How Precise Is the Clock in the Cloud?

Gee-Il Han, Filip Rezabek, Leander Seidlitz*
*Chair of Network Architectures and Services
School of Computation, Information and Technology, Technical University of Munich, Germany
Email: geeil.han@tum.de, frezabek@net.in.tum.de, seidlitz@net.in.tum.de

***Abstract*—A precise clock is a necessary element in a system and most devices have their own internal clock. Over time these clocks experience clock drift due to influences such as hardware variations or operating conditions. As a result, the time of internal clocks varies from the time they are supposed to be. Differences between internal clocks of cloud servers and devices can result in data inconsistencies and authentication errors, ultimately impacting the reliability and security of distributed systems.**

To overcome these deviations protocols are used, for instance, network time protocol (NTP), precision time protocol (PTP), or simple precision time protocol (SPTP). Some providers already implemented these protocols and can be used for clock synchronization, e.g., Google Public NTP, Amazon Time Sync, or Alibaba Cloud Time Synchronization Service. They are used to achieve synchronized clocks between the server and client.

***Index Terms*—NTP, network time protocol, PTP, precision time protocol, SPTP, Simple precision time protocol, synchronized clock, time sync service**

1. Introduction

Before devices within a system try to synchronize their clocks, they communicate with each other through data packets, containing timestamps of the sender's device and more to ensure its chronological order for time-sensitive data. The time values are read from an individual clock to each device.

However, a problem occurs when the clocks in a system are not synchronized. These deviations are called clock drifts and can occur for the following reasons:

Hardware variations between the components in each system result in different magnitudes of deviations leading to varying aberrations for every system [1].

Operating conditions in the context of clock consistency describe multiple influences on the clock through environmental factors. One of them is the electronic component aging. It decays over time, altering the components' electrical properties and causing changes in their frequency characteristics. This effect causes deviations in the clock. Another operating condition affecting the clock is temperature fluctuation which causes the materials in the clock to expand or contract, therefore affecting the timekeeping mechanism. This physical phenomenon leads to shifts in the oscillation frequency [1].

The desynchronization of clocks, which is an outcome of, e.g., clock drift and network and processing latencies across processes, is followed by anomalous behavior in the system [2]. To illustrate, two computers A and B issue requests a and b with the condition that a is sent earlier than b but the timestamp of a is later than bs. This situation can cause request b to be ordered before request a even though a was sent earlier than b, which is known as anomalous behavior [2].

The problem is solved by using protocols such as the network time protocol (NTP [3]), precision time protocol (PTP [4]), and its abbreviation, the simple precision time protocol (SPTP [5]). These protocols are applied to synchronize the clocks of the devices connected to a system with its grandmaster clock. PTP and SPTP have optional-PTP-enabled hardware that supports the synchronization process which leads to a more accurate offset time in order of nanoseconds. At the same time, the NTP is only a software-deployed protocol. The precision of NTPs is in order of milliseconds (ms) or microseconds (μs) [6].

Instead of implementing a protocol, it is more efficient to use clock synchronization options already available through cloud providers since this method saves time and effort. The services researched in this paper are Google Public NTP from Google Cloud Platform (GCP [7]), Amazon Time Sync provided by Amazon Web Services (AWS [8]), and Alibaba Clouds Alibaba Cloud Time Synchronization Service [9].

This paper presents background information about NTP, PTP and SPTP and their differences. Afterwards, available clock synchronization options are presented and details are provided.

2. Background Information

In order to understand the differences between the protocols, background information about NTP, PTP, and SPTP is provided.

NTP works by continuously exchanging time information between server and client in a hierarchical order to ensure synchronization across the clocks in all nodes with an acceptable deviation of milliseconds. The grandmaster clock is called Stratum 0 and after it has been synchronized with another device, the other device is considered a Stratum 1. Every clock that is synchronized with Stratum 1 becomes Stratum 2, every clock synchronized with Stratum 2 becomes Stratum 3, and so on. [3]. A simplified version of a single cycle is presented in Figure 1.

A simple cycle of the NTP can be described as follows: The client first sends a request to the server with its current timestamp attached. The server receives the message and and sends out a response containing three

 doi: 10.2313/NET-2024-09-1_12

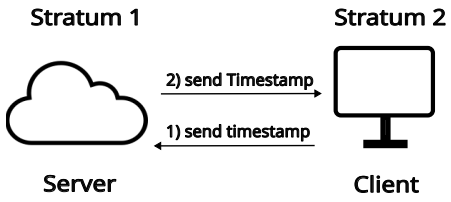

Figure 1: The Network Time Protocol [3]

timestamps: the arrival time of the message, the time when the response was sent, and the server's current time. After the client receives the response, the offset and the delay of the round-trip are calculated. Using the phase-lock loop (PLL) the clock is gradually adjusted to minimize abrupt changes [3]. This process is executed periodically in order to maintain synchronization.

PTP works by synchronizing clocks in a network to a master clock, reducing deviation to nanoseconds. This section references PTP to security extension of PTP, the PTPv2 Standard IEEE-1588-2019 [10]. During the first step of this protocol, PTP chooses a grandmaster clock using the Best Master Clock Algorithm (BMCA) by determining each clock's quality and accuracy. The grandmaster clock has the highest hierarchical order. Once it is established, the rest of the hierarchy is formed, which builds master and slave relationships. The latter has to synchronize with the master clock. In order to adjust the clock, messages have to be exchanged between master and slave. The master sends a Sync request to the slave and follows it up with a follow-up request. This message contains the timestamp the Sync message was sent. It assures the accuracy of the Sync timestamp since it could have diverged due to processing delays. The slave sends a delay request, which is needed to measure the time from slave to master after the follow-up message arrives. Afterward, the master responds to it with a delay response containing the timestamp when the delay request arrives.

Now, all important timestamps are available, and the offset and delay can be calculated, but two different kinds of delay calculation modes have to be considered. First is the End-to-End (E2E [11]) delay, where the round-trip time between a master and a slave is measured. The other method is measuring the Peer-to-Peer (P2P [12]) delay. This method is used in networks with redundant paths. It measures the link delay between each pair of devices and calculates the delay and offset [4]. After succeeding with the measurements comes the synchronization of the clocks. The delay and offset are calculated and the slave clock is adjusted. The Figure 2 shows a simplified cycle of a PTPv2.1.

It is also important to note, that this protocol differentiates clocks. First, there are transparent clocks which are clocks with a switch. The delay of the switch has to be calculated depending on the state of the switch but the transparent clock does not directly synchronize with other clocks. Instead, it works as a forwarding device with a delay that is added to the timestamp calculation. Second, boundary clocks receive the time in one port and distribute

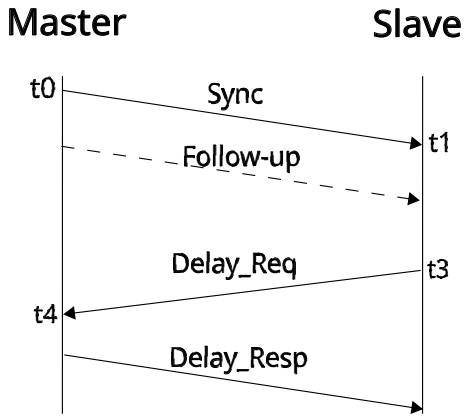

Figure 2: The Precision Time Protocol PTPv2.1 Standard IEEE-1588-2019 [13]

it through another port. They act as a master clock for devices with lower hierarchy therefore reducing the direct connection to the grandmaster. Last but not least are the ordinary clocks. They only have a single PTP Port and act either as a master clock or slave clock [4].

The SPTP is a simplified and advanced version of PTP that maintains compatibility with current equipment that is capable of supporting PTP. It also reduces the number of exchanges between master and slave nodes. As a result, more efficient network communication is possible [5].

3. Assessment of Time Protocols

Even though the concepts of NTP, PTP, and SPTP are similar, they have distinct variations in their performance, accuracy and component utilization. This section compares the previously mentioned protocols and sets their differences side by side.

3.1. NTP vs PTP

Although the network time protocol and the precision time protocol fulfill the same roles, both have significant differences in their implementation. While NTP uses Strata to determine the grandmaster clock and the hierarchical orders of the connected devices, the PTP determines its grandmaster by running the BMCA. NTP's Strata are more easily implemented but PTP's BMCA, while being more complex, finds the most optimal grandmaster clock to send Announce messages to. Figure 3 depicts a network with multiple grandmaster clocks but due to the BMCA only one of them sends out Sync messages while the other is dormant.

Another difference is the introduction of transparent and boundary clocks in the precision time protocol. This differentiation does not exist in the network time protocol resulting in more accuracy for the PTP. This can be explained since NTP is an entirely software-implemented protocol, which is sufficient should ms-level deviations be acceptable. NTP does not require any hardware support and can therefore be used on most networked devices [3] [6]. As a result, the NTP is commonly used for general-purpose time synchronization and is also the most common type of time synchronization protocol available to the

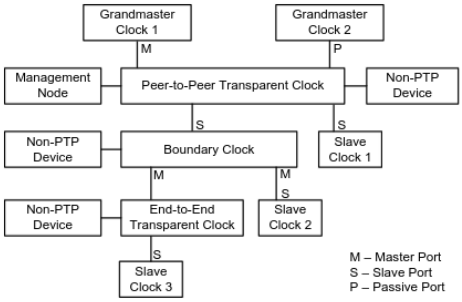

Figure 3: The Precision Time Protocol [4]

public. The network time protocol is also easier to implement since it is less complex. Unlike NTP, two approaches for the implementation of PTP have been accepted [6]. One approach is a precision time protocol with software-implemented time stamping resulting in clock deviations in microseconds μs. The other more accurate approach is hardware-supported PTP. This allows hardware time stamping, resulting in precision within nanoseconds [6]. That is the reason why PTP is the most commonly used where the accurate synchronization of time is crucial, such as in the financial area, telecommunication, and industrial automation. In addition, PTP slaves and masters exchange more messages with each other in order to maintain higher accuracy than NTP while the message size stays similar to NTPs. PTP also has a higher frequency of message exchanges compared to NTP. This can be explained by to the goal of making PTP as accurate as possible. While NTP sends messages in an interval of seconds, PTP exchanges data every few microseconds [3] [6]. As a result of the higher frequency, PTP has a higher processing overhead in addition to a higher network load.

Let t_0, t_1, t_2, t_3 be timestamps for a NTP cycle with t_0 as the sending time of the request, t_1 its arrival time, t_2 the sending time of the response, and t_3 its arrival time. In addition, let $a = t_1 - t_0$ and $b = t_2 - t_3$. The roundtrip delay δ and the clock offset θ of B relative to A at Time T is calculated as followed [3] :

$$\delta = a - b \text{ and } \theta = \frac{a+b}{2}$$

PTP calculates its offset δ and delays θ differently in order to achieve higher accuracy for the clock synchronization. Let t_0, t_1, t_2, t_3 be timestamps for a PTP cycle with t_0 as the sending time of the sync request, t_1 its arrival time, t_2 the sending time of the delay request, and t_3 its arrival time. There are two different kinds of delay in the PTP; therefore, there are two mechanisms to calculate the delay. One is the request-response mechanism for the End-to-End delay and the peer-delay mechanism for the Peer-to-Peer delay. During the following calculations, only the delay through the request-response mechanism without any switches between master and slave is considered. Additionally, CF_M is the sum of switch delays from the master to the slave and CF_S the sum of switch delays from slave to master [14]. The offset and the delay of the slave clock from the master clock are calculated as presented [4]:

$$\delta = (t_1 - t_0) - \theta \tag{1}$$

and

$$\theta = \frac{(t_3 - t_2) + (t_1 - t_0) - CF_M - CF_S}{2} \tag{2}$$

It is apparent that the calculations for the delay and offset are different from each other with PTPs calculation being more accurate by considering different types of delays and using more efficient algorithms.

3.2. PTP vs SPTP

The SPTP is a simplification of PTP and therefore shares many similarities. The typical complete exchange for IEEE 1588-2019 two-step PTPv2 unicast UDP flow is depicted in Figure 4

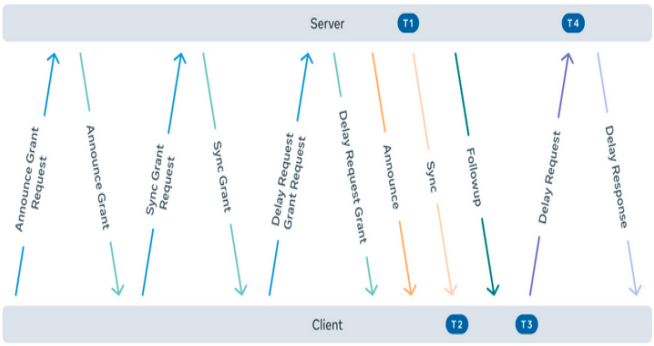

Figure 4: Two-step PTP exchange [5]

This sequence repeats itself and can be extended or reduced depending on the negotiation results between master and slave. Designing the PTP this way allows it to be flexible. The trade-off is that the slave and master have to keep their state in memory, resulting in excessive usage of resources, such as CPU and memory as well as increased code complexity.

SPTP does not need states to be preserved and reduces the number of exchanges needed while still being compatible with a two-step PTPv2 exchange, as seen in Figure 5 [5].

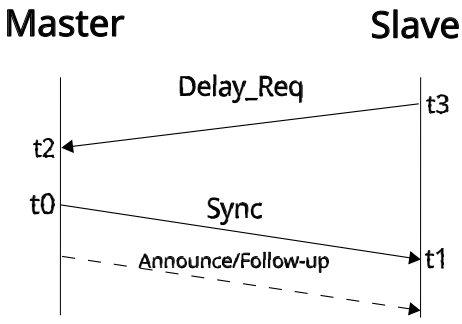

Figure 5: Simple Precision Time Protocol [5]

Unlike a PTP exchange, the SPTP starts with a delay request from the client to the server initializing the variables t_2 and t_3 as well as the correction field CF_S. The Sync message then gets dispatched from the server to the client containing t_0 and CF_M. Afterward, an Announce/Follow-up package is sent with t_2 and other

information such as clock class, accuracy, and so on. With the values of this process, the offset and delay are calculated using equation 1 and 2. As a result, the 11 exchanges in Figure 4 are reduced to 3 in Figure 5 while maintaining compatibility with PTPv2.

In general, SPTP is simpler and easier to deploy while maintaining microsecond-level accuracy, while PTP is more precise but also more complex.

4. Different Clock Synchronization Deployments

Implementing the protocols is a complex matter since many different factors have to be considered, e.g. security concerns, precision and accuracy requirements, scalability, and more. Clock synchronization services are supplied by cloud providers, removing the need to implement a time protocol. This section analyzes Google Public NTP from GCP, Amazon Time Sync provided by AWS, and Alibaba Cloud Time Sync Service.

Google Public NTP is a NTP that uses a 24-hour linear smear from noon to noon UTC as a leap smear in order to maintain system stability during the insertion of leap seconds by gradually adjusting the extra second across the hours before and after each leap [15]. The servers are Stratum 1 which means that the Google servers are referred to as more accurate clocks, such as atomic clocks or GPS clocks. This NTP is also publicly available without the need for an account or cloud service usage with the server address being `time.google.com` or from `time1.google.com` to `time4.google.com` [7].

The Amazon Time Sync service is also a Stratum 1 referencing GPS and atomic clocks but unlike Google Public NTP it is only available to EC2 instances within AWS. This ensures that the service is already integrated with AWS infrastructure. In addition, Amazon Time Sync service uses Leap Smearing for the same reason as Google Public NTP. Even though this service is only available for EC2 instances it is still accessible through the instance metadata at `169.254.169.123` for IPv4 address endpoints and `fd00:ec2::123` for IPv6 [8].

Last but not least, the Alibaba Cloud Time Sync service is just like the other two services, a Stratum 1 referencing atomic and GPS clocks. The leap smear used is a 12-hour smear on either side of the leap second [16]. Alibaba Cloud Time Sync is designed to be used within Alibaba Cloud Environments but can be used publicly. In order to access the server, the address `ntp.aliyun.com` can be used [9].

Another service to be noted is Meta's SPTP [5]. It can be used in a PTP environment due to its compatibility with the precision time protocol. SPTP may need to be used with PTP TLVs (type-length-value [5]) should the system, where the PTP/SPTP is used, require subscriptions and authentications [5].

4.1. Testing AWSs and GCPs Clock Sync Services

The device used for testing has the specifications listed in Table 1. This Listing 6 shows a screenshot of synchronization with Amazon Time Sync services and Google Public NTP for a duration of approximately 2 hours. Here, the table for the output is generated, and the Columns are defined using `chronyc`, which is an interface program used to interact with `chronyd` daemon for monitoring and controlling purposes. The commands used to monitor the output is `chronyc sourcestats -v`.

Specification	Details
OS	Fedora
CPU	Intel(R) Core(TM) i7-6700HQ CPU @ 2.60GHz
RAM	SODIMM DDR4 Synchronous 2133 MHz
GPU	GeForce GTX 960M
Motherboard	ASUS GL552VW
Networking	Intel Wireless 7265

TABLE 1: Hardware Specifications of the Test Device

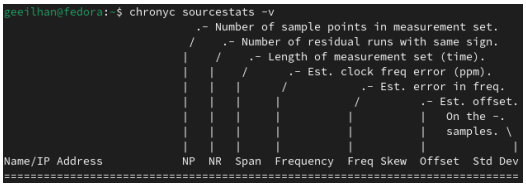

Figure 6: Screenshot of `chronyc sourcestats -v`

Following Listing 7 shows an offset of $\sim$400 μs. This is within the appropriate time of a network time protocol.

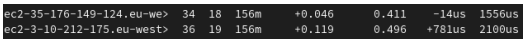

Figure 7: Screenshot of `chronyc sourcestats -v` output for Amazon Time Sync
clock synchronization options

The next Listing 8 shows for `time1.google.com` an offset of 553 μs which is within the NTPs deviation. `time4.google.com` shows an offset of 3303 μs which is significantly bigger than `time1.google.com`s offset. This can be explained to synchronization issues and network latency.

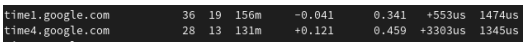

Figure 8: Screenshot of `chronyc sourcestats -v` output for Google Public NTP

This shows that many time sync services provided by AWS and GCP can be used as reliable NTPs. Depending on the environment the clock synchronization service applied should fit the instance it runs on if provided, e.g., Amazon Time Sync service should be used on an AWS instance, which reduces the workload of incorporating the service. If the instance does not provide a time sync service a good fallback service is the Google Public NTP.

5. Conclusion and Future Work

In conclusion, this study has demonstrated that the PTP is a more accurate protocol than a NTP, with the trade-off that the NTP is easier to implement than the PTP. Additionally, the time deviation of NTPs are calculated in microseconds, while the deviations of PTPs is calculated in nanoseconds. Due to its relatively easy implementation compared to PTP, NTP is mostly used for public time sync services.

SPTP is a simplification of PTP that reduces the number of exchanges needed during a full cycle. This leads to more efficient network communication therefore, improvements in resource utilization. The SPTP is also compatible with PTP, with an example of this being Meta implementing a SPTP compatible with almost any PTP environment.

A publicly available time sync service that can be used as a fallback service, for any instance, is the Google Public NTP. Otherwise, if an instance provides a time synchronization service, it should be used, such as the Amazon Time Sync service for AWS instances. Future studies should explore SPTP and its implementations since it is easier to implement the simple precision time protocol than the PTP while its offset is calculated in nanoseconds.

References

[1] F. Bizzarri and X. Wei, "Phase noise analysis of a mechanical autonomous impact oscillator with a mems resonator," in *2011 20th European Conference on Circuit Theory and Design (ECCTD)*. IEEE, 2011, pp. 729–732.

[2] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," in *Concurrency: the Works of Leslie Lamport*, 2019, pp. 179–196.

[3] D. Mills, "Internet time synchronization: the network time protocol," *IEEE Transactions on Communications*, vol. 39, no. 10, pp. 1482–1493, 1991.

[4] S. T. Watt, S. Achanta, H. Abubakari, E. Sagen, Z. Korkmaz, and H. Ahmed, "Understanding and applying precision time protocol," in *2015 Saudi Arabia Smart Grid (SASG)*. IEEE, 2015, pp. 1–7.

[5] O. Obleukhov and A. Byagowi, "Simple precision time protocol at meta," https://engineering.fb.com/2024/02/07/production-engineering/simple-precision-time-protocol-sptp-meta/, February 2024, [Online; accessed 07-June-2024].

[6] Z. Idrees, J. Granados, Y. Sun, S. Latif, L. Gong, Z. Zou, and L. Zheng, "Ieee 1588 for clock synchronization in industrial iot and related applications: A review on contributing technologies, protocols and enhancement methodologies," *IEEE Access*, vol. 8, pp. 155 660–155 678, 2020.

[7] Google, "Configuring Clients," https://developers.google.com/time, 2024, [Online; accessed 09-June-2024].

[8] Amazon, "Set the time for your Linux instance," https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/set-time.html#configure-time-sync, 2024, [Online; accessed 09-June-2024].

[9] Alibaba, "Manage the time synchronization service," https://www.alibabacloud.com/help/en/ecs/user-guide/alibaba-cloud-ntp-server, 2024, [Online; accessed 09-June-2024].

[10] F. Rezabek, M. Helm, T. Leonhardt, and G. Carle, "PTP Security Measures and their Impact on Synchronization Accuracy," in *18th International Conference on Network and Service Management (CNSM 2022)*, Thessaloniki, Greece, November 2022.

[11] Z. Gao, Y. Hua, X. Jin, S. Liu, and L. Tang, "End-to-end delay testing and research on the internet," in *2023 3rd International Symposium on Artificial Intelligence and Intelligent Manufacturing (AIIM)*, 2023, pp. 13–18.

[12] B. Zhang and S. Wang, "An optimization model of load balancing in peer to peer (p2p) network," in *2011 International Conference on Computer Science and Service System (CSSS)*, 2011, pp. 2064–2067.

[13] "Ieee standard for a precision clock synchronization protocol for networked measurement and control systems," *IEEE Std 1588-2019 (Revision of IEEE Std 1588-2008)*, pp. 1–499, 2020.

[14] O. Obleukhov and A. Byagowi, "How precision time protocol is being deployed at meta," https://engineering.fb.com/2022/11/21/production-engineering/precision-time-protocol-at-meta/, November 2022, [Online; accessed 07-June-2024].

[15] Google, "Leap Smear," 2023, [Online; accessed 09-June-2024].

[16] S. Moss, "A second look," https://www.datacenterdynamics.com/en/analysis/a-second-look/, November 2022, [Online; accessed 09-June-2024].