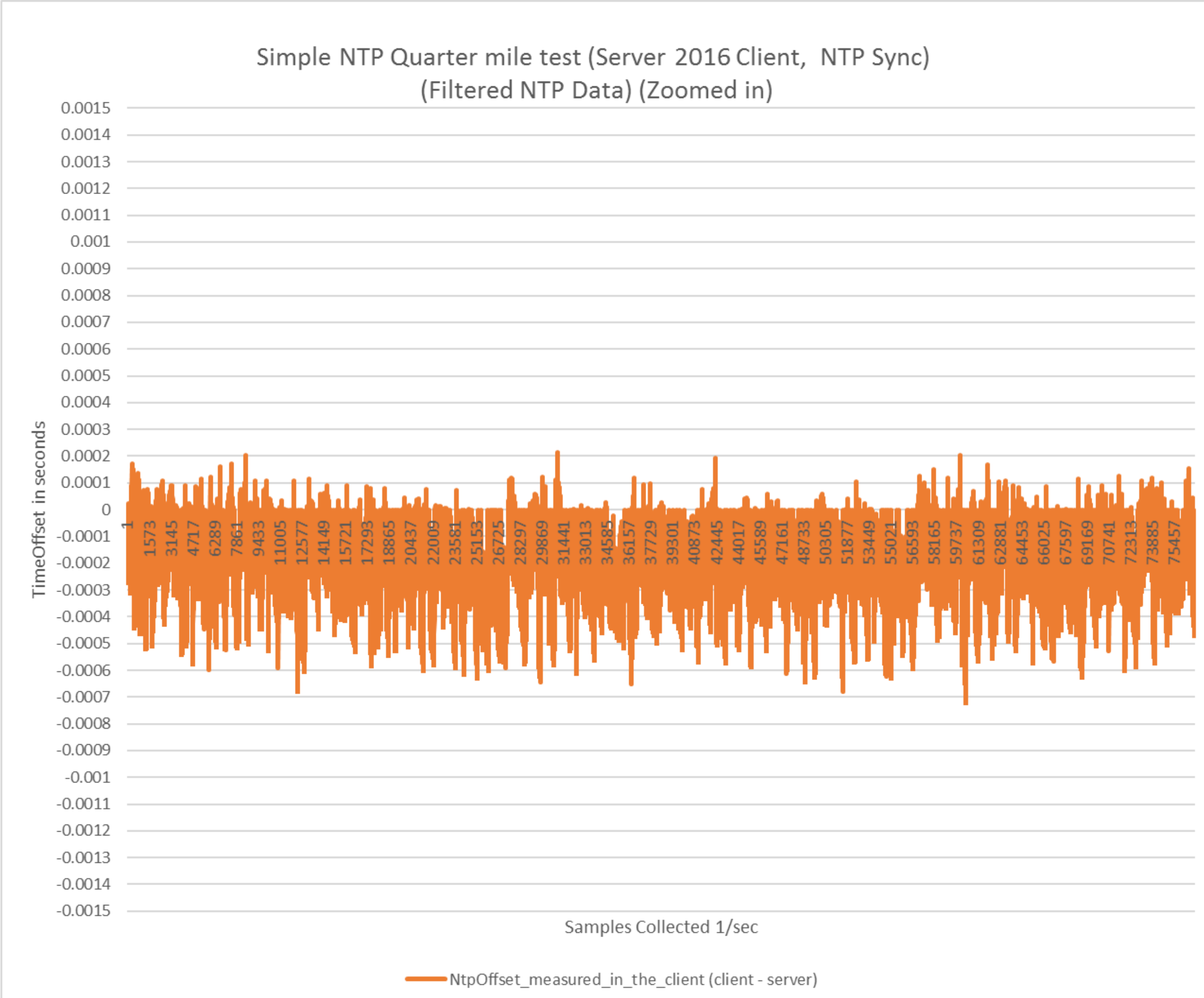


Figure 109 Filtered NTP Offset (Quarter mile test) (Zoomed in)

Scripts

Script #1 Setting up common high accuracy settings

```
@echo off

echo Setting the polling interval to 64 seconds
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w32time\Config /v MinPollInterval /t
REG_DWORD /d 6 /f
if %errorlevel% NEQ 0 (
   echo Failed to set the MinPollInterval in the registry. Error: %errorlevel%
   exit /b %errorlevel%
)
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w32time\Config /v MaxPollInterval /t
REG_DWORD /d 6 /f
if %errorlevel% NEQ 0 (
   echo Failed to set the MaxPollInterval in the registry. Error: %errorlevel%
```

```
    exit /b %errorlevel%
)

echo Setting the update interval to 1 second
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w32time\Config /v UpdateInterval /t
REG_DWORD /d 100 /f
if %errorlevel% NEQ 0 (
    echo Failed to set the UpdateInterval in the registry. Error: %errorlevel%
    exit /b %errorlevel%
)

echo Signaling W32time to pick up the updated settings
w32tm /config /update
if %errorlevel% NEQ 0 (
    echo Failed to signal w32time. Error: %errorlevel%
    echo The service may not be running. Attempting to start the service.
    net start w32time
)
if %errorlevel% NEQ 0 (
    echo Failed to restart W32time. Error: %errorlevel%
    exit /b %errorlevel%
)

echo done
```

Script #2 Enable VMIC Provider Only

```
@echo off

echo Mark VMIC provider as enabled
reg add
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w32time\TimeProviders\VMICTimeProvider /v
Enabled /t REG_DWORD /d 1 /f
if %errorlevel% NEQ 0 (
    echo Failed to enable VMICTimeProvider in the registry. Error: %errorlevel%
    exit /b %errorlevel%
)

echo Mark NTPClient provider as disabled
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w32time\TimeProviders\NtpClient /v
Enabled /t REG_DWORD /d 0 /f
if %errorlevel% NEQ 0 (
    echo Failed to disable NTPClient provider in the registry. Error: %errorlevel%
    exit /b %errorlevel%
)

echo Restarting W32time Service
net stop w32time && net start w32time
REM W32time was stopped and the above command failed. So just start the service
if %errorlevel% EQU 2 (
    net start w32time
)
if %errorlevel% NEQ 0 (
    echo Failed to restart W32time. Error: %errorlevel%
    exit /b %errorlevel%
)

echo done
```

Script #3 Enable NTP Client Only with manual server

```
@echo off

echo Mark NTPClient provider as enabled
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w32time\TimeProviders\NtpClient /v
Enabled /t REG_DWORD /d 1 /f
if %errorlevel% NEQ 0 (
    echo Failed to enable NTPClient provider in the registry. Error: %errorlevel%
    exit /b %errorlevel%
)

echo Mark VMIC provider as disabled
```

```
reg add
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w32time\TimeProviders\VMICTimeProvider /v
Enabled /t REG_DWORD /d 0 /f
if %errorlevel% NEQ 0 (
   echo Failed to disable VMICTimeProvider in the registry. Error: %errorlevel%
   exit /b %errorlevel%
)

echo Set the manual NTP server
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w32time\Parameters /v NtpServer /t
REG_SZ /d time.windows.com,0x8 /f
if %errorlevel% NEQ 0 (
   echo Failed to set the manual NTP Server in the registry. Error: %errorlevel%
   exit /b %errorlevel%
)

echo Set the sync type to NTP
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w32time\Parameters /v Type /t REG_SZ
/d NTP /f
if %errorlevel% NEQ 0 (
   echo Failed to set the sync typte to NTP in the registry. Error: %errorlevel%
   exit /b %errorlevel%
)

echo Restarting W32time Service
net stop w32time && net start w32time
REM W32time was stopped and the above command failed. So just start the service
if %errorlevel% EQU 2 (
   net start w32time
)
if %errorlevel% NEQ 0 (
   echo Failed to restart W32time. Error: %errorlevel%
   exit /b %errorlevel%
)

echo Done
```

Script #4 Enable NTP Client Only with Domain Sync

```
@echo off

echo Mark NTPClient provider as enabled
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w32time\TimeProviders\NtpClient /v
Enabled /t REG_DWORD /d 1 /f
if %errorlevel% NEQ 0 (
   echo Failed to enable NTPClient provider in the registry. Error: %errorlevel%
   exit /b %errorlevel%
)

echo Mark VMIC provider as disabled
reg add
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w32time\TimeProviders\VMICTimeProvider /v
Enabled /t REG_DWORD /d 0 /f
if %errorlevel% NEQ 0 (
   echo Failed to disable VMICTimeProvider in the registry. Error: %errorlevel%
   exit /b %errorlevel%
)

echo Delete the manual NTP server configuration
reg delete HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w32time\Parameters /v NtpServer
/f
if %errorlevel% NEQ 0 (
   echo Ignoring error in deleting the manual NTP server setting in the registry. Error:
%errorlevel%
)

echo Set the sync type to NT5DS
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w32time\Parameters /v Type /t REG_SZ
/d NT5DS /f
if %errorlevel% NEQ 0 (
   echo Failed to set the sync typte to NTP in the registry. Error: %errorlevel%
   exit /b %errorlevel%
)

echo Restarting W32time Service
net stop w32time && net start w32time
REM W32time was stopped and the above command failed. So just start the service
```

```
if %errorlevel% EQU 2 (
   net start w32time
)
if %errorlevel% NEQ 0 (
   echo Failed to restart W32time. Error: %errorlevel%
   exit /b %errorlevel%
)

echo Done
```

Script #5 Enable NTP Server

```
@echo off

echo Enable NTPServer provider
reg add HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\w32time\TimeProviders\NtpServer /v
Enabled /t REG_DWORD /d 1 /f
if %errorlevel% NEQ 0 (
   echo Failed to enable NTPServer provider in the registry. Error: %errorlevel%
   exit /b %errorlevel%
)

echo Restarting W32time Service
net stop w32time && net start w32time
REM W32time was stopped and the above command failed. So just start the service
if %errorlevel% EQU 2 (
   net start w32time
)
if %errorlevel% NEQ 0 (
   echo Failed to restart W32time. Error: %errorlevel%
   exit /b %errorlevel%
)

echo Done
```

Index of Figures