Masters of Time: An Overview of the NTP Ecosystem

Teemu Rytilahti, Dennis Tatang, Janosch Köpper, and Thorsten Holz
Ruhr-University Bochum, Germany

Abstract—**The Network Time Protocol (NTP) is currently the most commonly used approach to keeping the clocks of computing devices accurate. It operates in the background of many systems; however, it is often important because if NTP fails in providing the correct time, multiple applications such as security protocols like TLS can fail. Despite its crucial practical role, only a limited number of measurement studies have focused on the NTP ecosystem.**

In this paper, we report the results of an in-depth longitudinal study of the services provided by the *NTP Pool Project*, which enables volunteers to offer their NTP services to other Internet users in a straightforward manner. We supplement these observations with an analysis of other readily available NTP servers, such as those offered by OS vendors or those that can be freely found on the Internet. The analysis indicates a reliance on a small set of servers that are (at least indirectly) responsible for providing the time for the Internet. Furthermore, this paper considers the impact of several incidents that the authors observed between December 2016 and April 2017.

To complement this study, we also perform an analysis of multiple geographical regions from the operator's perspective, spanning a period of 5 months. A coarse-grained categorization of client requests allows us to categorize 95 percent of our incoming traffic as NTP- and SNTP-like traffic (the latter being a simpler, but more error-prone, form of NTP); we observe that up to 75 percent of all requests originated from SNTP-like clients. With this in mind, we consider what kind of harm a rogue server administrator could cause to users.

1. Introduction

The availability of correct time information is crucial in many scenarios (e. g., PKI, authentication, cryptocurrencies, and much more) and affects many facets of contemporary interconnected services, ranging from network and security protocols to attestation and authorization services. As ubiquitous networking strategies exist in the era of the Internet of Things (IoT), an accurate clock is also required by many embedded systems. In practice, there are also many services with highly time-dependent applications (e.g., financial services and process controlling) that use highly accurate clocks. However, many scenarios do not require this kind of precision. Hence, instead of building expensive clocks into these devices, manufacturers are taking advantage of the ability to synchronize clocks over the Internet. In practice, the Network Time Protocol (NTP) is the standard protocol used to synchronize clocks. Considering the ubiquitousness of NTP, we believe that it represents one of the Internet's core protocols and that it has been overlooked so far.

Recent research in this area has focused on the security features of this protocol. The discussion was prompted by the Distributed Denial-of-Service (DDoS) reflection attacks that took place in 2014 [1], [2]. In 2016, Malhotra et al. [3] examined the functionality of NTP and discussed security-related attacks against the protocol itself. However, not much attention has been paid to the infrastructure of the NTP ecosystem. The questions that arise are, among others, (i) who provides these services, (ii) how wide-spread is their usage, and (iii) how interconnected is the entire infrastructure? The most recent study on the entire server ecosystem is that of Minar et al. [4] published in 1999; we thus believe that it is time to revisit this topic and review what is known about the NTP infrastructure in general.

In this paper, we explore the time-synchronization ecosystem from multiple viewpoints, without overlooking its potential vulnerability to determined attackers. We provide an overview of the current NTP infrastructure, ranging from vendor-specific synchronization servers to voluntarily maintained servers, such as those offered by the *NTP Pool Project*. Beyond analyzing the server infrastructure, we also analyze the client side from the perspective of server operators, with the help of multiple NTP servers deployed in various geographical regions. From the collected data, we categorize different client implementations as NTP- or SNTP-like, with the goal of understanding how susceptible these clients would be to malicious service providers. Finally, we discuss security implications of our observations.

We see our work as a call to raise awareness of NTP's importance, and we hope that this study will raise interest and foster research on this topic.

In summary, we make the following four contributions:

- We provide a comprehensive overview of the current NTP ecosystem from both a server and client perspective. We use active probing on most well-known NTP servers and complement this approach with information obtained by means of Internet-wide network scans;
- We deploy multiple NTP servers in various geographical regions to perform an client analysis which allows us to categorize our incoming traffic as NTP- and SNTP-like traffic;
- We present several case studies of unusual events we observed during our study, with examples including

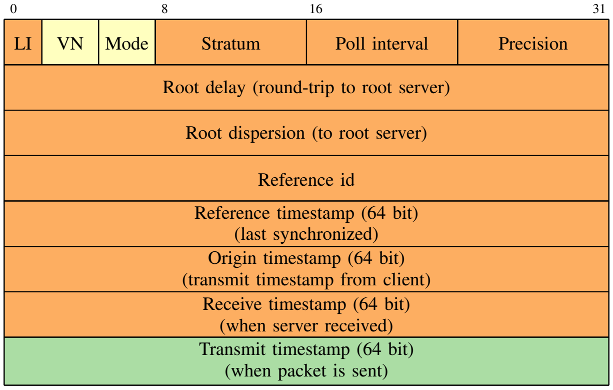
0
8
16
31
LI
VN
Mode
Stratum
Poll interval
Precision
Root delay (round-trip to root server)
Root dispersion (to root server)
Reference id
Reference timestamp (64 bit)
(last synchronized)
Origin timestamp (64 bit)
(transmit timestamp from client)
Receive timestamp (64 bit)
(when server received)
Transmit timestamp (64 bit)
(when packet is sent)

Figure 1. NTP payload

the leap second that occurred recently and the failure of several NTP servers deployed by Microsoft; and

- We release various data sets we collected for these analyses to foster research in this area.

2. Network Time Protocol

The Network Time Protocol (NTP) is a development of time services offered in the past. Its first version, RFC 958, was introduced by David Mills in 1985 [5], while RFC 5905 represents its newest (version 4) incarnation [6]. NTP uses UDP datagrams for communication and supports a variety of operating modes, of which client/server is the most common form in practice. Note that, for the protocol to function correctly, it is necessary that multiple servers are contacted (to be precise, at least three) in order to obtain a majority vote regarding the correct time.

2.1. NTP Packet Header

The essential communication for synchronizing time between two systems using NTP is to repeat two message exchanges periodically. The client sends an NTP request, and the server responds, using the same packet format, as shown in Figure 1 [6]. The data within the response message includes all of the timing information for the client. The server sets almost all fields (orange), except the transmit timestamp, which is set by the client (green), and the `VN` (version) and `Mode` field, which are set by both (yellow).

The first three fields include the leap indicator, the NTP version number, and the mode. The leap indicator field is used to inform of an impending leap second. A leap second is a one-second adjustment intended to keep the time closer to the mean solar time; this correction is necessary due to irregularities in the Earth's rotation speed. NTP offers eight different modes: The values '3' and '4' are client and server messages, respectively. The following stratum value indicates the distance of the server from its physical clock source: e. g., a server using a stratum 1 server as its synchronization server will set its stratum to '2', and so on. A client will usually send a '0' as its stratum value if it is not also operating as a server. Figure 2 displays the hierarchical construction of the NTP infrastructure with the aid of the stratum values. One might think that the stratum values could also represent a metric of reliability. However, this is not the complete truth—these values are simply used to indicate the distance of the server from its original clock source [6]. Stratum '16' and, in some cases, '0' indicate that the server is not in sync and is therefore unable to fulfill its purpose. Values greater than '16' are reserved and should not be used.

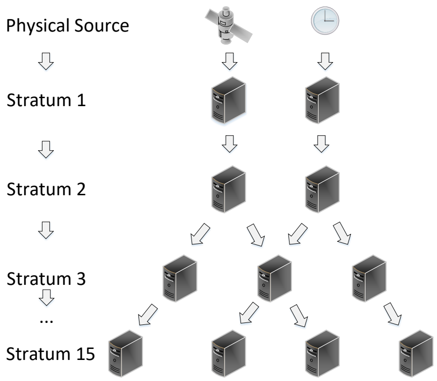

Figure 2. Hierarchical structure of NTP

The poll and precision fields are integer values that represent the maximum interval between sequenced messages (poll) or the accuracy of the system clock (precision). The root delay represents the total round-trip delay to the reference clock. The root dispersion field reflects the total dispersion to the reference source at the root of the synchronization subnet.

The reference ID (`refid`) is significant because it is used to avoid cyclical formations in the network, which occur when two servers use each other as their upstream server. In practice, it is selected from the clock source that is currently being used: for IPv4 upstreams the refid is the address of the upstreams, whereas for IPv6 upstream, the IP address is hashed with MD5, of which the first four octets are used as refid [6]. This is an important feature because it allows us to induce connections between servers.

There are two stratum-dependent special cases in terms of refid handling. First, a particular instance is stratum 1 (which indicates that the server is being synchronized directly from a hardware clock source), in which case the reference identifier contains an ASCII-encoded string, such as Global Positioning System (GPS), DCF77 (a clock signal transmitted via radio waves), and so forth [6]. The second special case for refid is providing an in-band signaling mechanism for informing clients without the need to change the protocol in order to cope with problematic clients [7].

In addition to the header fields, the NTP payload includes four different timestamps, three of which (namely, origin, receive, and transmit) are used to calculate the offset, the message round trip time, and a single reference timestamp. Since this paper focuses on a structural analysis of the NTP infrastructure, we omit further details here.

2.2. Simple Network Time Protocol

The Simple Network Time Protocol (SNTP, defined in RFC 4330 [8]) is a simplified version of NTP and functions in the same manner. This has the result that SNTP clients can synchronize with any NTP server, and an NTP server cannot determine whether a request originates from an NTP or SNTP client [8]. The main difference is the time setting itself: NTP uses algorithms that are intended to maintain a highly accurate time. For this reason, multiple time servers are consulted and checked for accuracy. NTP adjusts the system clock using small, skewed adjustments in order to ensure seamless time correction; in order to make time changes, the clock is sped up or slowed down slightly. In contrast, SNTP usually uses a simpler approach: e. g., some implementations utilize time jumps to adjust their clocks. This simple protocol functions for most applications where a timestamp is needed, and it is common to use only a single time server when doing so.

3. NTP Server Ecosystem

In order to understand the NTP ecosystem, we employ two different approaches: First, we explore results of *Internet-wide network scans* to complement our subsequent crawling effort with information about the servers at large. Second, we perform *active probing* on a list of domains known to belong to established, widely used NTP servers such as those of the NTP Pool Project and those of vendors such as Apple and Microsoft.

Although our primary focus lies on the data gathered through active probing, we begin by exploring the NTP server infrastructure. Thereafter, we move on to analyze the results of the active probing.

3.1. Internet-wide Network Scans

The first two data sets used in this section are adapted from those employed by Kührer et al. [1]. The data sets consist of responses for the NTP's `monlist` and `version` commands, from 2013 to the end of 2016. These make it possible to analyze the historical structure of the network, the changes that have occurred to it in the wake of `monlist` misuse (which has been, and is still, used for amplification attacks [9]), and the network's structure today.

The `monlist` command, which is an administrative command that returns information concerning a server's recent clients, has been used widely for reflective DDoS attacks. The `version` command is likewise an administrator command, that queries servers regarding their status.

Because those two data sets are based on the responses for non-conventional NTP features (which are not always implemented or disabled) and we wished to understand the populations of all of the NTP servers, we modified the scanner used to obtain the previous data sets to send regular NTP requests. This was done in order to incite responses from all of the servers involved.

This data set was created solely for the purposes of this paper; it consists of two separate scans of the IPv4 address-space. The first scan was performed on the 13th of January, 2017 and the second exactly a week after the first. Therefore, our analysis is based on three separate data sets, each of which addresses the ecosystem from a different angle:

1) Regular time synchronization (client mode) responses, which help us to identify a lower bound of all of the available NTP servers;
2) The NTP ”version” responses, which can potentially reveal information concerning the servers; and
3) The NTP ”monlist” responses, which have been, and are still being, used for amplification attacks.

3.1.1. Client Mode Requests. We begin our study by gathering information concerning all of the NTP servers available on the Internet. This is done by means of performing Internet-wide scans with a regular NTP client mode payload against the entire IPv4 address space.

These requests are used by regular NTP clients to obtain the time from servers, who respond using server mode responses. To ensure the widest coverage, our scans used a version value of '1' for the NTP packets (shown as `VN` in Figure 1), as this is the most widely accepted version in the server implementations that we investigated.

To minimize the possibility of ill-chosen time, we decided to perform two scans, one on the 13th of January, 2017, and the second a week thereafter. The first scan identified 9,463,068 unique IP addresses, while the second produced 9,450,138; the number of intersecting servers was thus 8,112,486. Moreover, we removed all servers that responded with either stratum '0' or '16', as these servers were not properly synchronized; hence, they do not represent actual servers that can be used by others. This subtraction leads to a data set of 6,988,532 available servers, which we use as a lower bound of all of the NTP servers available on the Internet.

3.1.2. Version Requests. Our second data set consists of control mode requests (`mode 6`). This data set was obtained by leveraging the `READVAR` feature, which allows us to query servers regarding their associations; a special case here is the ability to query a server's own status information.

These status responses are partially standardized; they can also contain variable-length data that provide a variety of information concerning the status of the server. In this section, we use the `version` and `system` fields as used in previous works [1], [2], which, after some normalization, make it possible to perform a coarse-grained categorization of the different types of NTP servers. For a complete listing of the information available in these responses, we refer the reader to the `ntpq` manual [10].

As shown in Figure 3, immediately before the outbreak of `monlist` abuse (see Section 3.1.3), the number of responding servers was almost seven million. After the issue became prevalent and widely reported, e. g., with the release of CVE-2013-5211 [11], system administrators began to patch their systems, and a drop of almost two million servers can bee seen, which is followed by a much larger drop later. For the sake of intelligibility, the ntpd versions are aggregated by their minor version.

What remains impressive is that, even though the number of responsive servers has slowly declined, the total number of servers has been roughly steady, with small changes from March 2014 to the present. A reason for this could be its unsuitability for conducting DoS attacks due to its smaller response sizes.

To reflect this in our previous results regarding NTP client requests, about 30 % of the publicly visible NTP infrastructure still seems to support these requests, a fact that may have been overlooked up to now, even after the potential of attacks has been identified e. g., by Malhotra et al. [3], [12].

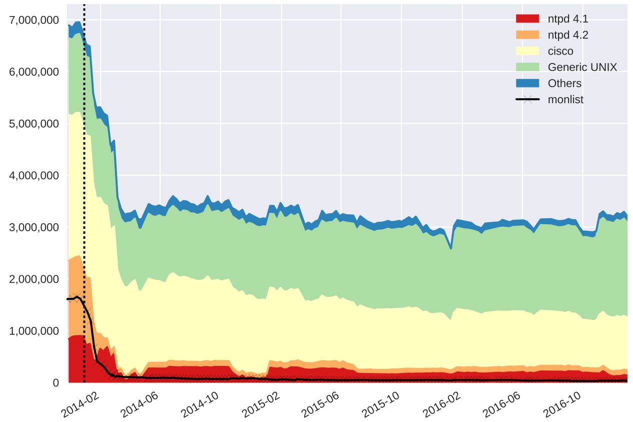

Figure 3. Aggregated server identification

3.1.3. `Monlist` Requests. Our third data set consists of responses to `monlist` (`mode 7`) requests. The black line in Figure 3 indicates the change in number of monlist amplifiers, starting from more than 1.5 million and, at the end of the data set, declining to approximately 30,000 servers. While the number of vulnerable servers has been declining, according to Akamai [9], they still represent attack vectors. As the monlist incident has already been addressed by other authors (e. g., Kührer et al. [1] and a thorough investigation by Czyz et al. [2]), we do not go into details here.

However, we wish to emphasize one particular finding, namely that which confirms the suspicion of Rytilahti et al. [13] regarding dormant amplifiers. We came to this conclusion as a result of the fact that our regular NTP client request scans (described in Section 3.1.1) were visible in the monlist responses of almost 18,000 servers (i. e., over 50 % of responding servers), which, however, did not display any evidence of them being used for amplification attacks. That is, the responses from these servers had only limited numbers of clients, and none of them were recorded as using "monlist" requests but only client and server modes.

According to Akamai's "State of the Internet" report from the fourth quarter of 2016, they recorded attacks from up to 300,000 [9] amplifiers. This report reveals a significant discrepancy between what our data set shows and the information provided by the Shadowserver project [14]. Therefore, we conclude that a population of dormant amplifiers seems to exist.

3.2. Active Probing

In the previous section, we analyzed the visible NTP infrastructure, but this alone does not provide much information regarding those servers that are actively used in order to synchronize time. Therefore, we now introduce our main contribution. It takes the form of active probing, in which we constantly query well-known service providers in order to obtain a better understanding of their services. To this end, we categorize the servers into three groups:

1) The NTP pool [15], which simplifies the process of accessing volunteer-provided verified NTP servers and whose servers are widely used by open-source operating systems and various appliances;
2) The NTP.org server lists, which are manually administrated lists of servers for users to pick from; and
3) Servers offered by various vendors, such as Apple and Microsoft, which are used by the users of their products by default.

We start by explaining the functionality of our crawler; thereafter, we move on to discuss these groups separately and explain why we chose them for this study. In order to understand the NTP infrastructure and the need for active probing, it is crucial to understand how the NTP Pool Project functions. Therefore, we first describe it in more detail, with the two other groups following.

3.2.1. Crawling & Probing the Servers. For the active probing, we developed a crawler, running on a single server to request `A` and `AAAA` records for all of the collected domains directly after their Time-To-Live (TTL) values expired. In order to avoid potential data loss, the DNS requests were made using TCP transport. For each of the returned IP addresses, we also requested the time using the `ntpdate` utility, but, to minimize the burden on servers that appeared often, we requested time only after one hour had passed. In addition, to track the servers that were not visible in subsequent DNS responses—which occurred often within the pool—we also made an NTP request on every seen IP address every 6 hours. When IPv6 addresses were returned, we leveraged a tunnel provided by Hurricane Electric [16] to make the necessary requests. All of the responses regarding both name resolution and NTP communication were saved in PCAP files for later analysis.

3.2.2. The NTP Pool Project. In the past, it was customary to find servers for time-synchronization purposes using manually maintained address lists (such as those mentioned in Section 3.2.3) and entering those into the configuration file of the NTP client.

Fortunately, since 2003, there has been a more scalable and reliable way of doing so provided by the NTP Pool Project [15] (which, according to its website, was created to solve the problem of the large-scale [mis]use of a small amount of servers). Instead of requiring users to find and define a static list of servers, the project relies on DNS to provide servers to users according to their locations; it also handles load-balancing in order to avoid overloading a small set of servers.

Today, the NTP Pool Project represents the largest effort to centralize access to NTP services, with almost 4,000 servers in 92 countries. Its services are used, e. g., by most, if not all, open-source operating system distributions.

Although the NTP Pool Project is particularly used by open-source operating systems, it is also utilized by various well-known router vendors, including Linksys, TPLink, Asus, and Zyxel. We empirically verified this fact with the aid of Firmadyne's [17] firmware crawler. Furthermore, our empirical exploration indicates that the NTP pool is also used in the realms of home automation systems, security cameras, and household appliances. One recent example is the furniture retailer IKEA, which uses the pool for their new line of smart home appliances [18]. Park et al. [19] also report that Android-based devices use the pool on specific circumstances where they are unable to obtain the time from the phone provider's network.

Structure of Pool Domains – The pool is structured into *zones*, based on continent (e. g., asia.pool.ntp.org or europe.pool.ntp.org) and country (e. g., us.pool.ntp.org or cn.pool.ntp.org), thus providing a more localized service for users. However, thanks to the custom DNS server delivering results based on the geolocation of the client's IP address, it is no longer necessary to manually configure these domains into NTP clients. It has to be noted, however, that, in some cases (such as when an adequate number of servers does not exist in a country), it may be beneficial to do so, as we shall see in Section 4.3.

In addition to these location-based pools, there also exist zones for vendors such as Debian, OpenWrt, and Linksys [20]. These can be used by the pool administrators to identify problematic clients, which has happened before [20]. According to our empirical analysis, the vendor zones simply mirror the functionality of the main pool, using a zone based on the client's geolocation.

Due to the necessity of contacting multiple servers against false chimers, as discussed in Section 2, and since clients tend to arbitrarily choose only one IP address with which to connect to DNS answers (although the pool returns up to four IP addresses), historically, four separate domains that resolve to a different set of IP addresses have been used. For this reason, the zones are further divided into four *subpools*, indexed from '0' to '3' (e. g., 0.us.pool.ntp.org or 2.cn.pool.ntp.org, where only '2' offers requests to IPv6 'AAAA' requests).

Working Principles – The pool consists of two loosely coupled separate systems: the administration and controlling system and the DNS server. The pool operates on an interface [21] that allows users to add their servers to the pool and to adjust a number of settings, such as bandwidth allocation. This part of the system is also responsible for monitoring the state of the servers (e. g., controlling the correctness of time given out by the servers and handling load balancing between servers) and the creation of DNS zone files to be fed into a geolocation-sensitive DNS server [22].

In order to track the correctness of the participating servers, a control server regularly requests the time from them. If the control server deems a server's response as deviating excessively from its time, or, even worse, if a server does not respond at all, it will eventually be removed from circulation.

It is essential to understand the dynamic nature of the pool, which is the result of its voluntary-based nature. It is important to bear in mind not only the DNS-based load-balancing with short TTLs that it performs but also its geolocation-based responses. It is not possible to understand the NTP ecosystem without actively checking the respective domains in order to collect information concerning the servers. To this end, we developed a crawler, which was previously described in Section 3.2.1.

For the crawler, we generated a list of all of the domains for possible zones based on continents, countries, and the 31 vendor pools that we were able to find empirically. The crawler was initialized with a total of 289 pool zones, including all of their sub-pools.

3.2.3. ntp.org lists. As mentioned previously, before the development of the NTP Pool Project, it was necessary to identify servers to use manually. Beyond being more cumbersome than simply directing the NTP client to use a set of domains, this approach also required the user to rely on the longevity of the servers chosen.

To that end, there exist manually maintained lists of public NTP servers (separate lists exist for stratum 1 and 2), such as those located in the `ntp.org` (hence the name of the group) [23]. These two lists identify servers that are recommended for use as stable, reliable upstreams when setting up a new server. In November 2016, we took a snapshot of their contents aggregated in a summary listing [24] and added them to our probing system.

3.2.4. Vendor Servers. The last group consists of the servers provided by various vendors which can be considered as being of good standing and providing the correct time to their clients. In this category, we include the servers of Alibaba (Aliyun, `time[4-6] .aliyun.com`, which we did not explicitly add to our crawler but encountered in the pool; these servers deserve separate handling, as we will see later), Apple (`time{-ios}.apple.com`), Canonical (`ntp.ubuntu`

.com), Google (time{[1-4]}.google.com), and Microsoft (time.windows.com).

In addition to these vendor-specific servers, we also added the NTP servers from National Institute of Standards and Technology (NIST) [25] to this group, as they are very widely used and considered a reliable time source [26].

4. Understanding the NTP Infrastructure

Using the data sets described previously, we now analyze the networked structure of the NTP ecosystem. Considering the connected nature of the servers and the availability of the upstream information provided with the help of the refid, we decided to leverage a graph for the analyses.

This chapter is structurally similar to Chapter 3: We discuss our findings based on the different server groups and, thereafter, briefly discuss two events that we noted during the observation period.

4.1. NTP Infrastructure Graph

In order to analyze the networked NTP infrastructure, we developed a graph based upon our previously collected network traces, which consist of DNS and NTP replies. We defined the following nodes: *Domain*, *Server*, *Stratum*, *Hardware*, *Land*, *AS* (for ”autonomous systems”), and *NetScan* (for network scans). Figure 4 depicts the concept of this graph, including the relations between the elements.

We used Cymru’s [27] IP-to-ASN mapping to create *AS* nodes for the servers found by means of active probing, whereas pyasn [28] was used to perform the same task on network scan data. For geolocation, we leveraged MaxMind’s GeoLite2 City database [29]. For pool domains, we merged all of the sub-pools (e. g., 2.us.pool.ntp.org) into one node (us.pool.ntp.org), with the assumption of fair distribution between them. Where relevant, we also store a counter, e. g., *seen_in* edges for how many times the server has been seen in the domain and *seen_in_scan* for how many servers were seen as its downstream during the scan.

Returning to Section 3.1.1, in which we discussed the Internet-wide scans. The information provided by these scans is not directly inserted into the graph; rather, we add a node for both scans. In addition, we add edges between the seen upstream servers, with a counter indicating the number of downstreams.

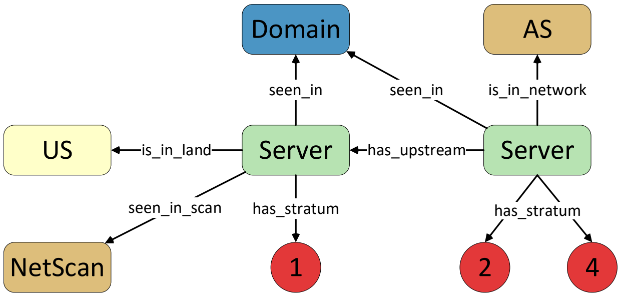

Figure 4. Simplified extract from our graph

Completeness of our graph – Although we performed initial data collection during the last quarter of 2016, in this paper we concentrate on a more complete data set gathered between the 20th of January and the 17th of April 2017, a period spanning almost three months. Some domains with large numbers of IP addresses did not receive new IP addresses after the beginning of March; however, this fact neither has an effect on our analyses, as they are based on relative amounts, nor on the NTP responses, as they were collected from servers that were already known.

From total of 4,928 servers belonging to the actively crawled groups in the graph, only 6 had no path to stratum 1 through upstream edges. It shall also be noted there is an intersection between the groups of servers—1141 servers from the NTP.org list and 13 servers from the vendor group are also in the pool.

4.2. Internet-wide Servers

Considering our interest in determining the importance of the servers selected for active probing, we start by analyzing the Internet-wide scans described in Chapter 3.1.1. The results in this section are based on the intersection of those two scans. It is important to note that only the upstream information was connected to the graph; the rest of the analyses were performed separately.

Autonomous systems and countries – There were servers from a total of 35,196 autonomous systems in the data set, with the median number of servers per system being seven. In total, 99 % of ASes had fewer than 3,000 NTP servers, and a mere 101 ASes had more than 10,000 servers (see Table 1 for a summary).

According to Maxmind’s database, the servers from both netscans were located in 238 different countries; however, almost 25 % of all of the responsive servers were in the USA, followed by 16.6 % in China.

Interconnectivity – In total there were 92,464 unique upstreams from which 84 % (or 77,510) is the intersection from both scans. 47 % of those upstreams were private IP addresses, which leaves us with 48,095 seemingly valid upstream servers we are using for the following analysis. Due to the characteristics of the refids they may not always point to a working IP addresses, however, for our analyses this is negligible.

Almost 2 million servers (24.2 %) used one of the 2,621 upstreams located in the pool, compared to $\sim$ 700,000 servers (7.4 %) that used upstreams from the vendor group and $\sim$ 1 million servers from the ntp.org list group. Furthermore, the median number of downstreams of those pool servers was 113 whereas the median for non-pool servers was merely 3.5. These facts highlight that the servers we chose for active probing are important on the global scale, as almost every fourth upstream was reported to be in our active probing data set.

Stratums and hardware – Comparing the stratum distribution (depicted in Table 3) to our actively probed server groups shows that these servers are proportionally further away from stratum 1 than those of our other groups.

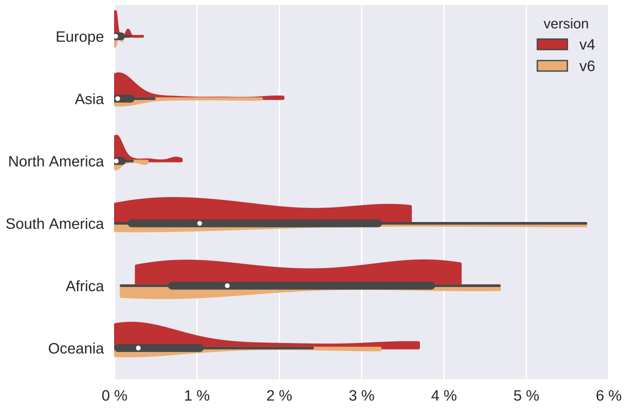

Figure 5. Probability of a server being seen in the continental zones

Nevertheless, 74,000 servers (or 0.8 % of all servers) had a hardware upstream, while almost 1.3 million used upstreams from the reserved RFC 1918 address space.

4.3. NTP Pool Servers

We begin the analysis of the NTP Pool servers by investigating if the pool favors particular servers. As can be seen from the Pool Project's official statistics, the distribution of servers is skewed towards Europe, followed by North America (mainly the USA), leaving the rest of the pools relatively sparsely filled. At the end of our observation period, the European pool was by far the largest, with 2,785 servers according to the pool's official statistics; this was followed by 929 in North America, 278 in Asia, 92 in Oceania, and a mere 44 and 25 in South America and Africa.

Comparing these statistics to those in our graph, we identified a total of 4,564 servers in the pool, of which 2,331 were located in Europe (83.7 % of the number identified in the statistics) and 853 in North America (91.8 %). This indicates that not all of the servers in the pool system were given out during the observation period. Furthermore, in Asia, we identified a total of 386 servers, indicating a high turnover rate (i. e., the rate at which servers are added and removed) in that continental pool.

Of the total of 289 NTP pool zones that we crawled, 155 (53 %) were empty during the entire duration of this study. Furthermore, 30 pools had a total of fewer than four servers, averaging to 2.4 servers per pool. The median number of servers in all of the populated pools was 86.

As our results for the global pool (`pool.ntp.org`) and `de.pool.ntp.org` contained similar numbers of servers, we conclude that the location-based DNS resolving functions as intended.

With the help of the *seen_in* edges counter, it is possible to determine the relative period of time that a single server has been visible in a specific zone. To obtain a global overview of how fairly the pool system divides the load, we concentrated on the continental pools (Africa, Asia, Europe, North America, Oceania, and South America). Figure 5 depicts the distribution of servers' appearing frequency in all continental zones as a violin plot, highlighting the fact that more populous zones are more evenly distributed when it comes to appearance frequency. As a case in point, our server in Germany was visible only twenty times (or 0.008 %) in the global pool (totaling fifty minutes in DNS responses per 150s TTL, cf. Table 5), whereas our Singaporean server received more attention, showing up in 2.3 % of all DNS responses under `sg.pool.ntp.org`, adding up to over nine full days in the pool. These results seem to confirm that a server in a sparsely populated zone will receive longer periods of time in DNS and that the bandwidth allocation provided when registering the server also has an effect.

Reverse DNS – Approximately 82 % of the IP addresses (3,755 of 4,564) seen in the pool had a reverse DNS record, of which almost 18 percent (or 670) had either `ntp`, `time`, or `clock`, revealing themselves to be NTP servers. When also including domains that started with common nameserver (e. g., `ns1`) or e-mail server prefixes, the number increased to 26.8 %. The addresses without reverses were distributed uniformly: Only nine autonomous systems had more than ten IP addresses without one, and all of them were cloud service providers. Although this may seem unimportant, having an informative reverse can be useful when attempting to account for irregular, but benign, traffic towards the IP addresses returned by the pool.

Autonomous systems and countries – Having investigated the structure of the pool, we move on to analyze the servers and their properties, beginning with their autonomous systems and countries in which they are hosted. Of a total of 1,206 ASes, the top 10 autonomous systems are responsible for almost 30 percent of all pool servers (the top five are shown in Table 1). Furthermore, 99 % of the networks had fewer than 34 servers, with the mean being 3.85 and median 2.0 servers per network. It should be noted here that the total number of ASes differs from that of lands, as some IP addresses are considered by Cymru to exist in multiple autonomous systems.

The pool servers were located in 91 different countries, with the median number being 10 servers per country and the mean 50. The distribution of servers, along with their geolocation, is shown in Table 2.

Interconnectivity of servers – Next, we explore the importance of crawled servers, using their direct downstream counts as a criterion, with our assumption being that the more important servers have a greater number of downstream servers. As can be seen in Figure 6, slightly over 1 % of the pool servers had more than 25 downstreams, indicating an imbalanced network. All ten mostly referenced servers had more than a hundred downstreams in our actively probed data set, the largest number being 445 (almost 10 % of all crawled servers directly, more than 1,000 *indirectly* through hierarchy).

Generally speaking, 75 % of servers had a mere two downstreams, while 95 % had only 10, with the mean number being 2.6. Even when each server has multiple upstreams, which we do not account for here, this clearly

Table 1. Top Autonomous Systems

Pool		ntp.org		Vendors		Internet-wide	
AS	Servers	AS	Servers	AS	Servers	AS	Servers
Hetzner (24940)	304 (6.5 %)	Hetzner (24940)	286 (19.4 %)	Apple (6185)	27 (39.7 %)	Chinanet (4134)	385,287 (5.5 %)
OVH (16276)	273 (5.9 %)	netcup (197540)	55 (3.7 %)	Microsoft (8075)	9 (13.2 %)	China169 (4837)	207,154 (3.0 %)
Linode (63949)	188 (4.0 %)	DFN (680)	42 (2.8 %)	Google (15169)	8 (11.8 %)	Alibaba (37963)	174,870 (2.5 %)
Hurricane (6939)	138 (3.0 %)	Strato (6724)	35 (2.4 %)	Canonical (41231)	6 (8.8 %)	OVH (16276)	105,269 (1.5 %)
Digital Ocean (14061)	94 (2.0 %)	Linode (63949)	35 (2.4 %)	ICST (49)	5 (7.4 %)	Hetzner (24940)	95,500 (1.4 %)
Rest	3,650 (78.5 %)	Rest	1,021 (69.3 %)	Rest	13 (19.1 %)	Rest	6,020,452 (86.1 %)
Total	4,647 (100.0 %)	Total	1,474 (100.0 %)	Total	68 (100.0 %)	Total	6,988,532 (100.0 %)

Table 2. Top Countries

Pool		ntp.org		Vendors		Internet-wide	
Country	Servers	Country	Servers	Country	Servers	Country	Servers
US	865 (19.0 %)	DE	729 (50.2 %)	US	36 (52.9 %)	US	1,731,599 (24.8 %)
DE	781 (17.1 %)	US	205 (14.1 %)	GB	11 (16.2 %)	CN	1,162,970 (16.6 %)
FR	420 (9.2 %)	FR	51 (3.5 %)	NL	5 (7.4 %)	RU	406,941 (5.8 %)
GB	300 (6.6 %)	CZ	42 (2.9 %)	CN	4 (5.9 %)	DE	383,075 (5.5 %)
NL	255 (5.6 %)	GB	36 (2.5 %)	DE	4 (5.9 %)	FR	269,222 (3.9 %)
Rest	1,943 (42.6 %)	Rest	388 (26.8 %)	Rest	0 (0.0 %)	Rest	3,034,725 (43.4 %)
Total	4,564 (100.0 %)	Total	1,451 (100.0 %)	Total	68 (100.0 %)	Total	6,988,532 (100.0 %)

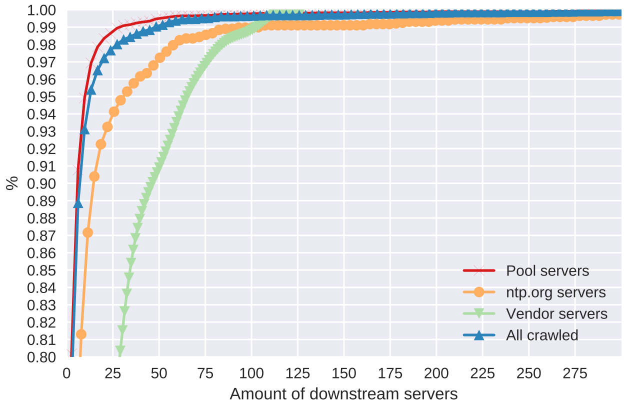

Figure 6. CDF of direct downstreams per group

indicates that a small subset of servers is being favored over the others. Considering the number of stratum 1 and stratum 2 servers that could be chosen as upstreams, as shown in Table 3, distributing the load could prove beneficial. On the other hand, the median upstream number of servers in the pool was three.

Stratums – The stratum value provided by the servers makes it possible to obtain an overview of a network's structure, considering that a server's stratum is its upstream's stratum plus one.

Table 3 depicts the distribution of server stratum values (ignoring the unsynchronized '0'), using the lowest stratum ever returned by the server. As can be seen, 10 % of the pool servers indicated their stratum as having been 1 at some point during the experiment, followed by a clear majority of stratum 2 servers, at almost 75 %. The second percentage in the table indicates the stability of servers, measured by responding using a single stratum value at all times; in this case, a total of 457 stratum 1 servers, of which 277 servers were also such during the entire course of the experiment, can be seen.

Stratum 1 and Hardware – The special case for a stratum 1 server informs the clients not about its upstream but about the reference clock it is using. In the pool, 202 (4.4 % of all servers) stratum 1 servers indicated that they used GPS, followed by almost the same number of PPS-based servers (187, 4.1 %). PPS (pulse-per-second), simply indicates that the server uses those signals to make its clock more accurate; it does not reveal what type of clock source it uses. However, considering that the other types of hardware upstreams were seldom seen— in the pool, there were 20 servers that used DCF77 and 14 that used CDMA (cell phone towers)— combined with the wide availability of GPS devices, it could be hypothesized that at least a large number of those are based on GPS time.

4.4. Other Service Providers

4.4.1. ntp.org lists. The ntp.org-listed servers represent a clear subset of all of the servers in the pool, especially in terms of their dependency on the same set of upstreams. In this group, 1,451 servers were shared among only 388 autonomous systems (compared to over 1,200 for the pool). However, Figure 6 depicts the preferred treatment of these servers when compared to the pool servers in general, which is in line with our other findings. As can be seen in Table 3, these servers have slightly lower stratum values (with 17 % of them being stratum 1) and seem to be more stable when it comes to remaining on the same stratum level. On the hardware side, these servers mostly used PPS (87 servers, or 5.2 %), followed by 82 that used GPS.

Table 3. Stratum Distribution (first % value from total, second % for stable servers)

	Pool	ntp.org	Vendor	Internet-wide
1	457 (10.0 %, 60.6 %)	247 (17.0 %, 66.4 %)	41 (60.0 %, 92.6 %)	66,388 (0.9 %, 98.9 %)
2	3,358 (73.6 %, 64.0 %)	1,049 (72.3 %, 68.8 %)	27 (40.0 %, 81.5 %)	1,525,039 (21.8 %, 94.1 %)
3	689 (15.1 %, 69.4 %)	153 (10.5 %, 79.7 %)	–	3,425,268 (49.0 %, 97.1 %)
4	56 (1.2 %, 82.1 %)	–	–	952,727 (13.6 %, 96.9 %)
5	4 (0.1 %, 100.0 %)	–	–	722,937 (10.3 %, 97.9 %)
Rest	–	2 (0.1 %, 0.0 %)	–	296,173 (4.2 %, 97.1 %)
Total	4,564 (100.0 %, 64.7 %)	1,451 (100.0 %, 69.5 %)	68 (100.0 %, 88.0 %)	6,988,532 (100.0 %, 96.6 %)

4.4.2. Vendor servers. We now discuss some of our findings that are specifically related to the services provided by different vendors. These servers were, with some rare exceptions, either stratum 1 or stratum 2, indicating that they are well maintained and reputable sources of time.

Alibaba Cloud (Aliyun) – The reason why we treat these servers as falling under vendor servers, despite the fact that we did not crawl them separately, stems from their importance on the global scale. During our observation period, we saw four Alibaba servers, all of which had upstreams located in an RFC 1918 address space; a brief search revealed it as being an internally used address for `ntp1.aliyun.com`.

In total, we identified 213,575 servers (in the scan performed on the 20th January 2017) as depending on the 10 time servers provided by Alibaba. Interestingly, 53,796 (25 %) of those depended on six servers that are only available through the RFC 1918 address space. The reason why we mark them as belonging to Alibaba is that a brief search on those IP addresses suggested that they are theirs.

Furthermore, the following two reasons explain the popularity of these servers: First, considering the numbers of potential users and the servers available (our active probing saw 37 servers geolocated to China, whereas 92 were seen in the CN zone, likely thanks to recent "bootstrapping" [30], in which servers from elsewhere were added), volunteering a server will require investing in hardware and network capacities. Second, the location of a control server and the potential packet losses that may occur between it and the server may make it hard for a server to stay available to the pool (see [31]).

Apple – Apple uses different domains for different device categories and geographical locations; however, these were all simply aliases of domains under `g.aaplimg.com` and were resolved to a total of 8 IP addresses, all of which had appropriate, geolocation-based reverse DNS records. All of these, with a single exception, reported stratum 1 and based their time on GPS.

Canonical/Ubuntu – Canonical, known for its Ubuntu operating system, uses `timesyncd` and `ntp.ubuntu.com` instead of the pool per default. All of its six servers (four IPv4, two IPv6 addresses) consistently reported themselves as being stratum 2, using a set of 17 well-chosen upstreams. A total of 94 servers in our data set used these servers as their downstreams at some point.

Google – In comparison to Apple and Microsoft, each domain from Google provided a static set of IP addresses, all of which were stratum 2 and provided both IPv4 and IPv6 connectivity.

Given the purposes of our study, it was interesting to note that a total of 27 downstreams seen in the pool used these servers as their upstream. At the end of 2016, Google announced that their time servers are open for public use [32]; however, due to their use of leap smearing—a practice in which the clock is either slowed down or made faster to account for leap seconds [33]—their usage is discouraged when used in the pool.

Microsoft – A total of nine IPv4 addresses from various Microsoft-assigned networks were identified behind `time.windows.com`, none of which had a reverse DNS set. Surprisingly, these are also stratum 2 servers, and, perhaps even more surprisingly, they reported only two NIST servers as their upstreams, `utcnist2.colorado.edu` and `time-c.timefreq.bldrdoc.gov`.

NIST Nine of the 14 NIST servers behind `time.nist.gov` were also seen in the pool (`north-america.pool.ntp.org` and `us.pool.ntp.org`). The importance of these servers was also clearly visible on the network scan results, where two of the servers were in the top ten of most common upstreams with over 30,000 downstreams each.

4.5. Case Studies

Leap Second on 31st December 2016 – At the end of 2016, we encountered an addition of a leap second, a fact that conforming NTP servers should have accounted for by setting their leap flag to '1' (see Figure 1). Although our data set does not contain information regarding which servers were in the pool on that date, we combined information from our earlier data set of NTP responses and the graph in order to perform this analysis.

The results of this experiment are depicted in Table 4. 90 % of the servers from the pool as well as from the stratum listing were reported as having adjusted to the upcoming leap second correctly; to our surprise, a greater percentage of the pool servers made this change than all other servers from ntp.org category. All of the NIST servers reported the upcoming leap-second and most of Apple's servers did the same. Unfortunately, we had not added Microsoft's servers to the crawler at this point. Google's servers did not announce the upcoming leap second, as was expected.

Table 4. SERVERS REPORTING ON THE UPCOMING LEAP

	Pool	ntp.org	Vendors
Leap	3,492 (90.6 %)	1,199 (90.3 %)	45 (75.0 %)
No leap	361 (9.4 %)	129 (9.7 %)	15 (25.0 %)
Total	3,853 (100.0 %)	1,328 (100.0 %)	60 (100.0 %)

Case: Microsoft providing incorrect time – On the 3rd of April 2017, Microsoft's timeservers were providing incorrect time when not completely unavailable [34]. It seems that some Microsoft servers offered incorrect time; at the same time, they signaled through stratum that they were not synchronized (see Figure 7). The fact that this even got reported seems to indicate that some NTP clients are not handling out-of-sync stratum values correctly, we did not however verify this.

5. Serving the Pool

After analyzing the NTP server infrastructure from a client's perspective, we now turn to the operator's viewpoint, as we wish to understand the overall NTP ecosystem. We operated NTP servers on three different continents (Asia, Europe, and North America) and utilized these servers for data acquisition purposes by inserting them into the NTP Pool Project. We used the NTP Pool Project because of its importance, as discussed in Section 4. As we wished to understand what an operator can learn about its clients, we had to operate servers, as it is not enough to gather these kinds of data passively.

Table 5 shows the servers' locations, the average number of unique IP addresses that contacted these servers, the average number of received NTP requests, and the average number of clients per day. The last column indicates the periods of time during which our servers were seen during the active crawling phase described in Section 3.2 in the most recognized domains (namely `de.pool.ntp.org` (Europe, Germany), `north-america.pool.ntp.org` (North America, USA), and `sg.pool.ntp.org` (Asia, Singapore)). This value can be used as a measurement of how often our servers were resolved.

Table 5. NTP SERVERS OVERVIEW

Location	Unique IPs	Requests/day	Clients	Seen
DE	~ 72,000	~ 600,000	~ 78,000	50 min
USA	~ 435,000	~ 3,500,000	~ 480,000	30 min
SGP	~ 4,700,000	~ 25,000,000	~ 5,300,000	9.5 d

During the data-gathering phase (between September 2016 and March 2017), we collected more than 4 billion NTP requests. In total, we received an average of 29 million NTP requests per day. However, there was a significant difference in the distribution: The server in Europe received approximately 2 %, the server in North America received approximately 12 %, and the server in Asia received approximately 86 % of all requests. Figure 8 shows the total number of requests received by each server over the 5-month period. The red line indicates the day of the 16th of December, 2016, on which the Snapchat incident occurred, which we describe in more detail later. The traffic load was distributed to more NTP servers in Europe and North America. Therefore, they received fewer requests than the server in Asia. One reason for this behavior was already discussed in Section 4.3: Our servers were seen a different number of times in different domains (see Table 5).

In the following section, we analyze the usage of these pool-serving servers in detail. Subsequently, we compare our results with the results of a usage analysis of the NIST Internet time service [26] to demonstrate the general validity and comparability of these results. Finally, we introduce a method for determining a client's NTP implementation and discuss how one can distinguish between NTP-like and SNTP-like clients. Moreover, we discuss several noteworthy incidents that we observed during our data-acquisition phase.

5.1. NTP Server Clientele

The lower bound of seen clients is presented in Table 5. The number of unique IP addresses observed is not sufficient to determine the number of unique clients, e. g., due to IP address changes (IP churn). Following Padmanabhan's [35] investigations into IP address changes, we utilize a maximum analysis time range of one day, as Padmanabhan's results indicate that a maximum of one day leads to the greatest probability of not encountering IP churn. Moreover, we build tuples of client information in order to determine the number of possible clients more precisely. We define the observed TTL value to the next larger value of 32, 64, 128 or 255 as the initial TTL value. We use this initial TTL value, the source IP, the used port range, and the NTP version for building these tuples. It is important to note that this approach does not allow us to identify clients behind a NAT system or a firewall, so the actual number of clients will likely be even higher. Therefore, we consider simply giving a lower bound of unique clients.

First, we describe general information concerning our servers and clientele; we begin with a traffic analysis and discuss our findings and results. The distribution of observed UDP source port number ranges in NTP requests shows that most of our clients use the range between 32,767 and 65,535. Linux-based systems often use this port range [36]. Therefore, it seems that the majority of our clients use Unix-like systems. Port 123 is used by approximately 20 % of all servers, so according to Sherman et al. [26] possible ntpd implementations. Interestingly, the other distributions are approximately equal on each server, except for the range between 10,000 and 32,767, where, we saw twice as many requests in Asia when compared to the other servers.

More than 90 % of our observed clients use an initial TTL value of 64. The distribution of the initial TTL values supports our assumption that the majority of our clients are Unix-like systems. All distributions were identical on each server.

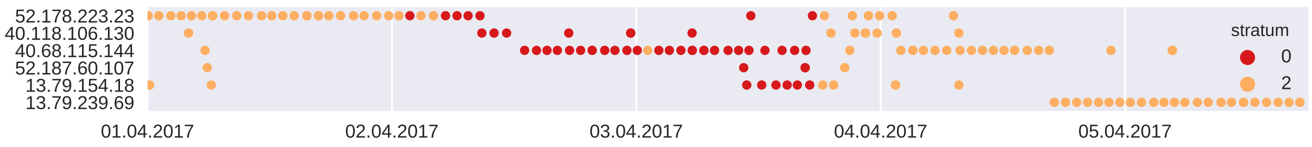

Figure 7. Unsynchronized Microsoft servers showing stratum and server changes for `time.windows.com` on 3rd of April 2017

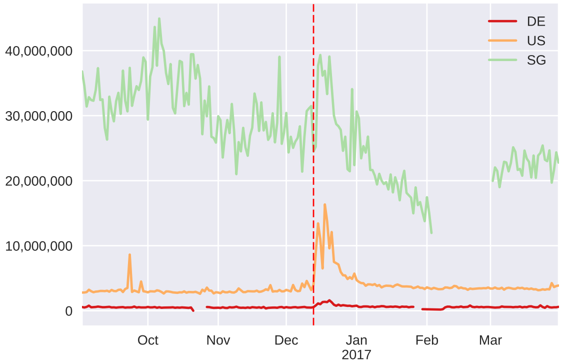

Figure 8. NTP requests per day on all servers

NTP packet headers – We investigate the NTP packet headers by inspecting the packet content in detail. Table 6 summarizes the relevant results of the statistical analysis of the header values of the NTP packets sent in the seen client requests. The results are representative because the average network load of each server seemed to be stable during our data-gathering phase, except during the second half of December. The most seen NTP version in Europe and North America is version 4, accounting for 62 % and 77 % of all requests, respectively. It is noticeable that, in Asia, NTP version 3 is most used, at approximately 68 %. This is interesting since the number of the old version 3 is greater than in Europe, with 36 %, and in North America, with 15 %. NTP version 1 is most common in North America, with a proportion of about 8 %. As expected, most of the clients did not fill the stratum field: In Europe and North America, approximately 65 % left it empty, with 81 % doing so in Asia.

The other values are relatively comparable among our three servers. The leap field was usually also '0'. Most of our analyzed queries do not use the refid field, with approximately 64 % in Europe and 78 % in Asia and North America. With the aid of the refid field, it is also possible to identify booting ntpd implementations; however, the number of such messages (INIT) was very low on each server.

When comparing our servers with each other and with the results of Sherman et al. [26], it is noticeable that the type of client differs between the different continents. The values of the leap field are comparable among the NIST servers and our servers; the values in Europe and North America are identical; while in Asia 91 % of our seen traffic had no leap second. On the other hand, the NIST server experienced approximately 10 % fewer requests with a value of '0'. However, the primary distribution remained the same on all servers. There are significant differences when it comes to the refid field: In Europe, the share of the unused refid fields is the lowest. Asia and North America are identical, and over 90 % of the requests on the NIST servers did not use this field. This result can be an indication that the least SNTP implementations can be found in Europe. A more detailed analysis on that can be found in Section 5.2. The distributions of NTP versions were also different. In Europe and North America, we mostly captured the newest NTP version 4, while Asia and the NIST servers saw the most requests with NTP version 3.

The distribution of stratum also shows differences. Most clients with stratum 0 request the NIST servers, with about 89 % compared to our client analysis. However, the server in Asia comes close to the NIST servers.

AS numbers – Finally, we consider the distribution of AS numbers under our recorded requests. As expected, large Internet providers from the corresponding country are represented in the top results: e. g., about 50 % of our clients in Europe (i. e., Germany) originate from Deutsche Telekom (3320, ∼31 %), Vodafone (3209, ∼15 %), and Telefonica (39706, ∼6 %). The top three AS numbers in the USA are the following: Comcast Cable Communications (7922) with ∼16 %, Amazon (14616) with ∼9 %, and Frontier Communications of America (5650) with ∼5.8 %. In Singapore, we see Amazon (38895) with ∼22 %, China Telecom Backbone (4134) with ∼18 %, and China Unicom Backbone (4837) with ∼10.7 % in the top three. The shares of specialized mobile communication providers in Asia are higher than in Europe and North America.

5.2. Client Software Identification

Using the results from the above client analysis, we can determine a class of client NTP implementations. If we can group NTP implementations, we can match clients with a greater probability. To identify subgroups in our client data, we had to find practical attributes. We attempt to make use of hard-coded values: e. g., the dispersion and delay field are only matched against two fixed values, or it is something else; the refid, stratum and interval are only considered in boolean form, and we consider only the values shown for the TTL, source port, and version. With these attributes, we build vectors that represent particular implementations.

Our approach of grouping objects by similarity makes use of a manually built decision tree. This decision tree

Table 6. NTP HEADER FIELD ANALYSIS OF OBSERVED CLIENT REQUESTS TO OUR SERVERS

Server	Stratum					Refid			Version				
	0	2	3	4	-	0	INIT	-	1	2	3	4	-
Europe	65 %	7 %	24 %	2 %	<1 %	64 %	<1 %	35 %	2 %	<0.1 %	36 %	62 %	<0.1 %
USA	65 %	7 %	24 %	2 %	<1 %	78 %	<1 %	22 %	8 %	<0.1 %	15 %	77 %	<0.1%
Asia	81 %	14 %	4 %	1 %	<1 %	78 %	<1 %	22 %	<1 %	<0.1 %	68 %	31 %	<0.1 %

Table 7. SHARE OF SEEN IMPLEMENTATIONS

	Europe	North America	Asia
ntpd	26.4 %	18.3 %	22 %
chrony	4.8 %	1.6 %	1.2 %
openntpd/...	4.4 %	24%	3.5 %
win10 nettime	1 %	0.2 %	0.5 %
ntpclient	1.2 %	1.6 %	2.9 %
Android	14 %	10.7 %	31.7 %
ntpdate/...	20 %	23.9 %	7 %

was constructed after analyzing data gathered from NTP implementations on various operating systems and appliances. In particular, we created a set of 38 different NTP client-operating system combinations, e. g., Android, Linux systems, routers and Windows systems. With these results, we were able to determine value combinations (vectors) to identify and match 31 different groups of NTP implementations. However, while we cannot always name the exact implementation, we can recognize similar implementations.

Most of our client requests come from Unix-like systems, using the reference ntpd implementation (on average ∼22 %). Table 7 summarizes the largest shares of other seen implementation groups that we were able to categorize. The share of the Android group is high in Asia; as such, there are probably more mobile devices that requested our server, since the share of mobile specific providers is higher, as was seen when analyzing the AS numbers previously.

We categorized the groups as NTP-like (20 groups, e. g., ntpd, chrony), SNTP-like (7 groups, e. g., ntpclient, Android, ntpdate/ios_ntp_lib/...), and unknown (4 groups) implementations.

The results on the real world data are satisfactory: We can distinguish about 95 % of our traffic and group it to SNTP-like or NTP-like implementations.

Figure 9 summarizes the results for our server in Europe. Generally speaking, about 50 % is NTP-like traffic, and 45 % is SNTP-like traffic, while the results from the US look very similar. In Asia, we see more SNTP-like traffic, with approximately 75 % SNTP-like and only 20 % NTP-like traffic. These results indicate that the usage of SNTP-like clients is especially widespread in Asia. This extends the results of Mani et al. [37] regarding all clients relying on the NTP Pool Project and not only mobile devices.

5.3. Case Studies

To enrich the observational nature of the client analysis, we describe case studies that occurred during the measurement period.

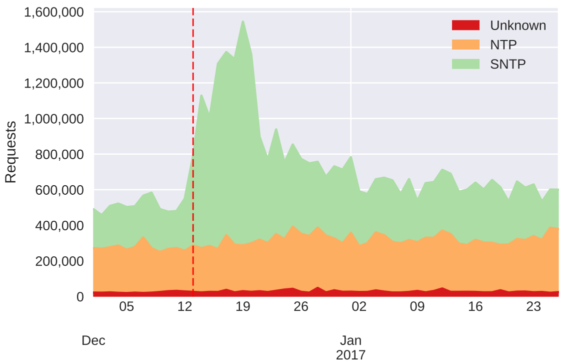

Figure 9. Traffic classification of Europe

Case: Snapchat incident – We noticed a substantial rise in clients requesting our servers during the middle of December, 2016. The NTP pool status website approved the increased load we saw, as they had also seen massively increased NTP traffic load in some countries.The vertical red line in Figure 9 highlights the starting point on the 13th of December 2016. It was identified as being a buggy Snapchat app update on the iOS operating system and was a global incident [38]; we were able to see a rise in the number of requests on each server (see Figure 8). This increase SNTP-like requests highlights that our NTP client grouping works correctly, as Snapchat makes use of the SNTP-implementing iOS NTP library [39]. The issue was fixed in a week, leading to normalization of the traffic volume.

Case: Leap second – We also see the effect of the leap second at the end of 2016. We notice on all servers a rise in requests with leap field '1' indicating a closing minute with 61 seconds. The share of these requests reached its peak in Germany at almost 24 %, in the USA at 5 %, and in Singapore at 13 %. Furthermore, few clients seemed to have problems with the leap second because two groups, i. e., timesyncd/openntpd/busybox_ntpd and ntpd 4.2.8p5, showed a slight increase in the number of requests around the turn of the year, particularly in North America.

6. Potential Security Implications

NTP plays a crucial practical role since the availability of correct time information is necessary for many applications. These applications include, e. g., public key infrastructures, authentication mechanisms, digital currencies, and more.

In this section, we discuss the possible security implications due to the centralized and partially voluntary nature of the NTP infrastructure. In comparison to attacks discussed by Malhotra et al. [3], which concentrated on attacks on the reference implementation, we demonstrate how a malicious service provider could attack its users.

In contrast to NTP with its clock selection algorithm, a large amount of users using SNTP (as shown in Section 5.2) are more vulnerable to malicious time sources. This is becoming more and more crucial in the era of the Internet of Things, given that such devices often utilize SNTP.

6.1. Centralized Infrastructure

The NTP ecosystem is centralized in two ways: On the one hand, the NTP infrastructure itself is centralized considering its tree-like hierarchy, as shown in Section 4. This centralization leads to possible security implications, since there are only a number of servers in charge of giving the correct time to a large number of other servers and clients.

On the other hand, the NTP Pool makes use of a single server in the USA to check whether the servers are returning correct time to the clients [40]. This may be at least partially responsible for the sparse server population in countries with poor connectivity to that server. In case of a failure, it would stop providing circulation of servers to all zones, which, while not directly affecting the clients, could possible cause overburdening the active servers in the longer run.

6.2. Poisoning the Pool

To verify that the pool system is serving the servers fairly, we analyzed the possibility of implanting a sufficient number of servers to some sparsely populated parts of the pool. The goal is to gain the majority for the purpose of all of those servers selected simultaneously from clients, thereby making them a majority for right chimer selection.

To test this, we added three more servers into the pool in Singapore, which is much more sparsely populated than many other zones. The fewer servers there are in the pool, the higher the probability that we will be selected multiple times. We analyzed the intersection of IP addresses seen on at least three of our servers during that time-period as a criterion. We define a client as being simultaneously on the servers when we see a request from the same IP address inside a minute's time span. On one occasion, we saw only 8,800 IP addresses within 12 hours on the 1st of December, 2016, indicating that this type of attack is merely of a speculative type on the sparsely populated Singaporean zone; this would be nearly impossible on the heavily populated pools in Europe or North America.

Hijacking a Country – In the normal case the Pool's DNS server resolves global zones (including the commonly hardcoded `pool.ntp.org` and vendor zones) to a nearby zone via GeoIP. If that zone is empty, issuing a DNS request *directly* on it will return no results. However, in the case of an empty country zone, the system goes up in the hierarchy (i. e., to the continental and global zones). Therefore the clients residing in an area without any servers in their primary zone will be provided results nevertheless.

The fact that at the moment there exist over 150 zones without any servers– and on top of that some zones having less than a fair number of servers—raises the question what will happen if a server is inserted into one of those zones.

In order to verify this, we tested adding one server to a zone with no other servers. Before our server was included into our desired zone, we verified that making a request from our target country—with the help of RIPE Atlas [41]—resulted in a response with four servers. After inclusion of our server the same query returned just our address confirming our suspicions.

Considering the nature of this experiment we carefully chose a small country located near our hosting location and made it only available over IPv6 to avoid larger exposure to the clientele. Using IPv6 also serves as a way to circumvent the requirement of our server to be in the target country; in case of a non-existing geolocation for the IP address, the operator is allowed to pick the location of the server. Furthermore we immediately removed the server from the pool—without receiving any substantial traffic—after verifying the expected behavior while leaving the server on for clients which may receive our IP address from DNS caches.

Although this problem has already been known to burden operators of small zones [42], its security and privacy implications may not be well understood. We have contacted the pool maintainers to clarify this issue, and we have prepared a preliminary patch to fix it.

We also used RIPE Atlas to see if DNS hijacking is prevalent by resolving commonly used NTP domains with all probes. Merely 25 probes out of 8,428 responded with DNS replies pointing to unexpected IP addresses: 11 resolved to private IP addresses and 14 to other known, benign NTP servers. This seems to suggest that network operators use such measures to keep their synchronization local, and thereby unintentionally bypass the protection offered by the clock selection algorithm.

6.3. Skewing the Clock

We tested empirically how several types of devices behave when being fed the incorrect time: an IP camera, a smart electronic socket, a consumer grade Network Attached Storage (NAS), and a desktop computer. The security camera was running the reference implementation with only one domain. Furthermore, the NAS used "chrony" and its pool option to avoid this issue by using all of the returned IP addresses, however, as shown earlier, some parts of the pool or vendor services may just provide a single IP address for a request. On the desktop, we tested out the reference implementation along with a SNTP-implementing `timesyncd` [43]. The smart socket used a proprietary implementation which kept rotating over a list of domains to find a usable server. Given enough time, all of these devices adjusted their clocks to times fed by us.

Another necessity for skewing the time successfully is to get around the limitations of the (S)NTP client used;

e. g., the reference implementation ntpd allows jumps only during the initialization and defines a maximal threshold of 1,000 seconds as a protection mechanism, as shown by Malhotra et al. [3], who also showed how this mechanism could be circumvented. What kind of protection mechanisms exist and how effective they are against time skewing is not discussed further in this paper, but it is sufficient to say that clients need, in general, to have a mechanism to handle large jumps in time.

7. Related Work

NTP Infrastructure Analysis – Sherman et al. [26] recently analyzed the usage of the NIST Internet Time Service, and Malhotra et al. [3] briefly discussed the status of nowadays NTP ecosystem. Previously, Minar et al. [4] conducted a survey in 1999 about the server infrastructure of the NTP network, in which they estimated the NTP network as containing at least 175,000 servers. Going back in history, Mills [44] gave a short introduction to the role of NTP in the Internet system in 1991. In 1994, Guyton et al. [45] used monlist responses to build an overview of contemporary NTP infrastructure. Mani et al. [37] described a new protocol particularly for mobile devices that was designed to be simple and efficient. They observed almost all mobile devices use SNTP; this fits our results.

Attacking NTP – Malhotra et al. [3], [12] focused on vulnerabilities in the NTP protocol and its reference implementation, aiming at ways to manipulate the time by both on-path and off-path attackers. Park et al. [19] presented a security analysis and discussed attacks on the clock in cellular networks (NITZ) and NTP on mobile devices. In addition, Czyz et al. [2] analyzed NTP amplification attacks, which still remain a major source of DDoS attack traffic.

Hardening the Protocol – The NTP working group of IETF and various other actors are currently working on making the NTP protocol more robust against malicious actors. The method we used to gather the data for our graph may become impossible shortly with a proposal of replacing unique refids with a single “NOT-YOU” refid [46]. Another proposal plans to minimize the amount of information sent to servers, which in turn would hinder the ability to fingerprint the clients. [47] There are also steps toward introducing authenticity, integrity, and confidentiality to communications between authenticated NTP peers with DTLS [48], [49]. On the implementation front, the NTPsec project has been working on hardening the reference implementation [50]. Lastly, Dowling et al. [51] presented an authenticated version of NTP to protect against desynchronization attacks.

8. Conclusions and Future Work

In this paper, we studied NTP from both a client and a server viewpoint and analyzed different aspects of the entire NTP ecosystem. Our analysis revealed a dependency on a small number of NTP servers on the global scale. The NTP pool plays a particularly crucial role, and we need to better study the implications of this observation to assess involved risks. The following topics could be studied as part of future work:

NTP pool changes – On the NTP Pool Project’s side, there have been ongoing discussions [52] regarding introducing DNSSEC for the zones, thus making DNS spoofing impossible when using validating resolvers. As some operators seem to forward their users to more local NTP servers over DNS, this option should be studied to understand its implications. Furthermore, there have been discussions about filling the sparsely populated zones with servers from other zones [30], [53], which in turn would make it harder for an adversary to fill country-specific zones with rogue servers. Besides balancing the burden away from operators serving sparsely filled zones, this would fix the possibility to overtake empty or sparsely populated pools and alleviate potential privacy issues.

Disabling control mode – Server administrators should be encouraged to close the control mode responses on external sources if they are not necessarily needed in order to minimize the potential attack surface. As we have seen, there exist many control-mode enabled servers which should be deactivated.

Side-channel “NTP” – To protect against clock skew, Windows 10 uses the information available within SSL handshakes to calculate limits for allowed jumps in the current time setting [54]. To the best of our knowledge, no such mechanism is currently employed by other operation systems so far. The potential implications and the implementation of such mechanisms on other operating systems could be explored. Taking multiple time signals into account when determining the current time seems to be a promising strategy to prevent certain kinds of attacks.

Access to data set. The data collected by our crawler, our unaltered graph database as well as the database consisting of NTP client mode scan results are available at `https://github.com/RUB-SysSec/MastersOfTime`. Furthermore anonymized snapshots of client requests are provided, taking the ethical considerations based on Allman and Paxson [55] into account.

Acknowledgment

We would thank all our reviewers for their helpful and valuable feedback. This work was partially supported by the German Federal Ministry of Education and Research (BMBF grant 01IS14009B “BD-Sec”).

References

[1] M. Kührer, T. Hupperich, C. Rossow, and T. Holz, “Exit from Hell? Reducing the Impact of Amplification DDoS Attacks,” in *USENIX Security Symposium*, 2014.

[2] J. Czyz, M. Kallitsis, M. Gharaibeh, C. Papadopoulos, M. Bailey, and M. Karir, “Taming the 800 Pound Gorilla: The Rise and Decline of NTP DDoS Attacks,” in *ACM SIGCOMM Conference on Internet Measurement*, 2014.

[3] A. Malhotra, I. E. Cohen, E. Brakke, and S. Goldberg, "Attacking the Network Time Protocol," in *Symposium on Network and Distributed System Security (NDSS)*, 2016.

[4] N. Minar, "A Survey of the NTP Network," http://www.media.mit.edu/ nelson/research/ntp-survey99/, 1999.

[5] D. L. Mills, "RFC 958: Network Time Protocol (NTP)," 1985.

[6] J. Burbank, D. Mills, and W. Kasch, "RFC 5905: Network time protocol version 4: Protocol and algorithms specification," 2010.

[7] D. Mills, J. Levine, R. Schmidt, and D. Plonka, "Coping with Overload on the Network Time Protocol Public Servers," DTIC Document, Tech. Rep., 2005.

[8] D. L. Mills, "RFC 4330: Simple network time protocol (SNTP) version 4 for IPv4, IPv6 and OSI," 2006.

[9] Akamai, "Q4 2016 State of the Internet / Security Report," https://content.akamai.com/pg7967-q4-soti-security-report.html.

[10] D. L. Mills, "ntpq manual," https://www.eecis.udel.edu/ mills/ntp/html/ntpq.html.

[11] "CVE-2013-5211," http://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2013-5211, 2013.

[12] A. Malhotra, H. Kennedy, M. Varia, M. Van Gundy, J. Gardner, and S. Goldberg, "The Security of NTP's Datagram Protocol," in *Financial Cryptography and Data Security*, 2017.

[13] T. Rytilahti and T. Holz, "Poster: The Curious Case of NTP Monlist," European S&P 2016, 2016.

[14] The Shadowserver Foundation, "Open NTP Monitor Scanning Project," https://ntpmonitorscan.shadowserver.org.

[15] "NTP Pool Project – pool.ntp.org: public ntp time server for everyone," http://www.pool.ntp.org.

[16] Hurricane Electric, "IPv6 Tunnel Broker," https://www.tunnelbroker.net.

[17] D. D. Chen, M. Egele, M. Woo, and D. Brumley, "Towards Automated Dynamic Analysis for Linux-based Embedded Firmware," in *Symposium on Network and Distributed System Security (NDSS)*, 2016.

[18] M. Garrett, "A quick look at the ikea trådfri lighting platform," https://mjg59.dreamwidth.org/47803.html, 2017.

[19] S. Park, A. Shaik, R. Borgaonkar, and J.-P. Seifert, "White Rabbit in Mobile: Effect of Unsecured Clock Source in Smartphones," in *Workshop on Security and Privacy in Smartphones and Mobile Devices*. ACM, 2016.

[20] "The NTP Pool for vendors," http://www.pool.ntp.org/vendors.html.

[21] A. Hansen, "NTP Pool Project's github," https://github.com/abh/ntppool.

[22] ——, "geodns: DNS server with per-client targeted responses," https://github.com/abh/geodns.

[23] "NTP.org public time server lists," http://support.ntp.org/bin/view/Servers/WebHome.

[24] S. Bishop, "Ntp stratum one and two server status," http://ntp.exactlywww.com/ntp/.

[25] "NIST Internet Time Service," http://tf.nist.gov/tf-cgi/servers.cgi.

[26] J. A. Sherman and J. Levine, "Usage Analysis of the NIST Internet Time Service," *Journal of Research of the National Institute of Standards and Technology*, vol. 121, p. 33, 2016.

[27] Team Cymru, http://www.team-cymru.org/, 2017.

[28] H. Asghari, "pyasn," https://github.com/hadiasghari/pyasn.

[29] MaxMind GeoLite2 Database, https://dev.maxmind.com/geoip/, 2017.

[30] A. Hansen, "Adding servers to the China zone," https://community.ntppool.org/t/adding-servers-to-the-china-zone/88, 2017.

[31] "[Pool] ntp pool service in China," http://lists.ntp.org/pipermail/pool/2016-June/007861.html, 2016.

[32] M. Shields, "Making every (leap) second count with our new public NTP servers," https://cloudplatform.googleblog.com/2016/11/making-every-leap-second-count-with-our-new-public-NTP-servers.html, 2016.

[33] "Leap Smear - Google Public NTP," https://developers.google.com/time/smear, 2017.

[34] Mix, "Windows Time Service is sending out wrong times and thats a big problem," https://thenextweb.com/microsoft/2017/04/03/windows-time-service-wrong/, 2017.

[35] R. Padmanabhan, A. Dhamdhere, E. Aben, k. claffy, and N. Spring, "Reasons Dynamic Addresses Change," in *ACM SIGCOMM Conference on Internet Measurement*, 2016.

[36] Cymru, "Ephemeral Source Port Selection Strategies," https://www.cymru.com/jtk/misc/ephemeralports.html.

[37] S. K. Mani, R. Durairajan, P. Barford, and J. Sommers, "MNTP: Enhancing Time Synchronization for Mobile Devices," in *ACM SIGCOMM Conference on Internet Measurement*, 2016.

[38] "Excessive load on NTP servers," https://status.ntppool.org/incidents/vps6y4mm0m69, 2016.

[39] J. Benet, "SNTP implementation for iOS," https://github.com/jbenet/ios-ntp, 2017.

[40] "[Pool] Why only one monitoring station," http://lists.ntp.org/pipermail/pool/2016-October/007963.html.

[41] Ripe, NCC, "RIPE Atlas," http://atlas.ripe.net.

[42] "[Pool] "Bulk" servers for underserved countries," http://lists.ntp.org/pipermail/pool/2016-July/007906.html.

[43] systemd-timesyncd, https://wiki.archlinux.org/index.php/systemd-timesyncd, 2017.

[44] D. L. Mills, "Internet Time Synchronization: the Network Time Protocol," *IEEE Transactions on Communications*, vol. 39, no. 10, pp. 1482–1493, 1991.

[45] J. D. Guyton and M. F. Schwartz, "Experiences with a Survey Tool for Discovering Network Time Protocol Servers," in *USENIX Summer*, 1994.

[46] Stenn, H. and Goldberg, S., "Network Time Protocol REFID Updates (draft)," https://tools.ietf.org/html/draft-ietf-ntp-refid-updates-00, 2016.

[47] Franke, D. and Malhotra, A., "NTP Client Data Minimization (draft)," https://tools.ietf.org/html/draft-dfranke-ntp-data-minimization-02, 2017.

[48] Sibold, D. and Roettger, S. and Teichel, K., "Network Time Security (draft)," https://tools.ietf.org/html/draft-ietf-ntp-network-time-security-15, 2016.

[49] Franke, D. and Sibold, D. and Teichel, K., "Network Time Security for the Network Time Protocol (draft)," https://tools.ietf.org/html/draft-ietf-ntp-using-nts-for-ntp-09, 2017.

[50] NTPsec project, "NTPsec homepage," https://www.ntpsec.org.

[51] B. Dowling, D. Stebila, and G. Zaverucha, "Authenticated Network Time Synchronization," in *USENIX Security Symposium*, 2016.

[52] A. Hansen, "Enabling DNSSEC signing," https://community.ntppool.org/t/enabling-dnssec-signing/57, 2016.

[53] Chen, CS, "Fill zero-server pools with CNAME," https://community.ntppool.org/t/fill-zero-server-pools-with-cname/132, 2017.

[54] Madakasira, Sarath, "Secure Time Seeding improving time keeping in Windows," https://blogs.msdn.microsoft.com/w32time/2016/09/28/secure-time-seeding-improving-time-keeping-in-windows/, 2016.

[55] M. Alllman and V. Paxson, "Issues and Etiquette: Concerning Use of Shared Measurement Data," in *ACM SIGCOMM Conference on Internet Measurement*, 2007.