Deep Dive into NTP Pool Popularity and Mapping

Moura, Giovane C. M.; Davids, Marco; Schutijser, Caspar; Hesselman, Cristian; Heidemann, John; Smaragdakis, G.

Publication date
2023

Document Version
Final published version

Citation (APA)
Moura, G. C. M., Davids, M., Schutijser, C., Hesselman, C., Heidemann, J., & Smaragdakis, G. (2023). *Deep Dive into NTP Pool Popularity and Mapping*. (SIDN Labs Technical Report). SIDN Labs.

Deep Dive into NTP Pool's Popularity and Mapping

SIDN Labs Technical Report – 2024-01-15

Update from 2023-10-12

GIOVANE C. M. MOURA, SIDN Labs and TU Delft, The Netherlands
MARCO DAVIDS, SIDN Labs, The Netherlands
CASPAR SCHUTIJSER, SIDN Labs, The Netherlands
CRISTIAN HESSELMAN, SIDN Labs and University of Twente, The Netherlands
JOHN HEIDEMANN, USC/ISI and CS Dept., USA
GEORGIOS SMARAGDAKIS, TU Delft, The Netherlands

Time synchronization is of paramount importance on the Internet, with the Network Time Protocol (NTP) serving as the primary synchronization protocol. The NTP Pool, a volunteer-driven initiative launched two decades ago, facilitates connections between clients and NTP servers. Our analysis of root DNS queries reveals that the NTP Pool has consistently been the most popular time service. We further investigate the DNS component (`GeoDNS`) of the NTP Pool, which is responsible for mapping clients to servers. Our findings indicate that the current algorithm is heavily skewed, leading to the emergence of time monopolies for entire countries. For instance, clients in the US are served by 551 NTP servers, while clients in Cameroon and Nigeria are served by only one and two servers, respectively, out of the 4k+ servers available in the NTP Pool. We examine the underlying assumption behind `GeoDNS` for these mappings and discover that time servers located far away can still provide accurate clock time information to clients. We have shared our findings with the NTP Pool operators, who acknowledge them and plan to revise their algorithm to enhance security.

Additional Key Words and Phrases: NTP; NTP Pool; DNS; Measurements; Client mapping

1 INTRODUCTION

Global time synchronization underpins modern life. It is crucial to the Internet and to critical systems such as financial markets, power grids, and telecommunications networks [30]. In businesses, precise clock information is also vital: distributed systems and applications such as backup systems are entirely dependent on precise clock information [40, 70]. Operational failures can occur whenever clocks are unsynchronized, potentially leading to data loss [27].

On the Internet, many commonly used applications, services, and protocols depend on clock correctness for secure operations. TLS [19], DNSSEC signatures [2], DNS caches [54], RPKI [11], Kerberos [58], and even Bitcoin transactions are are some of the applications that depend on clock synchronization to prove cryptographic freshness [18, 43, 50, 78]. In November 2021, the US Navy Naval Observatory's (USNO) NTP servers [71] reporting time roughly 12 years incorrect, resulting in outages in multiple places, including Active Directory servers and and routers [43, 46].

The Network Time Protocol (NTP) [50] is the Internet's default protocol for clock synchronization. It is designed to mitigate the effects of changing network latency (jitter) between client and server. NTP *servers* synchronize out-of-band with high precision references, such as atomic clocks, radio signals (*e.g.,* DC77 [10]), and satellites (GPS and Galileo). Clients and other secondary NTP servers, in turn, synchronize their clocks with NTP servers over the Internet. Clients either use servers they have been pre-configured with (*e.g.*,, `/etc/ntpd.conf`) or servers provided by their networks with DHCP [21, 24]. The Precision Time Protocol (PTP) [31, 32], in turn, improves NTP's precision, but typically requires layer-2 (LAN) access. PTP is often used in financial transactions, mobile phone towers and other industrial networks.

There are many publicly available NTP servers on the Internet. NIST [59] and the USNO [71] have been providing NTP services for decades. Later, several vendors such as Apple [1], Google [25], Cloudflare [14], Meta [70], Microsoft [48] and Ubuntu [84] started their own services.

The NTP Pool [65] provides a layer over NTP servers, providing a directory of publicly available NTP servers using DNS [51]; it does not directly operate NTP servers. The NTP servers themselves are run by volunteers, which range from home DSL users to large cloud operators. The NTP Pool currently lists 4,403 volunteer NTP servers, with 3,056 on IPv4 and 1,671 on IPv6 (2023-10-09) [61]. It has been operating for more than two decade, being popular among vendors [66, 78], including various Linux distributions and Android devices.

Our first contribution (§3) is to *demonstrate that the NTP Pool is not only in active use, but it has consistently been the most popular time-service provider on the Internet*, based on DNS traffic at the Root DNS servers [77].This popularity persists even with the introduction of new time services introduced by large vendors in recent years.

Our second contribution is to *demonstrate how these mappings are executed and which criteria are employed in this process.* We examine `GeoDNS` [6], the NTP Pool customized DNS software that perform the mapping, complemented by measurements taken from the public Internet Clarifying this process is important given the popularity of the NTP Pool.

Our third contribution (§5) is *to explore the implications of this mapping, from our ability to predict which NTP servers a client will use.* We find that assignments can be heavily skewed, producing time service monopolies. Even with more than 4k NTP servers, 27 countries are assigned to a single time provider—one operator serves 767M people and 465M Internet users. In addition, we find that another 101 countries and territories (comprising 260M Internet users) could be monoploized with the deployment of a *single* NTP server. This monopolization bestows immense power upon a single actor [60], which can then be misused (or exploited) to execute nation-wide scale time-shift attacks, particularly worrisome in today's world where conflicts extend into cyberspace.

fourth and final contribution (§6) is to *show that the current `GeoDNS` mapping algorithm can be changed to improve server distribution without compromising service quality.* Conversations with NTP Pool operators indicate that these mappings are designed to avoid asymmetric routing and alleviate concerns about packet loss. However, our experimental results contradict these apprehensions about substantial packet loss from distant servers: we demonstrate that far away NTP servers can also deliver high-quality timing services with minimal packet loss ratios. Consequently, we recommend that NTP Pool operators consider modifying their mapping algorithm to address these monopolization issues, which could potentially result in a complete or partial time synchronization takeover for entire countries (§7).

2 THE NTP POOL PROJECT

The NTP Pool project is a dynamic collection of thousands of NTP servers that provide accurate time via the NTP protocol [50] to clients worldwide. These NTP servers are assigned to clients using DNS, under the pool.ntp.org zone. It was proposed as a solution to reduce the abuse of publicly available NTP servers [86]. Instead of having a long list of public NTP servers (which individually could more easily become overloaded), the NTP Pool project proposed to "load balance" NTP traffic using DNS. The project has been active for more than twenty years.

The NTP Pool does not run NTP servers; volunteers run their own NTP servers which they add to the NTP Pool using a web interface [67], where they can also choose how much capacity they want to donate (values ranging from 512Kbps to 1Gbps). We show this process on the left side of Figure 1, where IP addresses are added with their respective *BW* capacity parameter.

To use the NTP Pool for clock synchronization, *clients'* time software is set with domain names from the NTP Pool DNS zone, as shown in the right side of Figure 1. In this figure, we see a client that uses the domain names [0--3].debian.pool.ntp.org in their config files (*e.g.*, `/etc/ntp.conf`). Whenever needed, the client will ask its local DNS resolver to fetch the IP addresses for these domain names (B in Figure 1), and the NTP Pool *authoritative* DNS servers ([a--i].ntpns.org) will

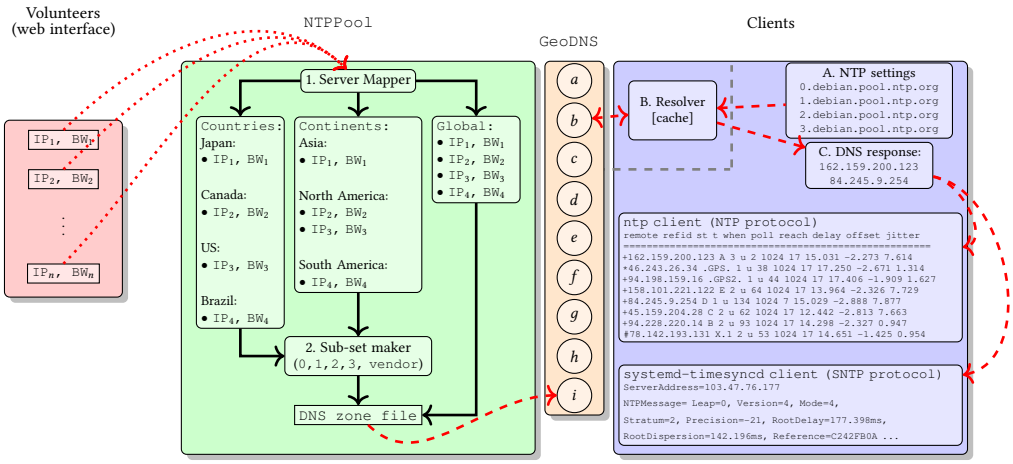


Fig. 1. NTP Pool operations: from volunteers to clients

answer with a list of IP addresses (C) to the DNS resolvers, which will then forward them to the clients. The NTP Pool runs `GeoDNS`, their customized authoritative DNS server software [6]. Upon receiving NTP servers addresses from the NTP Pool authoritative servers, the client's time synchronization software will contact these NTP servers for accurate time information.

Do clients blindly trust NTP servers? Consider a malicious NTP server that provides wrong time data – say, ten years ahead of the current time. To prevent attackers from tampering with the client's clock, the NTP protocol has built-in algorithms (§10 and §11 in [50]) that continuously evaluate the timing samples from various NTP servers, disregarding discrepant ones. Moreover, it can combine time information from multiple NTP servers to update the system's clock.

Figure 1 shows an example of a `NTP` reference implementation client status [57]. It shows 8 NTP servers under evaluation – all retrieved from the NTP Pool by the client (NTP clients use multiple NTP servers if they are available to select the best ones). Each server has several metrics (offset, delay, jitter), which are all part of the NTP protocol filtering specifications. This particular client combines data from all servers with marked with '*' and '+' symbols on the left side of the IP address to synchronize its clock. In this way, NTP implementations prevent the harmful effects of individual malicious NTP servers. The exception occurs when a server reboots and may trust whatever time information is provided with – or when ntpd is ran with the `-s` option.

SNTP [49], which is a simplified version of NTP designed to provide basic time synchronization functionality with minimal overhead, will blindly trust the time information provided by time servers. Even if an SNTP client receives multiple NTP servers from the NTP Pool, it will use only a single server. In Figure 1, we show the status of `systemd-timesyncd`, a SNTP implementation running on Ubuntu, where we see a single NTP server.

How does the NTP Pool prevent malicious volunteers? Anyone can add an NTP server to the NTP Pool. To prevent malicious or bogus NTP servers from being assigned to clients, the NTP Pool operators constantly monitor every volunteer NTP server. Bogus servers are removed from the zone and not served to the clients. (We demonstrate in §4.2 how this system works).

Changing system configurations: clients are typically configured with pre-configured NTP servers. They can manually change it or, if available, use the NTP servers provided by the DHCP [21]

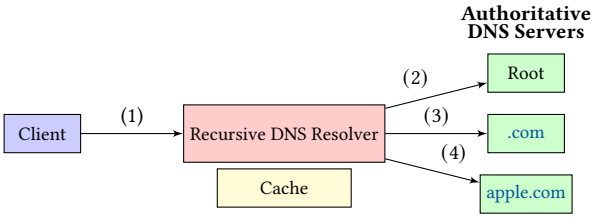


Fig. 2. Time servers domain name resolution.

protocol, which also enables dynamically setting NTP servers. For example, one of the authors institution provides NTP servers via DHCP, which causes the NTP client on Linux boxes to *not* use the NTP Pool while connected to the institution network.

3 EVALUATING THE IMPORTANCE OF THE NTP POOL

Given that several large cloud and content providers now have their own NTP services (Microsoft, Google, Facebook, Cloudflare, and Apple), one may wonder how relevant is NTP Pool still for keeping time on the Internet. In areas such as DNS resolution, no-cost services by commercial providers quickly captured a majory of their market [52]. Has the same happened for NTP, reducing the NTP Pool relevance?

The direct way to answer this question would be to compare NTP traffic across different time providers. That, however, would require access to vantage points inaccessible to us. Thus, we address this question by comparing the NTP Pool popularity with other NTP services by analyzing DNS traffic collected at the Root DNS servers [77].

3.1 Root DNS and time keeping

Before synchronizing clocks with NTP servers, clients must first resolve the domain names associated with the time service. Consider Figure 2 as an example, where a client first (step 1) sends a DNS query (time.apple.com) to its recursive DNS resolver, typically provided by its local ISP. This recursive resolver converts the name into the IP address providing service, either from its cache or by asking one more more authoritative name servers.

Assuming nothing is cached, the resolver must first contact one of the 13 Root DNS servers, asking for the authoritative server of .com zone authoritative servers (step 2 in Figure 2). The recursive resolver learns where .com, is, then asks the .com authoritative DNS server for authoritative servers of apple.com (step 3). Next, the apple.com authoritative servers can tell the resolver which IP addresses are associated with time.apple.com (step 4) and finally can answer its client. The client then uses the IP addresses to synchronize its clock using the NTP protocol.

While we cannot see DNS traffic from each time provider, we have access to traffic from the Root DNS servers with DITL datasets [20], a two-day-a-year traffic capture of the Root DNS servers. The Root DNS traffic provides a view of the top of global DNS traffic [13, 28, 41]. DITL can provide a lower bound estimate of the number of time services users.

3.2 Limitations

The DITL datasets from the root DNS have several limitations regarding our research question:

They do not see real clients: The root servers only see recursive resolvers used by clients (Figure 2), not actual clients. Since recursive resolvers employ caches and may provide cached results to many clients, observations at the DNS root do not allow us to tell how many are behind the resolver.

Provider	Server Name	TTL	TLD TTL
Apple	{time,time[1–7],time.euro, time.asia}.apple.com	2h	2 days
Cloudflare	time.cloudflare.com	5min	2 days
Facebook	{time,time[1–5],}.facebook.com	1h	2 days
Google	{time,time[1–4],}.google.com, time.android.com	4h	2 days
Microsoft	time.windows.com	1h	2 days
NIST	{time,time-[a,b,c,d,e]-[g,wwv,b]}.nist.gov,{utcnist,utcnist2.colorado}.edu	30min	2 days
NTP Pool	*.pool.ntp.org	2.5min	1h
Ubuntu	ntp.ubuntu.com	1min	2 days
USNO	{u,tock,ntp2}.usno.navy.mil	(<5min)	6h
VNIIFTRI	ntp[1–4].vniiftri.ru,ntp[1-2].niiftri. irkutsk.ru,vniiftri[,2].khv.ru	1 day	4 days
Rest	137 NTP servers – see §A	–	–

Table 1. Evaluated Time Providers and their records TTL, and their TLD's own TTL

Caches also hide repeated requests. Not only will caches hide multiple clients, they also hide repeated requests by single clients (to reduce latency [54, 55]).

Query name (qname) minimization hides services: qname minimization [9] improves user privacy by provided only a single DNS component to each authoritative DNS server. For example, instead of querying the root DNS servers for time.apple.com (which the roots cannot directly answer, they can only point to where the .com authoritative servers), a recursive resolver using qname minimization will only query the root for .com, hiding the full name. While study has shown that qname minimization is still not widespread [17], its use is growing [42]. Our method applies only to resolvers that do not use qname minimization.

Localroot avoides Root servers completely: A recent informational RFC suggested a mechanism where recursive resolvers preemptively fetch an copy of the root zone, allowing them to avoid querying root servers entirely [38]. We believe localroot is used far less than qname minimization, but we cannot see traffic for recursive resolvers using this mechanism.

3.3 Datasets

There are thirteen root DNS servers on the Internet. Each one is referred to by the first letter in its name ([a--m].root-servers.net). We analyze traffic collected at twelve of the thirteen root servers – the Day In The Life of the Internet (DITL) 2022 datasets [20]; we omit I-Root, since it's DITL datasets anonymizes IP addresses, preventing our analysis. In addition, E-root provides only partial data. This dataset was taken from April 12–14, 2022. For a historical comparison, we compare against data at ten servers from DITL 2017 dataset, with data taken from April 11–13, 2017.

For each query q, we extract its query name and match it against a list of server names used by time providers we compiled using multiple sources (Table 1). We then compute the number of unique queries, clients, and autonomous systems (ASes) for each time provider (we use CAIDA's Routeviews Prefix2AS datasets to map IP addresses to ASes [12]).

Table 2 shows the DITL datasets after processing. In 2022, we identified 126 million queries from 491 thousand resolvers and 22 thousand ASes that queried for names matching the domain names in Table 1. We notice that the distribution varies per root letter as resolvers employ their own criteria to choose which root server to contact [56].

Srv.	Queries		Resolvers		ASes	
	2017	2022	2017	2022	2017	2022
A	29,178,992	30,088,926	197,721	117,175	913	10,747
B	7,449,043	357,4484	131,362	45,932	7,022	6,107
C	8,359,883	13,153,018	15,6942	89,692	8,517	9,506
D	410,6686	7,498,890	127,895	68,408	6,499	8,204
E*	5,693,446	144,861	152,961	1,065	8,108	219
F	2,361,662	3,692,906	44,032	17,083	4,366	2,817
G	NA	3,862,762	NA	48,307	NA	6,353
H	834,493	4,545,561	76,836	50,538	4,389	6,740
J	6,692,983	13,311,972	157,677	95,582	8,086	10,021
K	6,402,332	15,835,168	146,007	92,450	8,154	9,234
L	5,882,535	16,294,733	134,487	74,854	7,199	6,055
M	NA	14,200,343	NA	98,377	NA	9,187
Total	76,962,055	126,203,624	873,543	491,764	17,047	22,167

Table 2. DITL datasets: Matching queries per Root Servers (IPv4 and IPv6). *E-root 2022 datasets are incomplete. April 11–13, 2017 and April 12–14, 2022.

3.4 Comparing time services

Table 1 lists the 10 main and 137 other NTP services we consider, as well is the cache durations (TTLs) for each. We also include their top-level domain (TLD) TTL, such as .com and .net. To appear in DITL, both TLD and server name records must be expired from cache (DNS records are cached independently [54]).

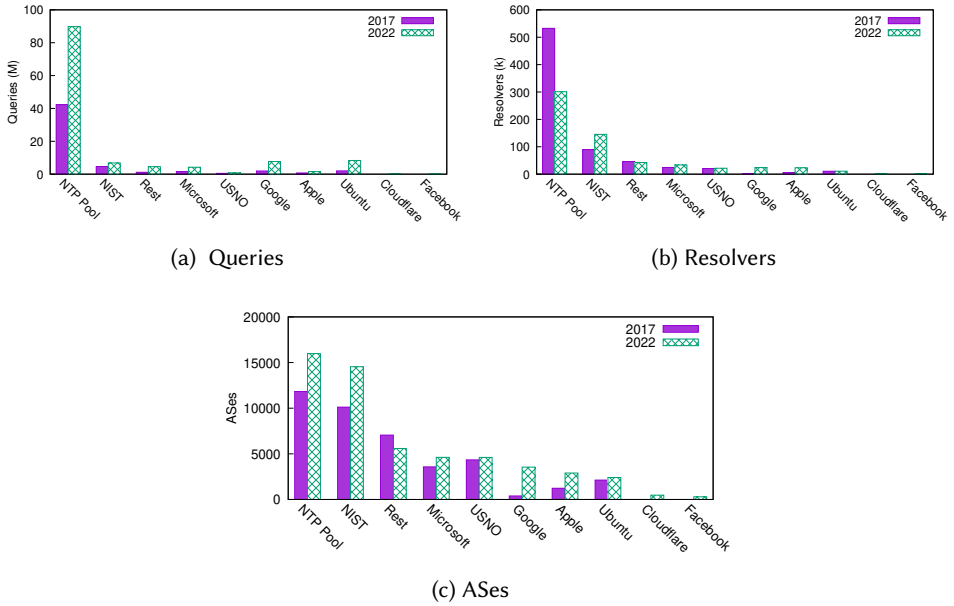
Figure 3a shows the query distribution per time provider from the DITL datasets. We see that NTP Pool receives roughly 90M out of 126M queries in total, being far more popular than all other time providers combined.

These query counts, however, may be inflated in favor of the NTP Pool servers, due to two main reasons: DNS time-to-live [51] values and multiple NTP Pool subzones.

TTL values associated with DNS records determine how long resolvers will cache the results. TTL values can range from 1 second to approximately 65 years [51], but many resolvers cap it at a maximum of two days, and many respect the TTL values [54, 55]. Operators are free to choose the TTL values that best suit their needs. Table 1 shows a list of time providers domains and their associated TTL values. In the case of the NTP Pool, the DNS records have a TTL of 150 seconds, which means that once a resolver obtains a response, all subsequent client queries within 150s are responded to from the cache instead of triggering new queries. These cache-hit queries we do not observe (§3.2).

To compensate for the differences in TTL values, we also compare time providers using two other metrics: the number of unique resolvers (source IP addresses) and the number of unique ASes associated with them. Both metrics are not inflated by repetitive queries due to low TTL values – they count one resolver or AS once per dataset, regardless of the number of queries sent. For both resolvers and ASes, we still see the same pattern: the NTP Pool is the most popular, followed by NIST (Figure 3b and Figure 3c).

A second reason queries may be artificially high is for pools that use multiple subdomains. In addition to the general pool.ntp.org, the NTP Pool has many subdomains that allow clients to make queries to servers by geography (such as europe.pool.ntp.org) or vendor (such as android.pool.ntp.org). As such, this large number of subdomains increases the query volume in compared to other providers that use one or few domain names.



(a) Queries (b) Resolvers

(c) ASes

Fig. 3. DITL results and top 10 time providers. Dates: April 11–13, 2017 and April 12–14, 2022.

To rule out the effect of multiple subdomains, we single out queries to pool.ntp.org only, which is the general NTP Pool zone (not used by vendors). In total, 25M queries from 138k resolvers and 10.2k ASes have queried for it. For comparison, NIST in 2022 had 6.8M queries from 145k resolvers and 14k ASes – thus more resolvers and ASes than the general NTP Pool zone.

Changes over time: when we compare the results from 2022 with 2017, we see that the NTP Pool was also the most popular service by any metric. We also note that newcomers did not have a significant impact in 2022 (Facebook and Cloudflare did not offer time services in 2017).

Despite the large cloud providers entering the market later, we observe that the NTP Pool is still the most queried service in terms of DNS traffic (at least a the Root level), followed by NIST. Even though Microsoft, Ubuntu, Google, and Apple set their devices to use their own time services, the NTP Pool attracts more resolvers and ASes, at least at the DNS level.

NTP Pool subzone use: We found 158 vendor zones and 104 geographical NTP Pool zones, using the DITL 2022 datasets. Vendor zones had 39.2M queries, while geographical zones had 25.3M queries. The general zone (pool.ntp.org) had 25.2M queries. We show them in §B.

NTP Traffic from one server: we run an NTP server as volunteers within the NTP Pool. Our NTP server [81] serves multiple regions and uses IP anycast. We have collected 24 hours traffic during Jun 22–23 2022 and observed 7.2B queries from 158M resolvers from 52,014 ASes globally [8]. For comparison, in 2016, NIST reported 16B daily queries [80]. Bear in mind that this is a single server out of the more than 4k listed at the NTP Pool.

4 CLIENT-TO-SERVER MAPPING

The NTP Pool utilizes `GeoDNS` for the mapping of clients to volunteer NTP servers (middle box in Figure 1). It could be argued that examining the `GeoDNS` source code alone should be adequate for comprehending the mapping criteria. While this is a valid point, it is important to note that

a code analysis alone cannot be applied to determine the specific servers assigned to real-world clients. This is because such analysis does not encompass the dynamic state of the NTP Pool, which is defined by the list of NTP servers and their performance metrics. These are used to derive the input files of `GeoDNS`, which are frequently changing.

Hence, it is crucial to consider the state of the NTP Pool, encompassing the list of volunteer servers, their configuration parameters, and their status. This comprehensive understanding can only be attained through active Internet measurements. Bearing this in mind, we perform two types of measurements: (a) in a real-world scenario, employing 9.2k vantage points (VPs) (§4.1), and (b) in a controlled environment (§4.2).

4.1 View from the wild

The NTP Pool operators list that 4.7k NTP Servers (2023-10-10). We seek to understand the logic between client/server mapping *and* its implications for real-world clients, given the population of NTP servers. A previous work [78] observed the client for a single VP in Germany was mapped to NTP servers located in Germany by the NTP Pool. However, it did not explore the reasons why and how.

To understand the NTP Pool mappings in practice, we set up two measurements (for IPv4 and IPv6) using 9.2 thousand VPs using RIPE Atlas probes [75, 76] (RIPE Atlas probes are hardware devices or virtual machines (VMs) that can be remotely instructed to carry out active measurements). In total, our VPs cover 3,082 ASes in 166 countries.

We configure these 9 thousand Atlas probes to send DNS queries to one of the NTP Pool authoritative servers (b.ntpns.org over IPv4 – 185.120.22.23), so we bypass DNS resolvers (Figure 1) and avoid hitting the resolver's cache. By passing resolvers, we can retrieve new NTP Pool addresses for every new query. The probes are configured to send queries every 5 min – a safe limit that does not overload RIPE Atlas and the NTP Pool authoritative DNS servers.

Table 3 shows the experiments' details. In the first experiment (EnumV4), we configure Atlas probes to retrieve IPv4 NTP servers, whereas in the second (EnumV6) we retrieve IPv6 NTP servers. For both experiments, we see ~9.2 thousand active VPs, having 9.1 thousand received valid responses (some VPs are blocked or contain bogus responses – a problem reported in Atlas probes in other works [53], which we disregard). These 9.1 thousand VPs provide us with a view from ~3 thousand ASes, totaling ~2.5 million DNS queries/responses per experiment.

For each experiment, each Atlas VP sends roughly 275 queries, receiving up to 4 NTP server addresses per response (Table 3). Theoretically, this would allow each probe to retrieve up to 1,100 unique NTP server addresses from the NTP Pool, if the process were completely random and if each client would not receive repeated NTP servers (we refrain from running an experiment that could span over all servers to avoid overloading RIPE Atlas).

Figure 4 shows the cumulative distribution function (CDF) of the number of unique IPs retrieved by each probe. We see a very different distribution of NTP servers per probe for both IPv4 and IPv6. Roughly 10% of the clients see up to 12 NTP servers (EnumV4) and 5 NTP servers (EnumV6). Considering that the NTP Pool has thousands of NTP servers, it is quite remarkable such a limited number of assigned servers. Next, we explain the reasons behind these differences.

4.2 Controlled experiment

We first *replicate* the NTP Pool authoritative DNS server setup and then *replay* the DNS queries from our experiments in the wild. By doing that, we can obtain the `GeoDNS` logs from our own instance and use this information to understand how it maps clients to servers.

`GeoDNS` (v. 3.0.2) takes as input a DNS zone file, that lists all zones in the pool (geographical and vendors) and their respective NTP servers. The `GeoDNS` source code does not have the actual zone

Measurement	EnumV4	EnumV6
Target	185.120.22.23	
QNAME	`2.pool.ntp.org`	
QType	A	AAAA
Date	2021-08-2[6-7]	2021-08-3[0-1]
Interval	5min	5min
Duration	24h	24h
VPs	9,260	9,272
valid resp.	9,113	9,127
no resp.	147	145
ASes	3,116	3,133
valid resp.	3,082	3,095
no resp.	156	148
Countries	166	168
Responses	2,534,199	2,583,318
Valid Responses	2,469,211	2,535,981
invalid/empty	64,988	47,337
NTP servers	3,056	1,479
Queries/VP	275	275

Table 3. RIPE Atlas experiments. Datasets: [73].

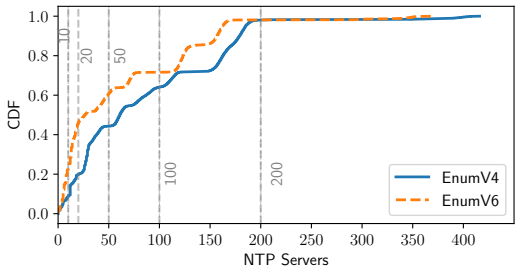


Fig. 4. CDF of NTP servers seen per Atlas VP.

file used by the NTP Pool, and we were not able to obtain them from the NTP Pool operators. It contains a demo zone file, which we use as starting point. We refrain from doing source code analysis given it does not include the zone files, which are a product of the NTP Pool monitoring systems and the networking conditions of each server. Therefore, we need to an empirical measurement to determine the status of the servers and understanding how the mapping works in practice.

4.2.1 Reversing the NTP Pool DNS zone. We resort to reverse engineering the NTP Pool zone files (sample in Appendix §C). We start by using the demo zone file available with the `GeoDNS` source code and populate it with servers that we have found with EnumV4 and EnumV6 experiments, in the following way:

(1) Generate a list of all NTP servers from EnumV4 and EnumV6 measurements
(2) Retrieve metadata (DNS zones) from each NTP server from the NTP Pool website
(3) Populate the demo zone file using the retrieved metadata

In the first step, we obtain 3,056 NTP server addresses. We then crawl each of them from the NTP Pool website using each their IP address. Each NTP server in the pool has a dedicated page (in

the form of https://www.ntppool.org/scores/IP), which lists the zones the NTP server is associated. For example, the NTP Pool page for 95.217.188.206 shows that this NTP server is allocated to the global (@), europe, and Finland's fi zones [64]. Then, we assigned this particular IP address to these subzones in our reverse-engineered zone file. We repeat this process for all 3,056 IPv4 and 1,479 IPv6 addresses we found from EnumV4 and EnumV6.

In the GeoDNS zone files, each NTP server has a *weight* associated with it, which is derived from how much service capacity the volunteer wants to donate to the pool (*BW* in Figure 1). In practice, servers with higher weight values are picked more often. For example, a server with a 100 weight will be seen 100 times more often than a server with one weight. We demonstrate the weights influence in Appendix §D.

Our reverse-engineered zone file has 126 non-empty zones in total – all country and continent zones (we found no vendor zones using this method). We found 125 zones for IPv4 and 112 for IPv6. We also found many countries (101 for IPv4, 145 for IPv6) that have zero servers in their zones.

4.2.2 Validation. The next step consists in replaying the DNS queries from the EnumV4 experiment on our controlled environment. We use the reverse-engineered zone file on GeoDNS and use Maxmind's GeoLite2 country IP2location database [47] from 2021-08-24 – a required input by GeoDNS to operate.

Client setup: To replay the queries from EnumV4, we send spoofed IP packets (forged source IP addresses [22]), using a customized Python script, and run our experiment on our server disconnected from the Internet – so our spoofed packets cause no harm.

Collected datasets: we collect two datasets, namely, network traces (pcap files), and GeoDNS log files (Listing 1, which lists the metadata associated with each DNS query and response), both from the same Linux server. We refer to this experiment as EnumV4-emul.

By analyzing GeoDNS log files, we see how the mapping occurs: first, the client's geographical information is retrieved from MaxMind's database (country and continent). These are used to populate a list of *candidate* zones that can be used to answer this client, which is shown by the Targets tag (Listing 1). Then, the tag LabelName shows which zone the client has been mapped. For this particular client, we see it could have gone to Israel (il), Asia, or the Global (@) zone, and it was ultimately mapped to Israel's zone. The logs do not show, however, which NTP servers were included in the DNS response. We analyzed the pcap files and confirmed they belong to the Israel's zone.

```
{ "Time": 1626941639825507800,
  "Origin": "2.pool.ntp.org.",
  "Name": "2.pool.ntp.org.",
  "Qtype": 1,
  "Rcode": 0,
  "Answers": 2,
  "Targets": ["il", "asia", "@"],
  "LabelName": "il",
  "RemoteAddr": "132.64.6.1",
  "ClientAddr": "132.64.6.1/32",
  "HasECS": false}
```

Listing 1. GeoDNS server log (v3.0.2)

Category		#Zones	#VPs
Equal	$S_{EnumV4_{emul}} = S_{EnumV4}$	93	2,265
More	$S_{EnumV4_{emul}} > S_{EnumV4}$	66	7,282
Fewer	$S_{EnumV4_{emul}} < S_{EnumV4}$	12	47

Table 4. Validation results per zone.

Results: For each VP (IP address from the Atlas in EnumV4), we compute two sets: S_{EnumV4} and $S_{EnumV4_{emul}}$, in which we list all NTP servers the VP has seen on each measurement – the experiment in the wild and our emulation. The latter we obtained from the pcap files. We then compare the sets for each VP.

Table 4 shows the results. We see three main categories: Equal shows that zones and VPs matched perfectly in the wild and our controlled experiments. These comprise 2.2 thousand VPs from 93 zones. The second category is More, in which the VPs in our controlled experiment saw more NTP

servers than those in the wild. These comprise most VPs (7.2k, 66 zones). We speculate this can be due to the use of uniform weights (1,000) in our emulation experiment, in which each server gets the same odds of being included in the response. In the NTP Pool, however, these weights vary by a factor of 2,000. As such, our Emulation retrieves most if not all servers in the zone, while in the wild, the distribution would have been shifted to servers with higher weights (see Appendix §D).

The last category is the more concerning one: 47 VPs in our controlled experiment saw *fewer* NTP servers than in our emulation experiments. We believe this may be due to two reasons: their DNS traffic being intercepted and ultimately to send to resolvers elsewhere, and dynamic changes in the NTP Pool NTP server population along our measurements. Next, we cover the second reason.

4.3 NTP Pool monitoring system

The second reason is that the NTP Pool continuously monitors the volunteer's NTP servers. Poorly performing servers (unreachable, providing incorrect time data) have points deducted up to a threshold and are evicted from the NTP Pool zone file if they cross this threshold (10 points). While evicting servers from zones should not change much our results, they change in a specific case: if a country zone has a single server and the server is evicted. If this happens, then the client will, from that point on, be mapped to its respective continent zone.

This case covers 34 VPs that see only one NTP server in our experiment hosted in Cameroon, Guernsey, and Reunion – the latter two islands belonging to the United Kingdom and France, respectively. These VPs are mapped to a single NTP server in their zone that was eventually evicted from the NTP Pool zone due to poor performance. This caused these VPs to fallback to its continent zone (Europe), which has many servers.

We demonstrate that with VP 17580 located in Guernsey. The EnumV4 experiment (in the wild) shows that this probe sees, in total, 21 unique NTP servers – even though its associated territory zone (`gg`) zone has only one NTP server. We plot the responses seen by this VP in the wild in Table 5. This VP initially receives a single NTP server in the DNS responses – 51.255.142.175 – an NTP server from Guernsey until 20:56. From 21:01 to 21:21, this VP receives 20 different NTP servers in 4 subsequent queries. These 20 servers belong to the `europe` zone, suggesting t this VP was mapped during this period to `europe` zone and not `gg` zone, which seem to have been empty.

While this shows that the probe sees more servers from Europe's zone, it does not show its country zone was empty at the same time. To show that, we analyze the scores associated with this NTP server from the NTP Pool website [63]) – a measurement carried independently from ours. We correlate our measurement results with the NTP Pool's server score logs, as seen in Figure 5. We see that this particular server score dropped below 10 – the minimum value for it to be used in the NTP Pool zones, otherwise a server is evicted – between 20:55:32 and 21:22:04 (2021-08-26, UTC). We show this in the gray area.

This period of scores lower than 10 coincides precisely when this VP receives 4 NTP servers per response (Table 5). Given these low scores, we can infer that this NTP server was likely evicted from the `gg` zone in this period. Since this was the *only* server in the zone, `GeoDNS` mapped this VP to its respective Continent zone (`europe`). Once the NTP server's score surpasses 10 again, after 21:22, we see the client again receiving 1 NTP server in the DNS response, likely from the NTP server joining the `gg` zone again.

New monitoring system: On March 2023, the NTP Pool operators released a new monitoring system, which consists of multiple measurement servers across the globe instead of a single one in California [69]. (Its beta version has been evaluated in [39]). Each NTP server is now evaluated by five monitoring servers instead of one, and the scores of the five are combined [68]. This prevents that failures on a single monitoring server wrongfully evicts well-performing servers. Whereas the scoring logic has changed, the eviction process and reinsert has not.

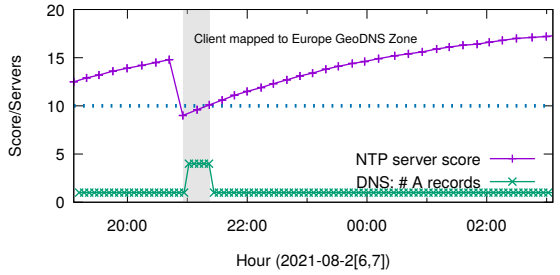


Fig. 5. NTP servers per DNS response from VP 17580 and server score from NTP Pool for server NTP server 51.255.142.175: low scores lead to NTP Pool eviction and fallback to the continent zone.

Time (UTC)	DNS responses (A records)
20:54:41	51.255.142.175
20:56:45	51.255.142.175
21:01:36	37.221.193.210, 149.156.70.60, 185.57.191.229, 94.16.114.254
21:06:49	213.239.234.28, 194.58.204.148, 95.215.175.2, 54.36.152.158
21:06:49	78.36.18.184, 138.201.16.22562.116.130.3, 212.83.158.83
21:16:38	49.12.125.53, 85.199.214.100, 85.236.36.4, 178.62.250.107
21:21:43	217.114.59.3, 217.114.59.66, 213.239.234.28, 130.208.87.151
21:26:47	51.255.142.175
21:31:38	51.255.142.175

Table 5. Atlas VP 17580 responses (2021-08-26)

Vendor zones: vendors zones (such as debian.pool.ntp.org) behave like the default zone: a client will be first mapped to its country of origin, and if its zone it is empty, to its continent. We show it in §E.

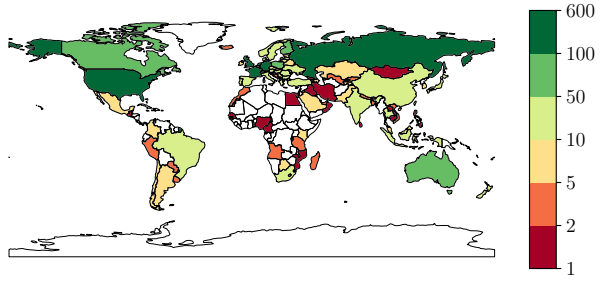
Bypassing `GeoDNS`: Users have the option to bypass `GeoDNS` mappings by adjusting their system settings to query their preferred zones. For instance, a user in Japan can query it.pool.ntp.org instead of the default pool.ntp.org to use Italian-only instead of Japanese NTP servers. However, this process requires change to the system settings. As a result, it is reasonable to assume that most users will not undertake this task. Instead, they are likely to rely on their default settings (most likely vendor zones), which by default, resort to `GeoDNS` mappings.

ECS support: We also found that if a resolver includes the client's EDNS client subnet (ECS) [15] in the DNS query, `GeoDNS` will then use the ECS IP address to map the client.

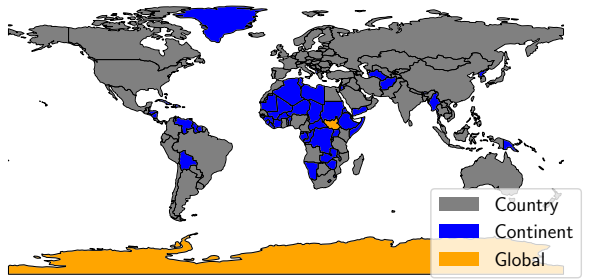
Takeaway: we have shown how `GeoDNS` maps clients to NTP servers. Clients are constrained by `GeoDNS` to NTP servers available in their respective country subzones. Clients in countries without NTP servers fallback to the continent or global zones.

5 PREDICTING MAPPINGS FOR ANY CLIENT IN THE WORLD

The experiments from §4 have demonstrated the process by which `GeoDNS` maps clients to NTP servers. However, these experiments are constrained to the clients for which we had vantage points, encompassing 166 countries in total (Table 3). Now that we understand this mapping process, we can expand our analysis to include all countries globally. This will allow us to assess the fairness and uniformity of the distribution of the 4k+ NTP servers among the entire client population.



(a) GeoDNS subzone sizes, in number of NTP servers.



(b) Client country and GeoDNS subzone mapping.

Fig. 6. GeoDNS mappings (IPv4).

5.1 Methodology

We first start with the analysis of all the GeoDNS DNS subzones we identified (§4.2). For each country/territory subzone, we show in Figure 6a the number of NTP servers that its respective GeoDNS subzone has (countries shown in white have zero NTP servers).

Next, we aim to determine the subzone to which the *client countries* will be assigned. As discussed in §4, if a country hosts NTP servers in its subzone (countries depicted in color in Figure 6a), then all its clients will be assigned to the corresponding country subzone (for instance, all Canadian clients will be assigned to ca.pool.ntp.org). This is shown in Figure 6b, represented in gray.

For the remaining countries, we need to determine if they will be mapped to either their continental or the global zone. To do that, we emulate VPs from these countries by identifying IP addresses from these countries and querying our local instance of GeoDNS in a controlled environment. Wy querying the Maxmind database for every /24 on the IPv4 space, we identify a total of 246 countries/territories. This is 80 more than our VPs as referenced in §4.1. For each of these countries/territories, we obtain an IP address.

Then, we send DNS queries using these IP addresses to our local GeoDNS instance and observe the mapping results (Figure 6b). We find that most countries with no NTP servers in their subzones are mapped to their respective continent zone. For instance, users in Morocco are mapped to africa.pool.ntp.org, which hosts 51 servers. Interestingly, we discover that 8 countries and territories, including South Sudan and Antarctica, are mapped to the global zone (highlighted in orange).

With the established mappings (client country → GeoDNS subzone), our next step is to ascertain the number of NTP servers available in each subzone. This will allow us to determine the fraction

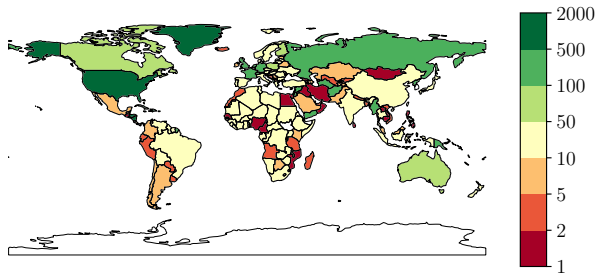


Fig. 7. Number of NTP servers assigned to each client country

of the 4k NTP servers that each client will be served by. To achieve this, we query the NTP Pool authoritative DNS servers 1000 times for each previously identified subzone. In total, we send 451k IPv4 queries and 337k IPv6 queries on 2022-11-24. Utilizing the mappings obtained from all countries/territories, we plot the clients' visibility, *i.e.*,, the *effective* number of NTP servers out of the 4k+ from the NTP Pool that serves all clients within a country.

To minimize the impact on the NTP Pool authoritative DNS servers, we select a random authoritative server before each query, thereby distributing the load among the 29 available servers, and we incorporate frequent pauses between measurements.

5.2 How fair is the NTP Pool mappings?

Figure 7 shows the number of NTP servers allocated to clients from each country. We observe a significant variation in the distribution of NTP servers among clients. For instance, clients in the US and Greenland are served by more than 500 servers, whereas clients in Egypt, Israel, Nigeria, and 24 other countries are served by only 2 NTP servers. Worse, Laos and Cameroon have a single NTP server allocated to it. Given that the NTP Pool comprises more than 4k NTP servers, we deem this skewed distribution as highly *skewed and unfair* for the client population.

This mapping scheme also *discourages* countries without NTP servers from contributing to the NTP Pool. It is more advantageous, in terms of diversity, to be mapped to their continental zone (with a larger number of servers) than to be mapped to a few local servers in their country. For instance, Peru, situated in South America, has 3 NTP servers in its zone, hence all its clients are mapped to them. In contrast, Bolivia has none and is therefore mapped to the South American zone, which consists of 50 servers. As a result, Bolivian clients have greater NTP server diversity than Peru, despite having no local NTP servers listed in the NTP Pool.

5.2.1 Time providers per country. Next, we calculate the number of time service providers for each client country. Considering that a single time provider may operate multiple servers, it is crucial to understand the implications of a time provider failure. To map NTP servers to providers, we utilize their associated AS number, retrieved from CAIDA's IP to AS maps [12]. Subsequently, for each client country, we compute the number of unique ASes serving it.

Figure 8 shows the number of time providers serving each client country. We observe that 27 countries, depicted in red, are served by a single AS (also shown in Table 6). Collectively, these countries account for 767M inhabitants and 465M Internet users. Countries in dark orange have all their NTP Pool clients served by only two time providers. These include Uruguay, Saudi Arabia, and Tanzania, encompassing 505M inhabitants and 288M Internet users.

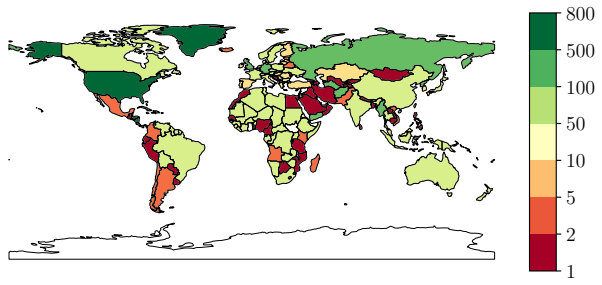


Fig. 8. Time providers (ASes) serving each country.

Bahrain	**Botswana**	Cambodia	Curaçao	Djibouti	Egypt	**Georgia**
Gibraltar	Guatemala	Haiti	**Iran**	Iraq	Israel	Kuwait
Laos	Lebanon	Macao	Mongolia	Mozambique	Nigeria	Oman
Panama	Philippines	Qatar	Rwanda	Senegal		

Table 6. Countries served by a single time provider: Cloudflare and other AS (bold)

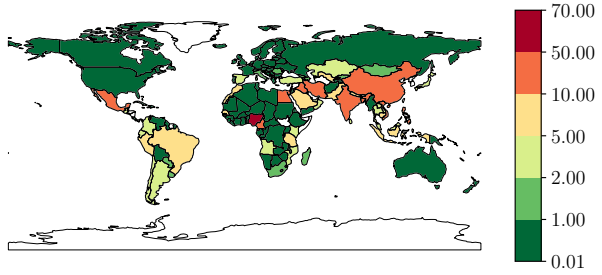


Fig. 9. Ratio of million Internet users per NTP server.

5.2.2 Users per NTP server. The final metric we employ to evaluate the fairness of the NTP Pool pertains to the number of Internet users assigned to NTP servers. This metric complements the previous ones by providing an estimate of the client population per NTP server. To compute the number of users per NTP server, we divide a country's Internet population (obtained from the ITU entry [33]) by the number of NTP servers that serve that zone (Figure 7).

Figure 9 displays the results, revealing a highly skewed distribution. Nigeria, with only two servers, averages 60M Internet users per server, followed by Egypt (40M) and Iran (34M). In stark contrast, Germany has 150k users per NTP server. These mappings result in a skewed distribution of users by NTP servers, with clients in both wealthier nations and countries without NTP servers in the NTP Pool receiving more NTP servers than the rest.

These time monopolies for entire countries poses security concerns. Ultimately, this concentration is an example of centralization and consolidation on the Internet [3–5, 35, 36, 52, 60, 79, 82]. In this case, the main drive is not large companies dominating a market – it is rather due to the current `GeoDNS` mapping algorithm. Next we discuss its security implications.

5.3 Security Implications

Despite the NTP Pool having over 4K+ NTP servers, these servers are distributed unevenly among the client population. This concentration of servers can be exploited by an attacker in two ways. Firstly, an attacker may aim to hijack all traffic from a specific country by hosting a single NTP server within it and adding it to the NTP Pool. We have demonstrated how a single NTP server from Guernsey serves as the sole provider for the entire region (§4.3). This type of attack was first demonstrated in [78, §6.2], where the authors added an IPv6 server to the NTP Pool. Although they attracted significant traffic with their method, they did not demonstrate a monopoly, as we have done. Notably, all countries in blue in Figure 6b, including Albania, Bolivia, Tunisia, Namibia, Venezuela, and Guatemala. This includes 101 countries and territories for IPv4 and 145 for IPv6, covering about 260M Internet users.

Another attack against the NTP Pool does not require monopolizing traffic; an attacker can simply control a *portion* of the traffic. The attack involves introducing numerous NTP servers into densely populated zones. This tactic aims to create a race condition, thereby increasing the likelihood of clients being served by their designated NTP servers (while maximizing the *BW* value, as shown in Figure 1). Particularly, in countries with fewer servers, the task of diverting a significant portion of the traffic becomes notably more manageable.

In both cases, an attacker can then execute time-shifting attacks on clients, which involve altering their clocks. In this scenario, an attacker can distinguish between real clients and NTP Pool monitoring servers by initially configuring their NTP server to operate in a "monitor only" mode. In a manner similar to previous studies such as [72] and [39], the attacker can provide accurate time readings to the NTP Pool's monitoring servers while deliberately providing incorrect time to other clients. This manipulation effectively circumvents the NTP Pool's eviction system [72].

6 GEODNS MAPPINGS: ARE THEY SOUND AND THEIR IMPLICATIONS

In §5, we demonstrated that `GeoDNS` employs a restrictive client-to-NTP server mapping approach, which imposes limitations on the number of NTP servers available for serving clients. We contacted the NTP Pool operators [7] and they argued the idea behind these mappings is twofold: reduce the risk of asymmetric routing and to minimize packet loss.

Asymmetric routing occurs when incoming and outgoing network traffic for a given connection follows different paths, can have a significant impact NTP and potentially lead to synchronization issues and time inaccuracies [26] (NTP assumes symmetric paths). It is questionable where keep clients within a country can reduce route assymetry. Recent work has shown that most Internet paths are asymmetrical [85], so it is not a NTP Pool only problem. Binding client to countries does not consider the large diversity in country sizes – a client in Belgium may be geographically and topologically closer to NTP servers located in neighboring Germany, while a client in Honolulu being served by a NTP server in Boston (both in the US but 8.2k km apart).

Packet loss can also impact clock synchronization, given NTP responses may simply not arrive. To determine whether these packet loss concerns are sound, we carry out active measurement experiments next.

6.1 Can far away NTP services provide good time information?

Is it possible for clients to receive accurate time service from servers located in distant regions, rather than being limited to restrictive in-country mappings? To examine this hypothesis, we undertake an experiment employing 132 RIPE Atlas probes as vantage points. These probes are drawn from 21 countries that are presently solely served by Cloudflare (highlighted in bold within Table 6). It's worth noting that the majority of these countries are situated in Africa, the Middle East,

Provider	Cloudflare	Africa	Asia	Europe	North Am.	South Am.
NTP Server	162.159.200.123	41.220.128.73	144.24.146.96	94.198.159.11	45.33.65.68	186.155.28.147
# Atlas Probes	132	132	132	132	132	132
Valid	131	130	130	130	130	90
Countries	21	21	21	21	21	21
Valid	21	21	21	21	21	16
Valid Queries	36,501	34,835	33,145	35,763	35,918	21,540
Avg. Offset (s)	1.96	1.97	1.78	1.97	2.03	1.66
Med. Offset (s)	0	0	0	0	0	0

Table 7. Evaluating NTP servers from clients located in clients only served by Cloudflare. Datasets: [74]

and Southeast Asia, as opposed to regions like the United States or Europe, where more favorable outcomes might be anticipated.

Our objective is to ascertain whether clients in these countries experience no significant packet loss among all servers. To establish a baseline, we compare the service offered by their current sole time provider, Cloudflare, with five additional NTP servers from the NTP Pool. We choose one server per continent, with our choice based on the NTP server that exhibits the highest frequency per zone, as shown in §5. We configure these Atlas VPs to conduct queries every 30 minutes over one week (Dec. 16–23, 2022). This extended observation period enhances our chances of identifying potential failures.

The details of our experiments are consolidated in Table 7. It provides an overview of the specific NTP servers for which we configured Atlas probes to sent NTP queries. Over the span of one week, we received a total of 33,000 to 36,000 queries per NTP server, with one notable exception being the NTP server located in South America. This server received 21,000 valid responses but from a reduced pool of only 90 probes.

Lack of NTP Responses: We compute, for each Atlas Probe (VP), the ratio of NTP queries that receive no response. It's important to note that RIPE Atlas does not provide specific reasons for this lack of responses; it could be due to timeouts, filtering, or other factors [37].

In Figure 10, we show a CDF of the Atlas VPs and their respective rate of unanswered queries. We see that *90% of our VPs have no queries loss* for the Cloudflare, Europe and North America servers, despite many of the VPs being in Africa, Asia, and Middle East. For the Asia NTP server, we see that 80% of the VPs have up to 10% unanswered queries. Only the South American server has not particularly good results: 40% of VPs have more than 50% of unanswered queries.

Even though we cannot determine the reasons why this South American server performed worse than the others for the same VPs, we see in Figure 11 that the same VPs that failed to retrieve responses from the South American server could retrieve responses from the other servers. In this figure, we show the ratio of non-responses per probe (each entry in x axis is a Atlas probe) and per time server (y axis). As such, we can disregard issues with specific Atlas probes, as they could retrieve responses from other servers. In practice, if these they run a NTP client, they would likely disregard the South American server (§2), so it would cause no harm.

We next set out to compute the quality of timing data provided by each server. For every NTP response, we extract its offset value, indicating the time difference in seconds between the local probe clock the time provided by the NTP server. To mitigate the effects of clock drifting, we exclusively consider results from probes whose clocks were synchronized within a maximum deviation of five minutes by the time our NTP queries were initiated. (Atlas probes are designed to synchronize their clocks approximately every three minutes [29]).

Subsequently, we consolidate all offset data points for each NTP server and compute their aggregate statistics. Our analysis reveals that the average offset for all NTP servers remains under

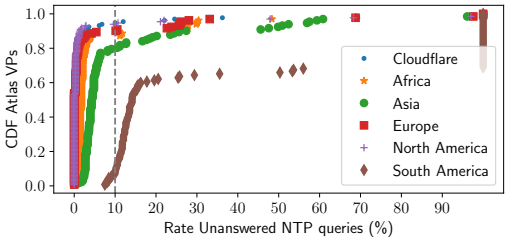


Fig. 10. Unanswered queries (%), per NTP server

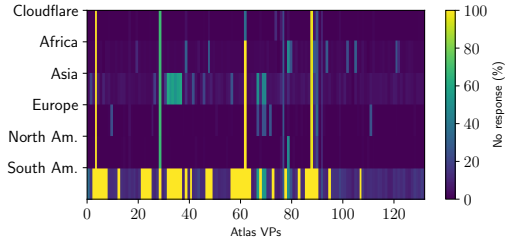


Fig. 11. Unanswered queries (%), per NTP server, for each Atlas VP.

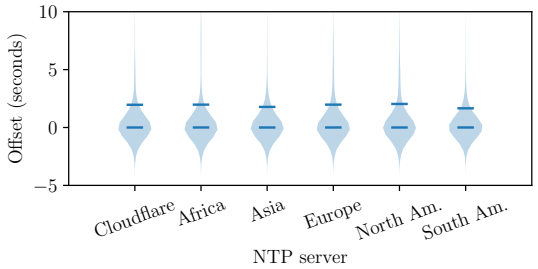


Fig. 12. Offset distribution, with bars showing average and median. We see that servers in other regions can provide similar quality service for these probes – all have similar offset values (<2 seconds).

2.1 seconds (Table 7), which is considered a favorable result. This finding is visually represented in Figure 12, where the majority of offsets fall within the range of ± 2 seconds for all probed servers, signifying reliable performance.

Takeaway: we have shown that NTP servers located in diverse geographical regions and continents can provide dependable time services for clients located elsewhere. Hence, this experiment serves to underscore that the assumption of tethering clients exclusively to servers within their own countries is unnecessary and opens up opportunities for potential attacks.

7 DISCUSSION

The NTP Pool is a time service provider that relies on the contributions of volunteer NTP server operators worldwide. It has been an active project for over 20 years. As shown in §3, it is and has been the most popular time service on the Internet, measured at the root DNS. It is time to recognize the NTP Pool as one of the most crucial timekeepers on the Internet.

We have also scrutinized `GeoDNS`, the NTP Pool's DNS component that determines which NTP servers will be assigned to clients. Prior to our work, there was anecdotal evidence that `GeoDNS` mapped clients to their own countries. We have demonstrated that the NTP Pool does not map individual users, but all users of entire countries (§4). Our findings reveal a very restrictive mapping: clients are bound to be served by the set of NTP servers in their country, even if there is only one server. We have shown the precise cases where countries fallback to their continent or global zones.

We also identified several issues with the current mapping algorithms. The most significant is that it produces a skewed and unfair server distribution among NTP clients worldwide. Users in wealthier countries, who can afford to donate NTP servers to the NTP Pool, are better off than users in less wealthy countries, in terms of NTP server diversity. In extreme cases, we have seen how the NTP Pool allows for the emergence of time monopolies, by mapping entire countries to

one or a few NTP servers/ASes (§5): it maps all clients in 27 countries, covering 767M inhabitants and 465M Internet users, to a single actor, which can have severe consequences in cases of attacks.

Our results reveal that the GeoDNS in-country mappings introduce unnecessary risks. Discussions with the NTP Pool operators [7] have been productive. They acknowledge the need for changes to enhance system security, which they plan to implement. Additionally, the use of DNS for load balancing across all volunteer servers must be considered when designing the new system. One potential solution could involve eliminating country zones in favor of larger continent zones.

8 ETHICS, PRIVACY, AND DISCLOSURE

Our paper has three ethical concerns: avoiding negative consequences of our measurements in both clients (Atlas VPs) and DNS/NTP servers, respecting the privacy of these VPs, and disclosing our findings to the NTP Pool operators.

Responsible experimentation: we design our experiments to minimize the impact on clients, measurement platform (RIPE Atlas), and measured DNS and Web servers. Whenever we use RIPE Atlas VPs, we use safe query rates (1 query per 5, 10, or 30 minutes, depending on the experiment). We also crawl web pages related to each NTP server on the NTP Pool website – fewer than 5 thousand pages. To minimize impact, we rate-limit our crawler to 1 webpage/second. Part of our experiments was done in an isolated network (§5), so no traffic was sent to the NTP Pool servers.

Privacy: We found cases of RIPE VPs that seem to use overseas DNS servers (which may be due to trying to bypass government censorship or DNS hijack). DNS hijack in RIPE VPs has been known for years [53, 83]. While we cannot determine which is the reason, we do not disclose details about these cases to protect these VPs and their owners, who volunteer to host them.

Disclosure to NTP Pool operators: We shared multiple versions of this manuscript with the NTP Pool operators, who provided valuable feedback. We also have exchanged e-mails and discussion on the public NTP Pool forum [7] . While we did not disclose any new attack models (they have been previously presented [39, 72]), we show how current GeoDNS mapping is restrictive and how measured its affected populations. We hope GeoDNS can be changed to introduce more diversity in the number of NTP servers each client sees.

9 RELATED WORK

NTP Pool measurement studies: our study is the most comprehensive evaluating the inner works and popularity of the NTP Pool. A previous study has also crawled the NTP Pool authoritative servers to enumerate them [78]. They used a single VP in Germany to query the NTP Pool authoritative servers. We scrutinize the inner works of GeoDNS and by unveiling how the NTP Pool monitors, evicts, and cleans its zone (§4), and show how clients all over the world see the NTP Pool (§5).

NTP Pool vulnerabilities: Previous studies have shown how the NTP Pool can be exploited to hijack traffic from countries with empty zones [78] – they run a brief experiment on IPv6 – but they do not confirm the traffic monopoly. Our measurements from §4.3 confirms it is feasible and demonstrate traffic monopoly, and we provide open datasets (by RIPE Atlas). Another study has shown that an attacker can also control traffic by introducing multiple NTP servers into densely populated zones [72]. The latter aims to create a race condition, thereby increasing the likelihood of clients being served by their malicious NTP servers to perform time-shift attacks.

Another work has identified vulnerabilities with the NTP Pool monitoring system [39]. The authors present multiple attack methods against the monitoring servers – which include BGP hijack and delay attacks. Another attack model they cover assumes an attacker controls one of the pool monitoring servers. While not directly related to ours, there are several other studies that focused on NTP security. They either cover the NTP protocol vulnerabilities [43–45], or show how NTP clients can be vulnerable to malicious time servers [18], or how NTP servers can be used in

distributed denial-of-service (DDoS) amplification attacks [16] (in which spoofed queries are sent with the source address of the target, which then receives unsolicited traffic), or study off-path attacks using DNS cache poisoning [34]. While related to ours, they do not focus on the NTP Pool itself, as we do.

With regards to NTP traffic characterization, previous studies have characterized traffic at the NIST's NTP servers [80] or running many NTP servers that are part of the NTP Pool [78]. We analyze Root DNS traffic to determine how popular time providers are and briefly cover 24 of traffic of a NTP server listed in the NTP Pool.

While NTP traffic is transmitted in clear (and thus prone to tampering), Network Time Security (NTS) [23] protocol provides client-server encryption and therefore eliminates the possibility of tampering between client and server. NTS, however, is currently not supported by the NTP Pool.

10 CONCLUSION

The NTP Pool has played a pivotal role in ensuring accurate timekeeping on the Internet. Operating as a community-driven initiative, akin to Wikipedia, it has proven to be the most widely used time service on the Internet, as demonstrated by our DITL datasets from the Root servers. We extend our gratitude to the volunteers who have dedicated their time and resources to support this endeavor over the past two decades.

In our investigation, we have delved into the intricate workings of the NTP Pool and highlighted an area of concern. While the client/NTP server mapping was implemented with good intentions (to avoid packet loss and traffic asymmetry), we have shown that these fears may be unfounded. Furthermore, these mappings can be improved to prevent attackers from exploiting current vulnerabilities to monopolize traffic, and to increase server diversity for clients worldwide.

Considering these findings, we raise awareness among NTP Pool operators regarding these shortcomings. We hope to encourage necessary improvements for the continued reliability and security of public and free time synchronization services on the Internet.

Acknowledgments

The authors express their gratitude to the anonymous SIGMETRICS reviewers for their insightful comments, and to our shepherd, Vishal Misra. We also extend our thanks to the various NTP Pool operators and community members who have provided us with valuable feedback.

The authors thank RIPE NCC for their Atlas measurement network and the Root Server Operators and DNS-OARC for the DITL datasets.

The work of Giovane C. M. Moura, Marco Davids, Caspar Schutijser, and Cristian Hesselman was carried out as part of the SIDN Labs funded project, `time.nl` (https://time.nl). Cristian Hesselman's work was also part of the Network Security program of the Twente University Centre for Cybersecurity Research (TUCCR, https://tuccr.nl).

John Heidemann's work on this paper is partially supported by NSF projects CNS-2212480 and CNS-2319409.

The work of Georgios Smaragdakis was partially funded by the European Research Council under starting grant ResolutioNet (679158), the Dutch 6G Future Network Services (FNS) project, and the European Union research and innovations programme Horizon Europe under grants SEPTON (101094901), MLSysOps (101092912), and TANGO (101070052).

REFERENCES

[1] Apple. 2021. Apple NTPService. time.apple.com.

[2] Roy Arends, Rob Austein, Matt Larson, Dan Massey, and Scott Rose. 2005. *DNS Security Introduction and Requirements.* RFC 4033. IETF. http://tools.ietf.org/rfc/rfc4033.txt

[3] Jari Arkko. 2019. Centralised Architectures in Internet Infrastructure. Internet Draft. https://tools.ietf.org/html/draft-arkko-arch-infrastructure-centralisation-00

[4] Jari Arkko. 2020. The influence of Internet architecture on centralised versus distributed Internet services. *Journal of Cyber Policy* 5, 1 (2020), 30–45. https://doi.org/10.1080/23738871.2020.1740753

[5] Arkko, Jari and Tramme, B. and Nottingham, M and Huitema, C and Thomson, M. and Tantsura, J. and ten Oever, N. 2019. Considerations on Internet Consolidation and the Internet Architecture. Internet Draft. https://tools.ietf.org/html/draft-arkko-iab-internet-consolidation-02

[6] Ask Bjørn Hansen. 2021. GeoDNS servers. https://github.com/abh/geodns/.

[7] Ask Bjørn Hansen. 2023. Minor New Features on the website. https://community.ntppool.org/t/minor-new-features-on-the-website/2947/8.

[8] Rushvanth Bhaskar. 2022. *A Day in the Life of NTP: Analysis of NTPPool Traffic.* Master's thesis. University of Twente and SIDN Labs, Enschede and Arnhem, The Netherlands. Master's thesis.

[9] Stephan Bortzmeyer, Ralph Dolmans, and Paul Hoffman. 2021. *DNS Query Name Minimisation to Improve Privacy.* RFC 9156. IETF. http://tools.ietf.org/rfc/rfc9156.txt

[10] Physikalisch Technische Bundesanstalt. 2022. FDCF77 - PTB.de. (Nov. 5 2022). https://www.ptb.de/cms/en/ptb/fachabteilungen/abt4/fb-44/ag-442/dissemination-of-legal-time/dcf77.html

[11] Randy Bush and Rob Austein. 2013. *The Resource Public Key Infrastructure (RPKI) to Router Protocol.* RFC 6810. IETF. http://tools.ietf.org/rfc/rfc6810.txt

[12] CAIDA. 2022. Index of /datasets/routing/routeviews-prefix2as. https://publicdata.caida.org/datasets/routing/routeviews-prefix2as.

[13] Sebastian Castro, Duane Wessels, Marina Fomenkov, and Kimberly Claffy. 2008. A Day at the Root of the Internet. *ACM Computer Communication Review* 38, 5 (April 2008), 41–46.

[14] Cloudflare. 2021. Cloudflare Time Service. https://www.cloudflare.com/time/.

[15] C. Contavalli, W. van der Gaast, D. Lawrence, and W. Kumari. 2016. *Client Subnet in DNS Queries.* RFC 7871. IETF. http://tools.ietf.org/rfc/rfc7871.txt

[16] Jakub Czyz, Michael Kallitsis, Manaf Gharaibeh, Christos Papadopoulos, Michael Bailey, and Manish Karir. 2014. Taming the 800 Pound Gorilla: The Rise and Decline of NTP DDoS Attacks. In *Proceedings of the 2014 ACM Conference on Internet Measurement Conference* (Vancouver, BC, Canada) *(IMC)*. ACM, 435–448. https://doi.org/10.1145/2663716.2663717

[17] Wouter B de Vries, Quirin Scheitle, Moritz Müller, Willem Toorop, Ralph Dolmans, and Roland van Rijswijk-Deij. 2019. A First Look at QNAME Minimization in the Domain Name System. In *International Conference on Passive and Active Network Measurement*. Springer, 147–160.

[18] Omer Deutsch, Neta Rozen Schiff, Danny Dolev, and Michael Schapira. 2018. Preventing (Network) Time Travel with Chronos.. In *NDSS*.

[19] Tim Dierks and Eric Rescorla. 2008. *The Transport Layer Security (TLS) Protocol Version 1.2.* RFC 5246. IETF. http://tools.ietf.org/rfc/rfc5246.txt

[20] DNS OARC. 2022. DITL Traces and Analysis. https://www.dns-oarc.net/index.php/oarc/data/ditl/.

[21] Ralph Droms. 1997. *Dynamic Host Configuration Protocol.* RFC 2131. IETF. http://tools.ietf.org/rfc/rfc2131.txt

[22] Toby Ehrenkranz and Jun Li. 2009. On the state of IP spoofing defense. *ACM Transactions on Internet Technology (TOIT)* 9, 2 (2009), 1–29.

[23] Daniel Franke, Dieter Sibold, Kristof Teichel, Marcus Dansarie, and Ragnar Sundblad. 2020. *Network Time Security for the Network Time Protocol.* RFC 8915. IETF. http://tools.ietf.org/rfc/rfc8915.txt

[24] Richard Gayraud and Benoit Lourdelet. 2010. *Network Time Protocol (NTP) Server Option for DHCPv6.* RFC 5908. IETF. http://tools.ietf.org/rfc/rfc5908.txt

[25] Google. 2021. Google Public NTP. https://developers.google.com/time.

[26] Mohammad Javad Hajikhani, Thomas Kunz, and Howard Schwartz. 2016. A Recursive Method for Clock Synchronization in Asymmetric Packet-Based Networks. *IEEE/ACM Transactions on Networking* 24, 4 (2016), 2332–2342. https://doi.org/10.1109/TNET.2015.2462772

[27] Stewart Hampton. 2018. Five Dangers of Poor Network Timekeeping + Easy and Cost Effective Solutions (Part 2 of 10). (Sept. 5 2018). https://www.microsemi.com/blog/2018/09/05/five-dangers-of-poor-network-timekeeping-easy-and-cost-effective-solutions-to-avoid-networks-fall-out-of-sync-part-2-of-10/

[28] Alden Hilton, Casey Deccio, and Jacob Davis. 2023. Fourteen Years in the Life: A Root Server's Perspective on DNS Resolver Security. In *32nd USENIX Security Symposium (USENIX Security 23)*. USENIX Association, Anaheim, CA, 3171–3186. https://www.usenix.org/conference/usenixsecurity23/presentation/hilton

[29] Philip Homburg. 2015. NTP Measurements with RIPE Atlas. https://labs.ripe.net/author/philip_homburg/ntp-measurements-with-ripe-atlas/.

[30] Nate Hopper. 2022. The Thorny Problem of Keeping the Internet's Time. *The New Yorker* (Sept. 30 2022). https://www.newyorker.com/tech/annals-of-technology/the-thorny-problem-of-keeping-the-internets-time

[31] IEEE. 2002. IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. *IEEE Std. 1588-2002* (2002). https://standards.ieee.org/ieee/1588/3140/

[32] IEEE. 2020. IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems. *IEEE Std 1588-2019 (Revision ofIEEE Std 1588-2008)* (2020), 1–499. https://doi.org/10.1109/IEEESTD.2020.9120376

[33] ITU. 2023. Statistics. https://www.itu.int/en/ITU-D/Statistics/Pages/stat/default.aspx

[34] Philipp Jeitner, Haya Shulman, and Michael Waidner. 2020. The Impact of DNS Insecurity on Time. In *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. 266–277. https://doi.org/10.1109/DSN48063.2020.00043

[35] Cecilia Kang and David McCabe. 2020. Lawmakers, United in Their Ire, Lash Out at Big Tech's Leaders. *New York Times* (July. 29 2020). https://www.nytimes.com/2020/07/29/technology/big-tech-hearing-apple-amazon-facebook-google.html

[36] Aqsa Kashaf, Vyas Sekar, and Yuvraj Agarwal. 2020. Analyzing Third Party Service Dependencies in Modern Web Services: Have We Learned from the Mirai-Dyn Incident?. In *Proceedings of the ACM Internet Measurement Conference* (Virtual Event, USA) *(IMC '20)*. Association for Computing Machinery, New York, NY, USA, 634–647.

[37] Robert Kisteleki. 2023. NTP empty results ('result': ['x': '*']). https://www.ripe.net/ripe/mail/archives/ripe-atlas/2023-October/005607.html.

[38] Warren Kumari and Paul Hoffman. 2020. *Running a Root Server Local to a Resolver*. RFC 8806. IETF. http://tools.ietf.org/rfc/rfc8806.txt

[39] Jonghoon Kwon, Jeonggyu Song, Junbeom Hur, and Adrian Perrig. 2023. Did the Shark Eat the Watchdog in the NTP Pool? Deceiving the NTP Pool's Monitoring System. In *30th USENIX Security Symposium*. https://www.usenix.org/conference/usenixsecurity23/presentation/kwon

[40] Leslie Lamport. 2019. *Time, Clocks, and the Ordering of Events in a Distributed System*. Association for Computing Machinery, New York, NY, USA, 179–196. https://doi.org/10.1145/3335772.3335934

[41] Ziqian Liu, Bradley Huffaker, Marina Fomenkov, Nevil Brownlee, and Kimberly Claffy. 2007. Two Days in the Life of the DNS Anycast Root Servers. In *Proceedings of the International conference on Passive and Active Measurements (PAM)*. 125–134.

[42] Jonathan Magnusson, Moritz Müller, Anna Brunstrom, and Tobias Pulls. 2023. A Second Look at DNS QNAME Minimization. In *Passive and Active Measurement: 24th International Conference, PAM 2023, Virtual Event, March 21–23, 2023, Proceedings*. Springer, 496–521.

[43] Aanchal Malhotra, Isaac E Cohen, Erik Brakke, and Sharon Goldberg. 2016. Attacking the Network Time Protocol. In *Proceedings of the 23rd Network and Distributed System Security Symposium (NDSS 2016)* (San Diego, California).

[44] Aanchal Malhotra and Sharon Goldberg. 2016. Attacking NTP's Authenticated Broadcast Mode. *SIGCOMM Comput. Commun. Rev.* 46, 2 (may 2016), 12–17.

[45] Aanchal Malhotra, Matthew Van Gundy, Mayank Varia, Haydn Kennedy, Jonathan Gardner, and Sharon Goldberg. 2017. The Security of NTP's Datagram Protocol. In *Financial Cryptography and Data Security: 21st International Conference, FC 2017, Sliema, Malta, April 3-7, 2017, Revised Selected Papers 21*. Springer, 405–423.

[46] Mark Morowczynski. 2012. Did Your Active Directory Domain Time Just Jump To The Year 2000? https://techcommunity.microsoft.com/t5/core-infrastructure-and-security/did-your-active-directory-domain-time-just-jump-to-the-year-2000/ba-p/255873.

[47] Maxmind. 2021. Maxmind. http://www.maxmind.com/

[48] Microsoft. 2021. Microsoft NTP Service. http://time.windows.com.

[49] David Mills. 2006. *Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI*. RFC 4330. IETF. http://tools.ietf.org/rfc/rfc4330.txt

[50] David Mills, Jim Martin, Jack Burbank, and William Kasch. 2010. *Network Time Protocol Version 4: Protocol and Algorithms Specification*. RFC 5905. IETF. http://tools.ietf.org/rfc/rfc5905.txt

[51] Paul Mockapetris. 1987. *Domain names - concepts and facilities*. RFC 1034. IETF. http://tools.ietf.org/rfc/rfc1034.txt

[52] Giovane C. M. Moura, Sebastian Castro, Wes Hardaker, Maarten Wullink, and Cristian Hesselman. 2020. Clouding up the Internet: How Centralized is DNS Traffic Becoming?. In *Proceedings of the ACM Internet Measurement Conference* (Virtual Event, USA) *(IMC '20)*. Association for Computing Machinery, New York, NY, USA, 42–49.

[53] Giovane C. M. Moura, Ricardo de O. Schmidt, John Heidemann, Wouter B. de Vries, Moritz Müller, Lan Wei, and Christian Hesselman. 2016. Anycast vs. DDoS: Evaluating the November 2015 Root DNS Event. In *Proceedings of the ACM Internet Measurement Conference*. ACM, Santa Monica, California, USA, 255–270. https://doi.org/10.1145/2987443.2987446

[54] Giovane C. M. Moura, John Heidemann, Ricardo de O. Schmidt, and Wes Hardaker. 2019. Cache Me If You Can: Effects of DNS Time-to-Live. In *Proceedings of the ACM Internet Measurement Conference*. ACM, Amsterdam, the Netherlands, 101–115. https://doi.org/10.1145/3355369.3355568

[55] Giovane C. M. Moura, John Heidemann, Moritz Müller, Ricardo de O. Schmidt, and Marco Davids. 2018. When the Dike Breaks: Dissecting DNS Defenses During DDoS. In *Proceedings of the ACM Internet Measurement Conference*.

ACM, Boston, MA, USA, 8–21. https://doi.org/10.1145/3278532.3278534
[56] Moritz Müller, Giovane C. M. Moura, Ricardo de O. Schmidt, and John Heidemann. 2017. Recursives in the Wild: Engineering Authoritative DNS Servers. In *Proceedings of the ACM Internet Measurement Conference*. ACM, London, UK, 489–495. https://doi.org/10.1145/3131365.3131366
[57] Network Time Foundation. 2022. Download NTP . https://doc.ntp.org/downloads/.
[58] Clifford Neuman, Tom Yu, Sam Hartman, and Kenneth Raeburn. 2005. *The Kerberos Network Authentication Service (V5).* RFC 4120. IETF. http://tools.ietf.org/rfc/rfc4120.txt
[59] NIST. 2022. NIST Internet Time Service (ITS). (Nov. 5 2022). https://www.nist.gov/pml/time-and-frequency-division/time-distribution/internet-time-service-its
[60] M. Nottingham. 2023. *Centralization, Decentralization, and Internet Standards.* RFC 9518. IETF. http://tools.ietf.org/rfc/rfc9518.txt
[61] NTP Pool. 2021. All Pool Servers. https://www.ntppool.org/zone.
[62] NTP Pool. 2021. Argentina — ar.pool.ntp.org. https://www.ntppool.org/zone/ar.
[63] NTP Pool. 2021. pool.ntp.org: statistics for 51.255.142.175 . https://www.ntppool.org/scores/51.255.142.175/.
[64] NTP Pool. 2021. pool.ntp.org: Statistics for 95.217.188.206. https://www.ntppool.org/scores/95.217.188.206.
[65] NTP Pool. 2021. pool.ntp.org: the internet cluster of ntp servers. https://www.ntppool.org/en/.
[66] NTP Pool. 2021. The NTP Pool for vendors. https://www.ntppool.org/en/vendors.html.
[67] NTP Pool. 2022. How do I join pool.ntp.org? https://www.ntppool.org/en/join.html.
[68] NTP Pool. 2023. Monitoring System - Technical details. https://news.ntppool.org/docs/monitoring/.
[69] NTP Pool. 2023. NTP Pool Monitoring v2. https://news.ntppool.org/2023/03/ntp-pool-monitoring-v2/.
[70] Oleg Obleukhov. 2020. Building a more accurate time service at Facebook scale. https://engineering.fb.com/2020/03/18/production-engineering/ntp-service/.
[71] United States Naval Observatory. 2022. Information about NTP, the time backbone of the Internet. (Nov. 5 2022). https://www.cnmoc.usff.navy.mil/Our-Commands/United-States-Naval-Observatory/Precise-Time-Department/Network-Time-Protocol-NTP/
[72] Yarin Perry, Neta Rozen-Schiff, and Michael Schapira. 2021. A Devil of a Time: How Vulnerable is NTP to Malicious Timeservers?. In *Proceedings of the 28th Network and Distributed System Security Symposium (NDSS 2021)* (Virtual Conference).
[73] RIPE NCC. 2021. RIPE Atlas Measurement IDS. https://atlas.ripe.net/measurements/ID. , where ID is the experiment ID: EnumV4: 32025718, EnumV6: 32058440, ArgV4: 31789516, ArgV4-Emul:31830680, ArgV4-Android: 31992051, DE-Android:31970486, ArgV6:32001506.
[74] RIPE NCC. 2023. RIPE Atlas Measurement IDS. https://atlas.ripe.net/measurements/ID. , where ID is the experiment ID: Cloudflare: 47865355, Africa: 47867480, Asia:47867358, Europe: 47867632, North America:47867336, South America:47867316:.
[75] RIPE NCC Staff. 2015. RIPE Atlas: A Global Internet Measurement Network. *Internet Protocol Journal (IPJ)* 18, 3 (Sep 2015), 2–26.
[76] RIPE Network Coordination Centre. 2020. RIPE Atlas. https://atlas.ripe.net.
[77] Root Server Operators. 2021. Root DNS. http://root-servers.org/.
[78] Teemu Rytilahti, Dennis Tatang, Janosch Köpper, and Thorsten Holz. 2018. Masters of Time: An Overview of the NTP Ecosystem. In *2018 IEEE European Symposium on Security and Privacy (EuroS P)*. 122–136. https://doi.org/10.1109/EuroSP.2018.00017
[79] Bruce Schneier. 2018. Censorship in the Age of Large Cloud Providers. https://www.schneier.com/essays/archives/2018/06/censorship_in_the_ag.html
[80] Jeff A. Sherman and Judah Levine. 2016. Usage Analysis of the NIST Internet Time Service. *Journal of Research of the National Institute of Standards and Technology* 121 (March 2016), 33. https://doi.org/10.6028/jres.121.003
[81] SIDN Labs. 2024. TimeNL. https://time.nl/index_en.html.
[82] Internet Society. 2019. Consolidation in the Internet Economy. https://future.internetsociety.org/2019/
[83] Stéphane Bortzmeyer. 2015. DNS Censorship (DNS Lies) As Seen By RIPE Atlas. https://labs.ripe.net/author/stephane_bortzmeyer/dns-censorship-dns-lies-as-seen-by-ripe-atlas/.
[84] Ubuntu. 2023. Ubuntu NTP Service. https://ubuntu.com/server/docs/network-ntp.
[85] Kevin Vermeulen, Ege Gurmericliler, Italo Cunha, David Choffnes, and Ethan Katz-Bassett. 2022. Internet Scale Reverse Traceroute. In *Proceedings of the 22nd ACM Internet Measurement Conference* (Nice, France) *(IMC '22)*. Association for Computing Machinery, New York, NY, USA, 694–715. https://doi.org/10.1145/3517745.3561422
[86] Adrian von Bidder. 2003. ntp DNS round robin experiment. https://groups.google.com/g/comp.protocols.time.ntp/c/cShrN7imCJ0.

A LIST OF TIME PROVIDERS AND SERVERS

Table 9 shows the list of time providers and their respective time services domain names used in §3. We built this list based on a public repository on Github[1].

Provider	Servers
Apple	time.apple.com, time.asia.apple.com, time.euro.apple.com, time1.apple.com, time2.apple.com, time3.apple.com, time4.apple.com, time5.apple.com, time6.apple.com, time7.apple.com
Cloudflare	time.cloudflare.com
Facebook	time.facebook.com, time1.facebook.com, time2.facebook.com, time3.facebook.com, time4.facebook.com, time5.facebook.com
Google	time.android.com, time.google.com, time1.google.com, time2.google.com, time3.google.com, time4.google.com
Microsoft	time.windows.com
NIST	time-a-b.nist.gov, time-a-g.nist.gov, time-a-wwv.nist.gov, time-b-b.nist.gov, time-b-g.nist.gov, time-b-wwv.nist.gov, time-c-b.nist.gov, time-c-g.nist.gov, time-c-wwv.nist.gov, time-d-b.nist.gov, time-d-g.nist.gov, time-d-wwv.nist.gov, time-e-b.nist.gov, time.nist.gov, utcnist.colorado.edu., utcnist2.colorado.edu
NTP Pool	pool.ntp.org, *.pool.ntp.org
Rest	asynchronos.iiss.at, chime1.surfnet.nl, clepsydra.dec.com, clepsydra.hpl.hp.com, clepsydra.labs.hp.com, clock.isc.org, clock.nyc.he.net, clock.sjc.he.net, clock.uregina.ca, cronos.cenam.mx, gbg1.ntp.se, gbg2.ntp.se, gnomon.cc.columbia.edu, gps.layer42.net, hora.roa.es, minuto.roa.es, mizbeaver.udel.edu, mmo1.ntp.se, mmo2.ntp.se, navobs1.gatech.edu, navobs1.oar.net, navobs1.wustl.edu, nist1.symmetricom.com, now.okstate.edu, ntp-ca.stygium.net, ntp-galway.hea.net, ntp-s1.cise.ufl.edu, ntp.atomki.mta.hu, ntp.colby.edu, ntp.dianacht.de, ntp.fiord.ru, ntp.fizyka.umk.pl, ntp.gsu.edu, ntp.i2t.ehu.eus, ntp.ix.ru, ntp.lcf.mx, ntp.mobatime.ru, ntp.nat.ms, ntp.neel.ch, ntp.neu.edu.cn, ntp.nic.cz, ntp.nict.jp, ntp.nsu.ru, ntp.ntsc.ac.cn, ntp.qix.ca, ntp.ripe.net, ntp.rsu.edu.ru, ntp.se, ntp.shoa.cl, ntp.time.in.ua, ntp.time.nl, ntp.vsl.nl, ntp.yycix.ca, ntp0.as34288.net, ntp0.nl.uu.net, ntp1.as34288.net, ntp1.fau.de, ntp1.hetzner.de, ntp1.inrim.it, ntp1.jst.mfeed.ad.jp, ntp1.net.berkeley.edu, ntp1.niiftri.irkutsk.ru, ntp1.nl.uu.net, ntp1.oma.be, ntp1.ona.org, ntp1.qix.ca, ntp1.stratum1.ru, ntp1.time.nl, ntp1.usv.ro, ntp1.vniiftri.ru, ntp2.fau.de, ntp2.hetzner.de, ntp2.inrim.it, ntp2.jst.mfeed.ad.jp, ntp2.net.berkeley.edu, ntp2.niiftri.irkutsk.ru, ntp2.oma.be, ntp2.qix.ca, ntp2.stratum1.ru, ntp2.stratum2.ru, ntp2.time.in.ua, ntp2.time.nl, ntp2.vniiftri.ru, ntp21.vniiftri.ru, ntp3.hetzner.de, ntp3.jst.mfeed.ad.jp, ntp3.stratum1.ru, ntp3.stratum2.ru, ntp3.time.in.ua, ntp3.usv.ro, ntp3.vniiftri.ru, ntp4.stratum1.ru, ntp4.stratum2.ru, ntp4.vniiftri.ru, ntp5.stratum1.ru, ntp5.stratum2.ru, ntps1-0.cs.tu-berlin.de, ntps1-0.uni-erlangen.de, ntps1-1.cs.tu-berlin.de, ntps1-1.uni-erlangen.de, ntps1.pads.ufrj.br, ntpstm.netbone-digital.com, otc1.psu.edu, ptbtime1.ptb.de, ptbtime2.ptb.de, rackety.udel.edu, rustime01.rus.uni-stuttgart.de, rustime02.rus.uni-stuttgart.de, sesku.planeacion.net, sth1.ntp.se, sth2.ntp.se, stratum1.net, svl1.ntp.se, svl2.ntp.se, t2.timegps.net, tempus1.gum.gov.pl, tempus2.gum.gov.pl, tick.usask.ca, time-a.as43289.net, time-b.as43289.net, time-c.as43289.net, time.esa.int, time.fu-berlin.de, time.nrc.ca, time.ufe.cz, time1.esa.int, time1.stupi.se, timehost.lysator.liu.se, timekeeper.isi.edu, tock.usask.ca, ts1.aco.net, ts2.aco.net, vniiftri.khv.ru, vniiftri2.khv.ru, x.ns.gin.ntt.net, y.ns.gin.ntt.net, zeit.fu-berlin.de
US Naval Observatory	ntp2.usno.navy.mil, tick.usno.navy.mil, tock.usno.navy.mil
Ubuntu	ntp.ubuntu.com

Table 9. List of Time Providers and their respective server names

B NTP POOL SUBZONES FOUND IN THE DITL DATASETS

In this section, we show the NTP Pool subzones found in the DITL 2022 datasets. We only consider *valid zones*, *i.e.*,, every subzone we find we resolve to determine if it exists or not (by resolving them using DNS A queries), so we can disregard zones such as debiaan.pool.ntp.org, which show up in the dataset due to typos.

Table 10 shows the breakdown of queries, resolvers, and ASes per subzone type: vendor (such as ubuntu.pool.ntp.org), geographical (such as spain.pool.ntp.org), and general, which is the pool.ntp.org zone. For both vendor and geographical zones, we convert [1--3].ZONE.pool.ntp.org to ZONE.pool.ntp.org. We see that most queries are for vendor zones, but most resolvers and ASes query for the general zone.

Type	queries	resolvers	ASes
vendor	39285892	96263	8657
geographical	25315204	113417	8003
general (pool.ntp.org)	25201533	138075	10262

Table 10. NTP Pool subzones categories observed on the DITL 2022 datasets, and query counts.

[1] https://gist.github.com/mutin-sa/eea1c396b1e610a2da1e5550d94b0453

Table 12 shows the vendor zones we have observed on the DITL dataset.

Rank	zone	queries	resolvers	ases	Rank	zone	queries	resolvers	ases
1	debian	19082520	24195	3038	80	xxter	166	36	26
2	centos	5730288	10174	2260	81	americantime	159	31	22
3	ubuntu	3205854	5801	1436	82	angstrom	150	1	1
4	android	2032889	21952	3026	83	cambridge-audio	141	13	9
5	ubnt	1783505	11778	2058	84	kodakalaris	123	8	6
6	rhel	1435650	2823	671	85	qumulo	115	5	3
7	openwrt	1427134	3226	771	86	avaya	111	27	20
8	shor	1281320	1073	336	87	planefinder	100	21	12
9	datadog	526589	5548	1223	88	github	93	4	3
10	lede	472461	314	139	89	viking	89	42	31
11	freebsd	368902	462	276	90	sinefa	72	4	4
12	pfsense	275785	7550	1651	91	comrex	67	11	10
13	opensuse	252256	431	223	92	scientific	66	12	8
14	amazon	159283	1723	589	93	dragonmint	61	19	8
15	vmware	120825	63	49	94	crosspoint	52	2	2
16	camlin	118057	2	1	95	datataker	45	4	2
17	control4	116320	125	56	96	whirlpool	38	7	6
18	ciscosb	112791	1828	756	97	nti1	36	22	15
19	inovonicsinc	95190	23	22	98	netcomm	33	10	9
20	ipfire	74277	120	49	99	sailfishos	32	4	4
21	tandberg	70758	82	62	100	rbsh	31	11	7
22	suse	66246	276	153	101	patton	31	10	9
23	sourcefire	65596	171	108	102	idigi	30	11	3
24	ciscome	54937	160	93	103	askozia	26	16	11
25	bose	35259	236	126	104	cham	25	10	5
26	irobot	28747	662	254	105	flatcar	21	13	7
27	opnsense	22951	1728	562	106	anki	20	11	9
28	aastra	20996	378	197	107	homewizard	19	5	4
29	sonostime	19251	521	204	108	barix	19	9	7
30	nettime	17162	2355	893	109	daktronics	16	8	8
31	fedora	17002	490	276	110	collax	16	9	5
32	smeserver	16138	435	187	111	nubolabs	15	4	3
33	savantsystems	15988	36	19	112	novell	13	1	1
34	bdrthermea	15184	109	53	113	nodemcu	13	6	3
35	aerohive	14054	426	219	114	digitallumens	13	5	3
36	pepwave	11654	129	57	115	rwesmarthome	12	6	4
37	cloudgenix	11305	1679	349	116	catapult	12	4	1
38	gentoo	9896	128	89	117	aolt	12	9	7
39	exigent	9047	15	1	118	riverbed	11	4	4
40	barracuda	8552	102	64	119	rbtt	11	4	3
41	wled	7677	10	8	120	raumfeld	10	3	3
42	xenserver	7155	16	13	121	solus	9	6	4
43	sophos	6824	133	72	122	meteocontrol	7	4	2
44	arch	6440	267	137	123	freetalk	7	2	1
45	mitel	5255	67	38	124	progress	6	2	2
46	colubris	5162	85	65	125	m0n0wall	6	4	3
47	resinio	5050	42	29	126	foobar	6	2	2
48	boot2docker	4514	20	15	127	eyesaas	6	4	3
49	logitech	2871	152	79	128	piecesint	5	1	1
50	schneider	2422	12	9	129	dovado	5	4	2
51	ovcirrus	2208	61	38	130	cctv	5	5	5
52	peplink	2143	68	40	131	yoctopuce	4	4	4
53	vyatta	2023	6	4	132	smartos	4	4	3
54	siemens	1889	279	43	133	openmandriva	4	2	1
55	servertech	1830	38	29	134	ocedo	4	2	2
56	sapphire	1769	3	3	135	guix	4	1	1
57	manjaro	1580	80	47	136	gtantp	4	3	3
58	rgnets	1553	62	32	137	clearlinux	4	2	2
59	cumulusnetworks	1416	7	7	138	axsguard	4	1	1
60	vizio	1167	92	51	139	unicoi	3	3	3
61	axis	1127	60	40	140	irrigationcaddy	3	2	2
62	tradfri	1080	7	5	141	digitalstrom	3	1	1
63	kerio	1067	43	31	142	anetd	3	3	3
64	formlabs	994	73	46	143	uwclub	2	1	1
65	cloudlinux	977	147	89	144	trendcontrols	2	2	2
66	digium	520	158	98	145	tosibox	2	1	1
67	intra2net	512	102	35	146	teco	2	1	1
68	openembedded	465	23	17	147	ricohucs	2	1	1
69	ubiquita	429	17	15	148	grandcentrix	2	1	1
70	zscaler	425	19	14	149	doccirrus	2	1	1
71	ooma	407	43	27	150	conceptronic	2	1	1
72	nixos	348	28	21	151	bigswitch	2	1	1
73	endian	341	22	16	152	wahsega	1	1	1

Rank	zone	queries	resolvers	ases	Rank	zone	queries	resolvers	ases
1	us	10702754	21277	3036	53	gr	1430	255	43
2	cn	4967144	21543	2523	54	ir	1392	99	46
3	asia	2785155	15477	1989	55	no	1391	142	58
4	north-america	778743	29346	1859	56	lu	1366	29	16
5	europe	735803	9657	1876	57	gb	1316	54	37
6	tw	590657	1862	283	58	kz	1033	11	8
7	ru	524261	3656	1198	59	uy	866	115	27
8	id	473657	7808	804	60	is	808	3	2
9	sg	472444	6333	948	61	my	586	92	35
10	de	376184	7684	1089	62	sk	513	248	32
11	fr	371864	1645	371	63	by	383	17	10
12	au	323778	3092	870	64	cl	346	197	43
13	uk	287959	6299	840	65	ec	314	24	19
14	nz	254326	2086	668	66	lv	306	13	7
15	south-america	156831	5328	801	67	ie	261	102	48
16	ca	156164	939	278	68	uz	225	8	6
17	oceania	139595	1042	360	69	ae	187	50	32
18	jp	123247	2431	366	70	cr	182	32	23
19	ch	105990	1256	288	71	rs	164	31	16
20	nl	101761	1576	399	72	qa	133	57	15
21	hk	93671	1788	279	73	lt	95	24	14
22	pl	92878	278	140	74	ke	75	9	5
23	it	85431	889	277	75	bd	71	5	5
24	in	78420	673	286	76	lk	62	14	8
25	br	53473	596	271	77	py	54	9	8
26	africa	48349	877	350	78	tj	51	4	3
27	za	42968	192	91	79	co	48	38	18
28	th	41391	165	51	80	jm	37	6	4
29	pt	38502	1578	227	81	dz	32	1	1
30	ua	35589	351	197	82	kg	30	13	5
31	kr	31511	1446	323	83	ba	25	11	6
32	at	26761	613	164	84	mu	16	2	2
33	vn	24302	75	30	85	bh	16	2	1
34	be	24005	342	73	86	cy	12	9	7
35	mx	21933	207	97	87	zw	11	3	3
36	fi	21531	151	54	88	al	11	7	6
37	ps	17197	1	1	89	cm	8	4	3
38	dk	14063	176	71	90	am	8	3	3
39	si	11281	29	17	91	nc	7	5	5
40	ar	9139	162	74	92	md	6	5	4
41	hr	9076	26	14	93	np	5	5	3
42	se	7148	200	87	94	ge	4	2	2
43	il	6500	163	93	95	ao	4	4	3
44	ee	6370	31	19	96	mg	3	2	2
45	es	6296	312	108	97	ma	3	3	3
46	sa	5879	69	39	98	li	3	3	2
47	cz	4937	140	66	99	do	3	3	2
48	ro	4931	194	71	100	bw	3	2	2
49	tr	2857	232	64	101	mk	2	2	2
50	bg	2547	92	49	102	ni	1	1	1
51	pk	2377	11	9	103	kh	1	1	1
52	hu	1664	188	46	104	gh	1	1	1

Table 13. Geographical NTP Pool zones found in the DITL 2022 datasets

Rank	zone	queries	resolvers	ases	Rank	zone	queries	resolvers	ases
74	purestorage	304	15	11	153	vasco	1	1	1
75	bctelectronic	269	9	6	154	pexip	1	1	1
76	ipaccess	243	5	5	155	itcloud	1	1	1
77	lenbrook	200	25	15	156	inovonics	1	1	1
78	vornexl	193	19	15	157	fireeye	1	1	1
79	coreos	168	9	9	158	echo360	1	1	1

Table 12. Vendor NTP Pool zones found in the DITL 2022 dataset

Table 13 show the geographical zones found on the DITL datasets from 2022, and their queries, resolvers, and ASes. We rank them according to the number of queries.

C SAMPLE GEODNS ZONE FILE

Listing 2 shows a GeoDNS sample DNS zone file. The GeoDNS zone file has multiple DNS subzones, like Turkey's tr (tr.pool.ntp.org). Each subzone has a list of IPv4 and IPv6 addresses (A and AAAA records), which lists NTP servers available to that particular country – and clients from these countries will see the A/AAAA records showed in this subzones[2]. Each A/AAAA records is followed by a *weight*, which is a non-standard DNS feature used by GeoDNS to sort the frequency in which records should be returned to clients, a method that allow NTP Pool volunteers to set indirectly the amount of traffic they want to receive at their NTP servers.

In Listing 2 example, the server 203.17.251.1 is likely to appear 100x more often in responses than 149.255.99.71 (calculated by the ratio between their weights).

```
{
   "ttl": 390,                % DNS TTL
   "serial": 1345449135,      % DNS Zone file serial number
   "data": {
      "": {                   % Empty string indicates the "global" pool
         "ns": [              % Authoritative DNS servers
            "a.ntpns.org",
            "b.ntpns.org",
            "x.example.com"
         ],
         "a": [                % IPv4 addresses of all NTP servers in the global zone
            [
               "203.17.251.1", % IPv4
               "1000"          % Weight
            ],
            % Additional IPv4 entries...
         ]
      },
      "tr": {                 % Subzone: tr.pool.ntp.org
         "a": [               % IPv4 addresses for the subzone
            [
               "77.243.184.65",
               "1000000"
            ],
            [
               "212.175.18.126",
               "100000"
            ],
            % Additional IPv4 entries...
         ]
      }
   }
}
```

Listing 2. GeoDNS demo zone file for pool.ntp.org

D WEIGHTS VALIDATION

Next, we evaluate how each NTP server weight determines the distribution of NTP servers among clients when weight is considered. To do that, we carry out two experiments: a baseline experiment measured in the wild (using the NTP Pool DNS servers), which we compared against a controlled emulation in our setup, , which we use our own GeoDNS instance, as in §4.2.

In the baseline experiment, we query the authoritative server of one of its country subzones – Argentina's ar. We choose it because it has only eight active IPv4 NTP servers (on 2021-08-02 [62]), reducing the number of necessary queries to evaluate the weight's influence. By directly querying Argentina's 3.ar.pool.ntp.org, we *bypass* GeoDNS's geolocation steps, obtaining records only listed in the ar subzone. (We confirm this behavior experimentally by running a test locally).

[2]Traditional authoritative DNS server use standardized text zone file formats [51], but GeoDNS uses JSON zone files instead [6].

Measurement	ArgV4	ArgV4-Emul
Target	185.20.22.23	54.93.163.251
QNAME	`3.ar.pool.ntp.org`	`wilson.ants`
QType	A	A
Date	2021-08-02	2021-08-06
Interval	10min	10min
Duration	2h	2h
VPs	9219	9229
valid resp.	9068	9052
no resp.	783	382
ASes	3127	3128
valid resp.	3080	3067
no resp.	474	262
Countries	1	1
Responses	107031	110292
Valid Responses	104331	107793
invalid/empty	2700	2499
NTP servers	8	8
per response (median)	2	2
per response (q1)	2	2
per response (q3)	2	2
Queries/VP	11.6	11.9

Table 14. NTP Pool RIPE Atlas experiments with weights validation. Datasets: [73].

		ArgV4		ArgV4-Emul	
IP	**ASN**	**Counts**	**Ratio**	**Counts**	**Ratio**
162.159.200.1	13335-Cloudflare	37580	18.3%	37504	17.7%
168.96.251.227	3597-InnovaRed	31142	15.2%	31763	15.1%
170.210.222.10	4270-Red de Inter.	28599	13.9%	29288	13.9%
168.96.251.226	3597-InnovaRed	25707	12.5%	26737	12.7%
181.93.10.58	7303-TelecomArg	24878	12.1%	25836	12.2%
168.96.251.195	3597-InnovaRed	24731	12.1%	25812	12.2%
168.96.251.197	3597-InnovaRed	17223	8.4%	18288	8.7%
162.159.200.123	13335-Cloudflare	14838	7.2%	15832	7.5%

Table 15. NTP Servers occurrence for ArgV4 and ArvgV4-Emul experiments. Datasets: [73].

As shown in Table 14 (experiment ArgV4), we send 107k queries from 9.2k Atlas VPs. Each valid response received only *two* A records, and in total, we see eight distinct A records associated with NTP servers under Argentina's zone, as also reported in [62].

Table 15 shows the results. We see that each server receives from 7.2% to 18.3% of all queries – so, in the case of Argentina's subzone, the popular NTP service may appear at least twice as often than the less popular server. We use these results as a *baseline*.

For the emulation experiment, we create a test zone using the A records from Table 15 and, as weights, we use the counts value. We configure `GeoDNS` with this zone file on an AWS EC2 Frankfurt Ubuntu VM and use ~ 9k Atlas probes to query this zone, as shown in Table 3 (dataset ArgV4-Enum), use the same parameters (frequency, duration) in the ArgV4 experiment.

Similarly to EnumV4-Emul experiment, we reproduce the ArvgV4 experiment as ArgV4-Emul. We generate 110k responses from 3128 ASes, as shown in Table 15. We then compute the occurrence

NTP server	ArgV4-Android	ArgV6
162.159.200.1	748	182
162.159.200.123	748	182
168.96.251.195	747	181
168.96.251.227	379	52
168.96.251.197	195	61
168.96.251.226	121	63
170.210.222.10	54	7

Table 16. Query distribution for ArvgV4-Android and ArgV6 experiments. Datasets: [73].

of each IP address from our demo zone in the Atlas responses and find that the query distribution per IP is *very similar* to the *original* experiment using the production servers of the NTP Pool. We can conclude that frequency counts can be used to infer weights in the NTP Pool zones.

E VENDOR ZONES AND IPV6 CLIENTS

The NTP Pool operators encourage vendors to ask for their own DNS subzones [66]. However, we did not find any vendor zones while reverse engineering in NTP Pool zone files (they are not publicly disclosed, and server's report pages do not list them). Are the vendors zones kept apart from the geographical zones? If so, how `GeoDNS` handles them?

We found out experimentally that they are a *replica* of the geographical zones. Their job is to allow the NTP Pool operators with a easy way to *remove* problematic vendors from service without affecting other users.

To determine that, we carry out experiments with RIPE Atlas, asking 32 probes located in Argentina to query for the A record of the Android vendor zone (2.android.pool.ntp.org). We analyzed the A records returned to these responses (dataset ArgV4-Android in [73]), and found only 7 distinct IP addresses,as shown in Table 16, all of them belonging *also* to the `ar` geographical zone. On the same day (2021-08-23), there were only 7 servers active in the `ar` zone [62] .

Therefore, we can conclude that the vendor zones seem to be a *replica* of the geographical zones – only that they give the ability to the NTP Pool operators to remove them in case of a vendor specific errors that can lead to DDoS attacks (CNAME records in DNS can be used to link both zones). As such, clients using vendor subzones are still bound by the geographical zones.

E.1 IPv6 clients

Clients can send queries over IPv4 and IPv6 to the NTP Pool authoritative servers, and they can be used to retrieve both A or AAAA records. To determine if IPv6 clients have a different view from the NTP Pool, we configure 12 RIPE Atlas probes to send queries over IPv6 from Argentina to the NTP Pool authoritative servers. Our goal is to determine if they would be also mapped to the Argentina's `ar` subzone, or if they would use other criteria.

Table 16 shows the results (Argv6 column and dataset).We see that IPv6 clients geolocated in Argentina are also mapped to the ar subzone when asking for A records 2.pool.ntp.org, are also mapped to the `ar` subzone. We confirm that by manually checking the IP address against Maxmind's geolocation database. Therefore, we can conclude that `GeoDNS` uses the same mapping process for IPv4 and IPv6 clients.

Received October 2023; revised January 2024; accepted January 2024