Diving into the NTP Pool

Giovane C. M. Moura (1,2) Marco Davids (1)
Caspar Schutijser (1) Cristian Hesselman (1,3)
1: SIDN Labs 2: TU Delft 3: University of Twente

Technical Report 2021-22
2022-02-03
Updated from 2021-10-29

ABSTRACT

Time is of essence also on the Internet: core protocols and services correct functioning depend upon the ability to have synchronized clocks. The Network Time Protocol (NTP) is the default protocol to synchronize clocks on the Internet. Among time service providers, we find the `NTPPool` project, a volunteer-based project which facilitates the matching between clients and NTP servers – a project which is daily used by millions of users, servers, and an extensive array of devices. The `NTPPool` project has been continuously providing its services for over 18 years, and yet there is not much scrutiny into what servers each client gets to see – and its potential implications. In this paper, we dive into the `NTPPool`, and reveal, with active measurements and emulation experiments, how it matches clients to servers. We present a method to predict the time servers that any user on the Internet would see, and show that by using the `NTPPool`, the entire population of 23 countries are served by a single IPv4 time provider – which includes Israel, Pakistan, and Nigeria. We present three recommendations to improve the `NTPPool` service resilence and discuss the findings with its operators.

1 INTRODUCTION

Time is crucial on the Internet: a series of core Internet applications, services, and protocols that can be compromised or impaired simply by tampering with their hosting system's clocks. TLS [14], DNSSEC signatures [4], DNS caches [33], RPKI [10], Kerberos [34], and even Bitcoin are among the many that fundamentally depend on correct time information [13, 25, 29, 57].

The Network Time Protocol (NTP) [29] is the Internet's default protocol for clock synchronization. NTP *servers* are synchronized with reference clocks (*e.g.,* atomic clocks, GPS/Galileo) and provide correct time information to *clients/secondary time servers*.

Clients, in turn, comprise a very diverse group: servers, IoT devices, mobiles phones, among others. Similar to DNS, there are various (public) *time providers*. Vendors like Apple [3], Google [19], Cloudflare [11], and Microsoft [28] all run their own time services. Clients are typically configured with *hardcoded* NTP servers, which they can manually change or, if available, use the NTP servers provided by the DHCP [15] protocol, which also enables dynamically setting NTP servers [2, 17] (there is no evidence of the widespread usage of this feature [18], so most clients are likely to use the hardcoded servers).

There are many public NTP servers on the Internet. The `NTPPool` project [44] was proposed to simplify the access to these servers [57]. It lists 4634 active NTP servers – 3129 IPv4 and 1505 IPv6 (2022-01-28) [35], and it is the default time provider for Linux and various vendors, such as Linksys, FritzBox, Asus, Zyxel and Sonos [35]. Given such a large user base, one could easily see the `NTPPool` as one of the *core* services on the Internet, although a rather *overlooked* one.

Despite its importance, there is little public scrutiny on *how* the `NTPPool` maps clients to servers: the `NTPPool` documentation is incomplete (a fact acknowledged by its operators, who are volunteers primarily focused on *running* the service), and there is a lack of research works on this particular topic.

Therefore, the goal of this paper is to bring scrutiny on how the `NTPPool` maps users to NTP servers, and which criteria are used in this process. Our first contribution is to *characterize* the client's view from the `NTPPool` in the wild (§3). Drawing upon active measurements from roughly 9k real-world vantage points (VPs) from the Internet (RIPE Atlas probes [53, 54]) – the largest study to date in terms of VPs – we demonstrate that most clients are provided with fewer than 200 NTP servers, and 10% of our VPs see fewer than 12 IPv4 and 5 IPv6 NTP servers, despite the fact that 4.6k servers are listed on the `NTPPool`.

Our second contribution is to understand *why* this uneven distribution occurs (§4). To do that, we *reverse engineer* `GeoDNS` [9], the authoritative DNS server [20] used by the `NTPPool` to map users to NTP servers (the `NTPPool` uses DNS [30] in this mapping – we provide background information in §2). We *reveal* the reasons for this uneven client/NTP server distribution: clients are served with time servers in their own country only (or continent if there are none in the country). We find that that there are many countries

with very few NTP servers in the NTPPool project, so this determines what servers their population gets to be served.

Our third contribution is a method to determine how the entire world population sees the NTPPool, without having to have a single real VP (§5). We show that *all* clients from 13 countries (IPv4) and 42 countries (IPv6) are served by a single provider (Cloudflare). Even though the NTPPool has a fail-over mechanism, it currently does not support IP anycast [1, 47], which is used by Cloudflare, potentially creating a single point of failure. Ultimately, this single-provider dependency is an example centralization and consolidation on the Internet [5–7, 23, 24, 31, 58, 60]. We validate these results against our real world experiment.

Finally, our last contribution (§6) is to present three recommendations for the NTPPool operators on how to improve their services. Overall, our work is the first to show how NTPPool operates and how that influences what clients see. We will release all datasets upon acceptance.

2 BACKGROUND

The NTPPool project is a volunteer-based network of NTP servers [29] that are made available via DNS, under the pool.ntp.org zone and its subdomains. The idea originated in 2003 as a solution to reduce the abuse of publicly available NTP servers [66]. Instead of having a large list with public NTP servers (which individually could more easily become overloaded), the NTPPool project proposed to "load balance" the NTP traffic to servers using the DNS.

To illustrate how this works, consider an NTP client (Figure 1) that wants to synchronize its clock with the NTPPool. In the first step, the client would ask the DNS software on its computer for the IP addresses of the zone in question, using a DNS query. (This is done by DNS stubs, which, in turn, forward the query to recursive resolvers – step 2 – which are capable of performing the recursive resolution tasks). If the resolver does not know the answer from the cache, it will contact one of the NTPPool authoritative servers [20] (step 3, that run GeoDNS), which are the servers that can provide answers about the zone in question.

Upon receiving a query, the NTPPool authoritative servers return a permuted list containing n IP addresses to the resolver (step 4) – a subset of the available servers [35]. In turn, the resolver will respond the DNS stub (step 5) with these addresses, which finally passes it to the NTP client (step 6). Then, the client can contact any of the IP addresses provided and send an NTP query to it (step 7), allowing it ultimately to receive a response (step 8).

The NTP protocol, latency, and jitter: the NTP protocol is designed to mitigate the effects of changes in latency (jitter). NTP clients measure several latency-related metrics (§10 in [29]). It measures one-way delay between server and client

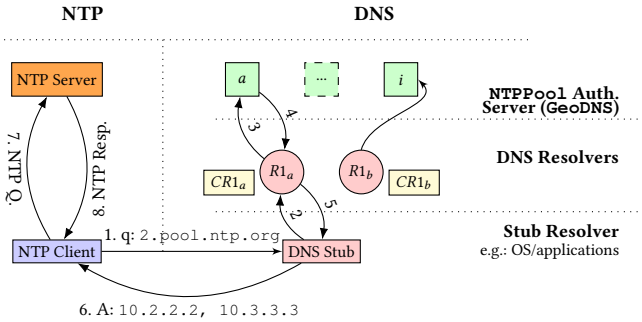

Figure 1: Relationship between NTP and DNS and its different parts: NTP client (blue), NTP Server (orange), DNS resolvers (red) with their caches (yellow), and the authoritative DNS servers (green).

and uses it to calculate the *time offset* between the server (reference clock) and client. The offset, in turn, is the value used to adjust the client's clock. Shorter and more symmetric round-trip times (RTT) lead to more accurate offsets. (This seems to be one of the NTPPool's motivating factors to develop GeoDNS, to be able to serve clients with servers likely geographically closer and with lower, more stable RTTs). NTP clients use the clock filter algorithm to mitigate jitter effects when determining the best time *offset* to synchronize the local clock and also to exclude varying jitter servers from the list of available servers.

The NTPPool project states on its website that there are between 5–15M clients [45][1], but the actual number is hard to know given that DNS heavily relies on caching [33] (*CRn* nodes in Figure 1), and most actual clients are behind resolvers that may cache answers (the DNS records have a 150 s TTL (2.5min), which is the upper limit for caching), and many IPv4 clients are behind Network Address Translation (NAT) [48]devices. In the case of the NTPPool, having low TTLs reduces the consequences of clients receiving only high jitter NTP servers; a new NTP servers can be fetched after the TTL expires.

DNS zones and subzones: authoritative DNS servers use *zone files* [30], which stores DNS *records* and are loaded into memory by a DNS authoritative server. In the case of the NTPPool servers, it is known that it has two main types of zones: (i) geographical zones, which cover five continents and multiple countries and are publicly listed [41], and (ii) vendor zones [45] (such as Android'subzone (android.pool.ntp.org) that are not publicly listed. However, it is not clear how GeoDNS uses these zones to choose what portion of the 4.6k NTP servers the clients see.

[1]Private feedback from NTPPool operators is that these public numbers are far too conservative: a single NTP server in the U.S. received more than 33M IPv4 clients in a 25h period in 2021

Software vendors are encouraged to obtain their own subzone from NTPPool and not use the default name servers. For example, the Debian Linux distribution uses debian.pool.ntp.org. These subzones allows the NTPPool operators to easily mitigate unintentional excess traffic from specific vendors and brands by simply managing the zone used by their devices [45].

Joining and using the pool: Any NTP server operator can join the pool of servers, providing the servers have static IP addresses and use static NTP upstream servers. The NTPPool recommends users to use the [0-3].pool.ntp.org zones to obtain NTP servers' IP addresses [38]. (Users can also query country or continent zones directly if they wish, bypassing GeoDNS decision criteria).

NTPPool authoritative DNS server infrastructure: in §A, we include an analysis on the authoritative DNS servers that are responsible for the NTPPool zone and subzones. These are also provided by volunteers, and currently the NTPPool authoritative DNS infrastructure is hosted in 17 IPv4 and and 12 IPv6 ASes, which is more diverse than most top-level domains (TLDs), such as .com and .it.

3 THE CLIENTS' POOL VIEW

The NTPPool operators self-report that 4.6k IP addresses are served in the pool. We are interested in determining what fraction of these a typical client gets to see.

To measure that, we employ ~9k vantage points (VPs) located in 3082 Autonomous Systems (ASes) and 166 countries, which act like real-world DNS resolvers (which ultimately provide the answers to NTP clients). These VPs are RIPE Atlas probes, which are hardware devices or virtual machines (VMs) that can be remotely instructed to carry out active Internet measurements. We configure these 9k Atlas probes to send DNS queries to one of the authoritative servers of the NTPPool (b.ntpns.org over IPv4 – 185.120.22.23, so we bypass DNS resolvers – Figure 1 – and avoid hitting the resolver's cache). In this way, we can retrieve new NTPPool addresses for every new query. The probes are configured to send queries every 5min – a safe limit that does not overload RIPE Atlas or the NTPPool authoritative servers.

Table 1 shows the experiments' details. In the first experiment (EnumV4), we configure Atlas probes to query for IPv4 NTP servers (defined as A records in DNS [30]). The second experiment (EnumV6), probes query for IPv6 NTP servers (AAAA records [65]). For both experiments, we see ~9.2k active VPs, having 9.1k received valid responses (some VPs are blocked/hijacked or contain wrong answers [32], which we disregard). These 9.1k VPs provide us with a view from ~3k ASes, totaling ~25M DNS queries/responses per experiment.

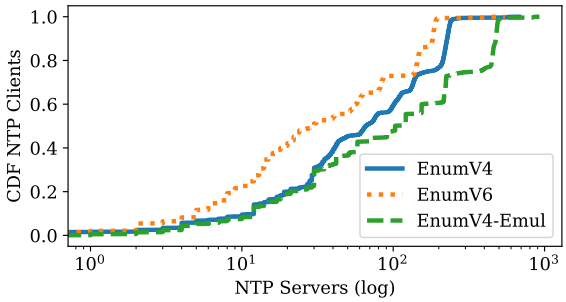

Figure 2: CDF of Pool IPs per Probe.

For each experiment, each VP roughly sends 275 queries, and it receives 4 NTP server addresses per response (Table 1). This, theoretically, would allow each probe to retrieve up to 1100 unique NTP servers addresses from the NTPPool, if the process would be completely random and if each client would not receive repeated NTP servers.

Figure 2 shows the cumulative distribution function (CDF) of the number of unique IPs retrieved by each probe. We see that most probes see fewer than 200 unique IPs, for both measurements, which is fewer than 20% of the maximum theoretical value of 1100 addresses. We see that 10% of the clients see up to 12 NTP servers (EnumV4) and 5 NTP servers (EnumV6). Given that there are fewer IPv6 NTP servers, we see the CDF shifted to the left.

These results show us that the NTPPool NTP servers distribution is somewhat uneven among clients in the wild. Next we investigate the reasons *why* this happens.

4 HOW GEODNS WORKS

The results from §3 shows how *real clients* see the NTPPool, but it does not explain *why*. We address this next.

4.1 GeoDNS behavior in a nutshell

We have determined, using active measurements, how GeoDNS operates, which we summarize next and expand in the following subsections. Before starting, GeoDNS requires two input files: a DNS zone file, which contains the list of domains and records (NTP servers) it will use to serve its clients, and Maxmind's IP geolocation database[26], which provides geographical information associated with IP addresses.

Then, whenever a client sends a DNS query to GeoDNS (step 1 in Figure 3), GeoDNS will then look up the IP geolocation data associated with the client's IP address (step 2) from Maxmind's database and receive the *country* and *continent* associated with the client's IP address (step 3).

Then GeoDNS will attempt to match the client's country of origin to one of sub DNS zones (for example, a client from Japan would be mapped to Japan's jp.pool.ntp.org subzone).

Measurement	EnumV4	EnumV6	ArgV4	ArgV4-Emul
Target	185.120.22.23		185.120.22.23	54.93.163.251
QNAME	2.pool.ntp.org.		3.ar.pool.ntp.org	wilson.ants
QType	A	AAAA	A	A
Date	2021-08-2[6-7]	2021-08-3[0-1]	2021-08-02	2021-08-06
Interval	5min	5min	10min	10min
Duration	24h	24h	2h	2h
VPs	9260	9272	9219	9229
valid resp.	9113	9127	9068	9052
no resp.	147	145	783	382
ASes	3116	3133	3127	3128
valid resp.	3082	3095	3080	3067
no resp.	156	148	474	262
Countries	166	168	1	1
Responses	2534199	2583318	107031	110292
Valid Responses	2469211	2535981	104331	107793
invalid/empty	64988	47337	2700	2499
#NTPPool Addresses	3056	1479	1481	8
per response (median)	4	4	2	2
per response (1st quartile)	4	4	2	2
per response (3rd quartile)	4	4	2	2
Queries per probe (median)	275	275	11.6	11.9

Table 1: NTPPool RIPE Atlas experiments. Datasets: [52].

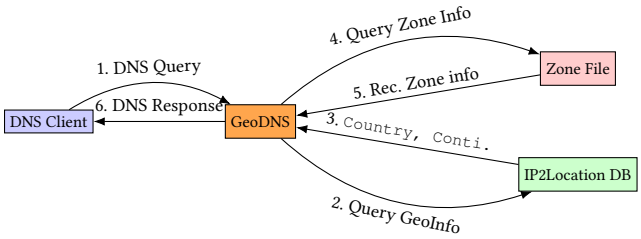

Figure 3: DNS client and GeoDNS relationship.

If there is no specific country zone (or the country zone is empty), then GeoDNS proceeds to a less specific subzones – first the continent (Asia in our example) and then the global zone (@), which lists all NTP servers in the NTPPool.

In this way, GeoDNS iteratively attempts to find a zone where the servers are more likely to be closer to the clients, which hopefully can have shorter and more stable RTT, resulting in better offset calculation (§2).

Bypassing GeoDNS's client/ NTP server matching: users may wish to bypass GeoDNS's decision and use NTP servers from whatever zones they choose. They can do that by changing their NTP settings to point to their desired geographical zone. For example, a client in New Zealand may choose to use Italy's it.pool.ntp.org zone if they wish, so they can use time servers located in Italy. In this paper, we assume that most users will simply rely on the default configurations in their systems, which is to query for vendor zones from the NTPPool that follow GeoDNS's mappings.

Choosing NTP servers within a country: some subzones may have hundreds of servers in them – for example, France's fr.pool.ntp.org has 208 IPv4 NTP servers (2021-12-27). So what criteria does GeoDNS to choose among them? In the GeoDNS zone files, each NTP server (A or AAAA record) has a *weight* associated with it, which is derived from how much service capacity a volunteer wants to donate to the pool. In practice, servers with higher weight values are picked more often (for example, a server with 100 weight will be seen 100 times more often than a server with a 1 weight).

To determine this behavior, we first had to *reverse engineer* the DNS zone file used by the NTPPool, which we did taking the following steps: (1) Determining the zone format; (2) Discover all subzones; (3) Populate all subzones with NTP servers; (4) Determine the weights associated with each server. Then, (5) we use the reverse-engineered zone files on our GeoDNS instance and analyze pcap traffic from our controlled environment. Next, we present these steps in more detail.

4.2 Reverse engineering NTPPool zone files

Given that NTPPool zone files are not publicly available, we resort to reverse engineer them. We could have asked the NTPPool developers for their zone files, but that would not allow us to account for its dynamic nature (§4.3). As such,

we set out to develop a method that allows us to reverse the zone files *anytime* we wish.

The first step consists of determining the zone file format. We use a sample zone file provided by the `GeoDNS` developers [8]. It is a JSON zone file, and it has a global (@) zone, and multiple subzones, each one representing a country or continent. Within each subzone, we have a list of NTP servers, each with its own weight (sample in §B).

The second step consists of determining all *DNS subzones* available on the `NTPPool`. The `NTPPool` website lists all zones and subzones [36], but it does not specify which NTP servers belong to which zones. We then have to *infer* it using active measurements, as follows. First, we compile a list of all NTP servers obtained from the EnumV4 and EnumV6 measurements (§3). We then scrape the `NTPPool` website using each IP address. Each NTP server in the pool has a dedicated page (in the form of https://www.ntppool.org/scores/IP), in which it lists *what zones* the NTP server is associated. For example, the page for the server `95.217.188.206` shows that this NTP server is allocated to the global (`@`), `europe`, and Finland's `fi` zones [43].

We then add these subzones to the demo zone file (if they are not yet present), and the respective IP address in these subzone. We repeat this process for all 3056 IPv4 and 1479 IPv6 NTP severs addresses we found[2].

Table 2 shows the results. The 3056 IPv4 NTP servers from §3 are found in 125 subzones and the 1479 IPv6 are found in 112 subzones. (Some subzones have only A or AAAA records, some are empty). We compare these results with the `NTPPool` self-reported zones, available on their website [36]. Our method shows more non-empty zones than the `NTPPool` self-reported website, which may be due to update delays in their website. (We found no vendor zones in this process, such as `android` zone – see §G).

Figure 4 shows the CDF of the `NTPPool` zone sizes that we have reverse-engineered. We see a large variety in size: 32% of the zones have up to 3 IPv4 NTP servers, and 48% of the zones have up to 3 IPv6 NTP servers. That is a somewhat uneven distribution of servers among all zones, we see that fewer than 9% of the zones (IPv4) have more than 100 NTP servers and fewer than 7% for IPv6.

4.3 Validation

To validate if our reverse-engineered zone file yields the same results as the `NTPPool` authoritative servers, we set up our own `GeoDNS` (v. 3.0.2) instance on a Debian Linux server.

[2]Each NTP server has a *score* on its page, which is provided by the `NTPPool` monitoring system, Only servers with score higher than 10 are used by the `NTPPool`. We consider all servers is regardless of their score, given we found experimentally that *some* servers with low score (<10) were seeing in responses from the `NTPPool` servers – which may be to the fact that scores can dynamically change in the course of the measurements– see §4.3.

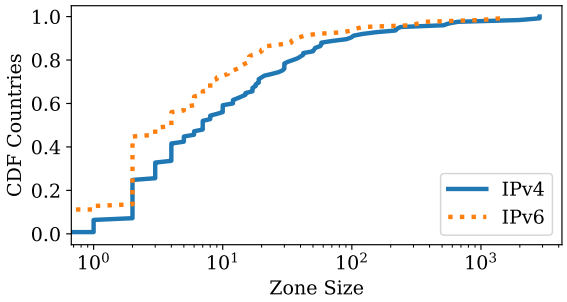

Figure 4: CDF of `NTPPool` subzone sizes.

	Rev. Engineered	Self-reported
Zones - IPv4	126	257
non empty	125	116
Zones - IPv6	126	257
non empty	112	110

Table 2: `NTPPool` zone files results.

Like the `NTPPool` authoritative DNS servers, we configure our server to respond with 4 DNS records per query. We configure it with Maxmind's GeoLite2 country IP2location database[26] from 2021-08-24. We assign all servers with the same weight (1000), so we eliminate the influence of the weights in the query distribution.

Client setup: Next we emulate the EnumV4 experiment, by sending the same queries using the same IP addresses to our local `GeoDNS` server. We do that by by sending spoofed IP packets (forged source IP addresses [16]), using a customized `Python` script, and run our experiment on a Linux server disconnected from the Internet (so our spoofed packets cause no harm).

Collected datasets: we collect two datasets: network traces (`pcap` files), and `GeoDNS` log files (sample in §C, 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.

Results: For each VP in our dataset, we compute two sets: S_{EnumV4} and $S_{EnumV4_{emul}}$, in which we list all NTP servers the VP has seen on each measurement (for the first, we use RIPE Atlas datasets, for the latter we use the `pcap` files from our emulation). We then compare all VPs across both datasets.

Table 3 shows the results. We see that 2265 VPs from both EnumV4 and EnumV4-emul see precisely the same NTP servers – which is all servers that belong to the zones the VPs are mapped (93 zones in total).

We also see 7.2k VPs (which are mapped to 66 zones) that see *fewer* NTP servers in the EnumV4 experiment that what they see from our our emulation experiment. We speculate this can be due to the use of uniform weights (1000) in our

	# Zones	VPs
$S_{EnumV4} = S_{EnumV4_{emul}}$	93	2265
$S_{EnumV4} < S_{EnumV4_{emul}}$	66	7282
$S_{EnumV4} > S_{EnumV4_{emul}}$	12	47

Table 3: Validation results per zone.

	Zones	VPs
=1 NTP server	`cm,gg,re`	34
> 1 NTP server	`es,ie,ir,it,lt,ru south-america,tr,ua`	13

Table 4: Atlas VPs that see more NTP servers than what is available in their respective country subzones.

emulation experiment, whereas in reality, we could expect more diversity in weights within a zone (see §4.3.1).

The last category is the more concerning one: 47 Atlas VPs see *more* NTP servers than our emulation experiments – and they are mapped to 12 subzones. If `GeoDNS` binds a client to a country or continent zone, having Atlas VPs reaching more NTP servers than we have in the `GeoDNS`'s subzones puts in question our findings of its mapping behavior.

We investigate these clients further and classify them into two categories (Table 4): 34 VPs from countries with a single NTP server and 13 VPs from countries with more than 1 NTP server. We explain the reasons for both categories here.

`GeoDNS` zone fallback: we start with VPs from countries with a single NTP server. We illustrate here with a VP from the Bailiwick of Guernsey, an island located in Europe (`gg` country code). The EnumV4 experiment (in the wild) shows that this probe sees, in total, 21 unique NTP servers – even though the `gg` zone has only one NTP server.

This VP (Atlas Probe ID: 17580) initially receives a single NTP server in the DNS responses – 51.255.142.175 – an NTP server from Guernsey (Listing 1), until 20:56. From 21:01 to 21:21, this VP receives 20 different NTP servers in 4 subsequent queries – all servers belonging to the `europe` zone. The presence of these Europe servers suggests that this VP was mapped during this period to `europe` zone and not `gg` zone. After these four queries, however, the same Atlas probe starts to receive a single NTP server in the DNS responses – the same 51.255.142.175.

```
#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|138.201.16.225
                  62.116.130.3|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
```

Listing 1: Atlas VP 17580 responses on EnumV4 experiment (2021-08-26).

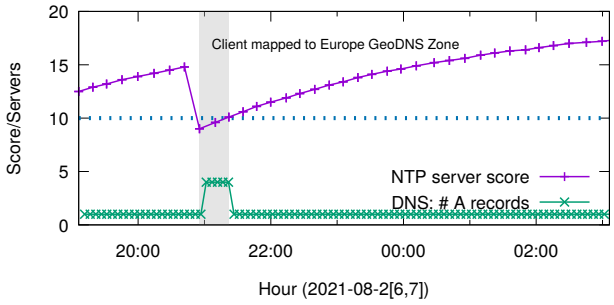

Figure 5: Number of NTP servers per DNS response from VP located in Guernsey and server score from `NTPPool` for server NTP server 51.255.142.175: low scores lead to `NTPPool` eviction and fallback to the continent zone.

So what could explain these changes in the responses? Our EnumV4 datasets do not allow us to explain why. However, the `NTPPool` page about this particular NTP server provides a history of its *scores*, as measured by the `NTPPool` monitoring system – in the form of a `csv` file [42]) – a measurement carried independently from ours.

We correlate our measurement results with the `NTPPool`'s server score logs, as can be seen in Figure 5. We see that this particular server score dropped below 10 – the minimum value for it to be used in the `NTPPool` zones – between 20:55:32 and 21:22:04 (2021-08-26, GMT). We show this in the gray area. (We show the `NTPPool` monitoring system logs also in §D). We also see that the VP located in Guernsey receives 4 NTP servers per response during this interval, completely *coinciding* with the time that this NTP server had poor scores (gray area). 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. The servers from `cm` and `re` zones show similar score issues.

DNS hijack: The remaining VPs with more than a single NTP server (Table 4) comprise probes belonging to multiple countries. We found that these probes are being mapped to another continent/country zones outside of their own, either

IP	ASN	ArgV4 Counts	ArgV4 Ratio	ArgV4-Emul Counts	ArgV4-Emul 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 5: NTP Servers occurrence for ArgV4 and ArvgV4-Emul experiments. Datasets: [52].

by an error in the IP2 location mapping or DNS hijacking, in which the original DNS query to the NTPPool authoritative servers is intercepted on the way – and other studies have shown that this occurs with RIPE Atlas probes [32]. Due to ethical implications [3] – some users may be using overseas DNS servers to bypass censorship, we do not explore this further here.

4.3.1 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, which we compared against a controlled emulation in our setup.

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 [37]), 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 (steps 2 and 3 in Figure 3), obtaining records only listed in the ar subzone. (We confirm this behavior experimentally by running a test locally).

As shown in Table 1 (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 [37].

Table 5 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 5 and, as weights, we use the counts value (we show the final zone file in §E). 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 1 (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 5. We then compute the occurrence 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 NTPPool. We can conclude that frequency counts can be used to infer weights in the NTPPool zones.

4.4 Vendor zones and IPv6 clients

Even though vendors are encouraged to register their own zones, we did not find any vendor subzone in our experiments. We found that the vendor zones seem to be either a alias or a replica of the geographical DNS zones. We also found that IPv6 clients are mapped the same way as IPv4 by GeoDNS. We present a discussion of both in §G.

5 HOW THE WORLD SEES THE POOL

Now that we have seen how GeoDNS works and that we can successfully reverse engineer the NTPPool DNS zone files(§4), we do not have to constrain ourselves to the limited set of VPs (as in §3): we can extend our coverage to the *entire* Internet population, given we can *emulate* the NTPPool authoritative servers with the reverse-engineered zone files.

5.1 Setup

Server side: We configure our local instance of GeoDNS with the zone file generated in §4, and run the server locally disconnected from the Internet.

Client side: Given that GeoDNS uses the client IP geolocation data (country or continent, §4), we only need a single VP *per country*. So we then use one IP address per country, for all countries in Maxmind's DB (§H includes the list of countries and IP addresses). Even though we use only IPv4 clients (IPv6 clients are mapped the same way as IPv4 – §G), we configure them to query for IPv4 and IPv6 addresses, specified by A and AAAA records – so we end up having a complete view of NTPPool for each country.

Experiment: we use IP spoofing and configure our client to send 10k queries per country/IP, for each experiment (CCv4 and CCv6). We choose 10k queries because it is larger than any subzone we have seen (the largest zone is the global zone A records, with 2834 addresses, Figure 4) and likely not large enough to go through the list of all IPs within a zone, given the role of server weights. Table 6 summarizes

[3] Note to the reviewer: we see that some Iranian probes seem to be mapped to the North American zone, which may suggest that they are using DNS resolvers in the US to bypass censorship

	CCv4	CCv6
Qname	idiazabal.kaas	
Qtype	A	AAAA
VPs/Countries	247	247
Responses	2.47M	2.47M
Records per resp		
4	2.06M	1.98M
3	1.00M	0.06M
2	0.23M	0.40M
1	0.08M	0.03M

Table 6: Country Code Experiments.

	CCv4			CCv6		
	CC	Servers	Users/Server	CC	Servers	Users/Server
1	ng	2	68.1	ng	2	68.1
2	in	21	37.1	in	17	45.8
3	eg	2	27.3	ir	2	39.1
4	ph	3	24.3	ph	2	36.5
5	ir	4	19.52	cn	28	35.3
6	cn	52	19.0	vn	2	34.1
7	vn	5	13.6	eg	2	27.3
8	id	18	11.8	id	9	23.5
9	mx	7	12.7	br	8	20.0
10	br	24	6.7	co	2	15.2

Table 7: Top 10 Countries with highest number of Internet Users per NTP server from the NTPPool (in million).

the experiments. We send 2.47M queries per experiment and collect both traffic traces (pcap) and GeoDNS log files.

5.2 Results

From each DNS response, we extract the client's IP address and the A/AAAA records included in the response, which represents the IPv4 and IPv6 NTP servers. Then, we create a set S of NTP servers each client/country has seen for the entire measurement duration. We then compute, for each client, its subzone visibility, which is the set S divided by the subzone size. For example, if a US-based IP client queries for A records (subzone size=561), and this IP retrieves 452 unique records in the 10k queries to GeoDNS, we say its visibility ratio is 452/560 = .80, or 80%.

Figure 6 shows a histogram of the subzones in our reverse-engineered zone file (x axis) for IPv4 NTP servers (we show the IPv6 graph in §F). On the left y axis, we show the number of unique NTP servers that the client retrieved from our servers with 10k queries for each zone (S). On the right y axis, we show the zone visibility for the country.

We divide the countries from Figure 6 into several groups. First, the salmon area on the left shows 41 countries that can only see up to 3 NTP servers: Israel in the Middle East, Nigeria in Africa, the Philippines in Southeast Asia, and Mozambique in Africa, among others. (For IPv6 servers, we see in §F that there are 48 countries). These are the most concerning countries, and we reported these finds to the NTPPool operators (we show the full list on §H).

The next category are countries highlighted in grey: these are countries that *do not* have a country subzone in the pool (or have empty subzones), and wind up being mapped to the continent or the global zone. These include Bolivia in South America, Ghana in Africa, the Fiji Islands in Oceania, and Yemen in Asia. As can be seen in the figure, users in these countries are served by a large number of NTP servers (at least 38 IPv4 and 13 IPv6 NTP servers). These countries are better off in server diversity than those 40+ countries with few (<= 3 NTP servers) in their own subzones.

We also see that 7 and 8 countries are mapped to the global (@) IPv4 and IPv6 zones, respectively. These include South Sudan (ss), Bouvet Island (bv), and Antarctica (aq). We found that South Sudan, for example, has no continent data in its target logs (Listing 3) – only country and global. Bouvet Island, in turn, has Antarctica as continent zone, which is an empty zone.

The final group is the areas left in white, which show countries with their own zone with the NTPPool. The leftmost white area covers countries with 4–32 NTP servers. Note that our 10k queries per country led to these zones' complete visibility. They include Greece, Ireland, Thailand, South Africa, and Brazil.

The better-off countries are those shown between Asia and North America: they have their own zones in the pool with many NTP servers. For A records, we have the US and Germany with 561 and 514 records in their zones, respectively. For AAAA records, we have Germany with 362 records.

Users per NTP server: next, we compute the number Internet users per each NTP server for each country. To do that, we divide a country's Internet population (obtained from the Wikipedia entry on Internet users, which draws from various data sources [68]) by the number of servers that server that zone. Table 7 shows the results: 68.1M Nigeria users are served by a single NTP server. India, despite having 21 IPv4 NTP servers, have a large population, which causes 37.1M users being served per NTP server.

The conclusion is that despite thousands of servers being available to the NTPPool, users from many countries can only receive time information from a limited set of NTP servers.

5.3 Validation

To determine if our reverse-engineered zone file and setup lead to similar results as in the real world, we compare the

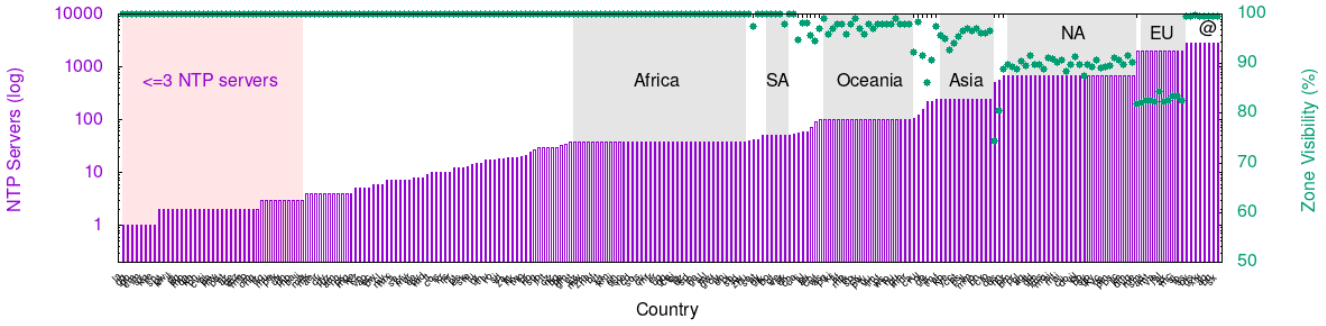

Figure 6: CCv4: countries' `NTPPool` visibility.

	S_{test}	S_{val}
Countries	246	166
$\bigcap$	166	
$I_r = 1$	112 (67.4%)	
$I_r < 1$	54 (32.6%)	
CC w/ subzone	5	
CC wo subzone	49	
<= 2 VPs	38	
2<VPs<=5	4	
VPs>=5	7	

Table 8: Validation Results.

outputs of our emulation (Table 6) with experiments and real-world experiments from §3.

To do that, we extract, for each country/IP from CCv4, the set of NTP servers it has seen (S_{test}). Similarly, from EnumV4, we produce, per country, a list of NTP servers that *all probes* located on the particular country has seen (S_{val}). We then compute the intersection ratio as (I_r) and divide it by the number of servers seen by S_{test}. In other words, the percentage of servers that belong to both test and emulation experiments, normalized by the size of emulation zone:

$$I_r = \frac{S_{test} \bigcap S_{val}}{S_{test}} \quad (1)$$

Table 8 shows the results. We see that 166 countries are presented on both datasets – our emulation experiment has more countries than Ripe has Atlas has VPs located on. Out of these, 112 countries have I_r equal to 1 — meaning that our emulated VPs see NTP servers that are observed in the real world by VPs from the same country.

For the 54 remaining countries, our emulation saw *more* NTP servers than what was observed in the wild. This is because our because RIPE Atlas has limited coverage in these countries (42 countries have fewer than 5 VPs), so our emulation experiments has better visibility than RIPE Atlas.

Only 5 countries had both subzones and more NTP servers on S_{val}. We analyzed them and found that some of them have probes that seem to have their DNS queries either hijacked – DNS hijacks in RIPE Atlas have been previously reported [32]. For example, for a particular country in Asia, some VPs see NTP servers listed in the US and North America zones.

Overall, we see that our emulation experiments can be used to predict the global client's visibility from the `NTPPool`.

6 IMPROVING SERVICE

6.1 Mitigate dependency on single providers

In §5 we measured the `NTPPool` visibility from every country. Next, we analyze how many time providers (instead of NTP servers) each country sees, defined by the number of unique Autonomous Systems (ASes) that the NTP servers belong (a same time provider may host multiple NTP servers for a given country).

Figure 7 shows the CDF of time providers (ASes) per subzone. 13 countries have a *single* IPv4 time provider, while 42 have a *single* IPv6 time provider (42 countries also have no IPv6 time provider). (We also see that only ~40% of the countries have more than 10 IPv4 time providers, and 22% of countries have more than 10 IPv6 providers). We include the full list in §H.

Cloudflare's participation: Cloudflare's NTP service is also offered freely on the `NTPPool` project. Given that Cloudflare operates a large CDN using anycast, it can afford to share many more resources than, for example, regular users with a single unicast server.

We compute Cloudflare's participation in all DNS responses, for each country we measured. (We analyze 247 countries in total). First, we see a spike on x=1, corresponding to countries that solely rely on Cloudflare as time provider. We found that 24 countries solely rely on Cloudflare for NTP service over IPv4, and 36 countries over IPv6, as shown in Table 9. The include populous countries such as Pakistan, Nigeria. (We

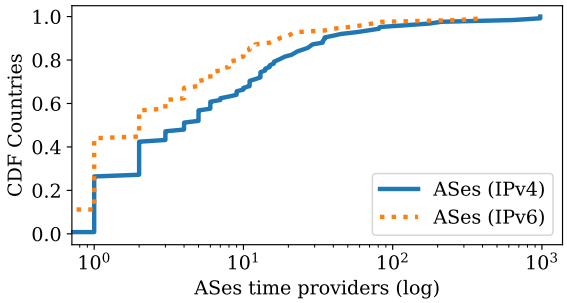

Figure 7: CDF time providers per country.

A (IPv4)	AAAA (IPv6)
az,bh,cw,dj,eg,	ae,ao,az,bh,co,cw,dj
gt,ht,il,kh,kw	eg,gt,hr,ht,il,iq,kw,lb
lb,mn,mo,mz,ng,	lk,mg,mn,mo,mu,mv,mz
om,pa,qa,rw,sn	ng,np,om,pa,pe,ph
pk,qa,rw	pk,py,qa,rw,sn,tz,vn

Table 9: List of Countries that are fully depend on Cloudflare for `NTPPool` time service.

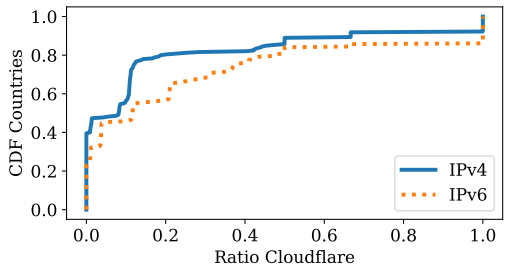

Figure 8: CDF Cloudflare's participation per country.

validate this by querying the `NTPPool` authoritative servers for each country subzone, and indeed, all NTP servers returned are from Cloudflare – as in the wild with EnumV4 and EnumV6).

The reasons for this dependency on Cloudflare is that those countries' zones only list Cloudflare's time servers, and `GeoDNS` DNS would then include them in all responses from IP addresses from these countries (some countries, however, are only dependent on Cloudflare for IPv6 NTP servers). As a consequences, entire populations depend upon a single time provider. Even though Cloudflare operates a robust network, we have seen in the past very large CDNs and DNS infrastructure experience stress, from AWS [69], and Dyn [49]. As such, diversity of providers can provide an extra layer of redundancy.

Recommendation: we recommend increasing the number of time providers for countries with a single or few ones. This can be done by either adding more NTP servers (which is problematic because the service is only volunteer-based – *e.g.*,, see [39]) or *changing* `GeoDNS` to allow that clients from countries with few time service providers can automatically fallback to their continent zones, and, as such, having more time service providers, which can be configured as a parameter in the configuration file.

While the `NTPPool` has a monitoring system that checks the responsiveness of NTP servers (as in Figure 5), it has a single VP, and, as such cannot measure an anycast provider like Cloudflare. We address this next.

6.2 Add multiple monitoring VPs to cope with anycast

IP anycast allows IP prefixes to be announced from multiple *locations* worldwide. It is heavily used by many CDNs and cloud providers as well in DNS – the Root DNS servers, for example, all employ IP anycast, and `L-root`, the largest Root Server, is deployed in 196 global locations (Dec. 21) [55].

BGP [51], the default inter-domain routing protocol on the Internet, maps different clients to one of the anycast locations. As such, each client sees only a single location, regardless of the size of the anycast network – and these mappings are typically quite stable [67].

We found that 9 IPv4 NTP servers on the `NTPPool` deploy anycast, by comparing their prefixes with the anycast census data from previous works [61, 63] (the census only covers IPv4). These anycast servers are present in 98 subzones, out of 125 non-empty ones (78.4% of all zones, Table 2).

Table 10 shows the IPv4 anycast NTP servers in the `NTPPool` (July 2021). We see that Cloudflare's NTP servers are the most widely used in terms of subzones (95), and announced from at least 50 locations. Four other operators also run anycast NTP servers, serving 4–21 subzones, and are announced from 3 or 15 locations at least (the census dataset may also not have a complete view of all sites).

For each anycast address from Table 10, the `NTPPool` monitoring system (which is a single VP located in San Jose, California), can only reach a single anycast location. As such, it will miss any stress or service disturbance experienced by any location that it cannot reach (previous works have shown that during a DDoS, many anycast locations can scape harmless, while others can be completely unreachable [32]). The consequences can be severe: countries that depend on single time provider may be left without NTP service if the respective anycast location serving these countries fail and are not detected by the single VP `NTPPool` monitoring system.

Address	Subzones	Any. Locations	Operator
162.159.200.1	95	50	Cloudflare
162.159.200.123	95	50	Cloudflare
194.0.5.123	21	15	SIDN
129.250.35.250	12	15	NTT
129.250.35.251	12	15	NTT
158.51.134.123	4	3	Nonexist
27.124.125.250	4	3	Dreamscape
27.124.125.251	4	3	Dreamscape
27.124.125.252	4	3	Dreamscape

Table 10: Anycast NTP servers on the `NTPPool` (IPv4).

Recommendation: we recommend that the `NTPPool` operators deploy a monitoring system from many global locations from different ASes to detect failures on multiple locations within anycast networks and devise a server eviction strategy that considers anycast locations.

6.3 `GeoDNS`: set limits to prevent dominant NTP servers

NTP servers listed in the `NTPPool` are provided by volunteers, ranging from home users to CDN operators. Volunteers choose how much server capacity they want to share with the `NTPPool`, which is ultimately converted into the weights we see on the DNS zone files. We can expect a significant difference in these weights, but how much they vary per subzone?

To investigate that, we analyze the weights from our reverse-engineered zone (§5), which were obtained using a real-world experiment (EnumV4). We show in Figure 9 the top 50 zone for IPv4 NTP servers, in terms of zone size (we include the country code mapping to country name in §H, and the we show the IPv6 figure in §F). For each zone, we compute its server weight distribution. On the left y axis we show the normalized weight distribution (w.r.t. the total weight) for all records within each zone, while on the right y axis we show the number of records in the subzone we reversed engineering.

To determine how much variation exists in each subzone, we compute its interquartile range (IQR), which is shown by the box's height in Figure 9. The green line in each box shows the median weight in the zone – and the distance between the green line and the top whisker shows how dominating the top server is in the zone. We see zones such as Germany's `de` subzone, with 514 NTP servers, having a median weight of 456. However, the top server for `de`, from Cloudflare, has a weight of 30391 – almost two order of magnitude larger than the median. Smaller zones tend to be more concentraded.

We choose 4 countries from Figure 9 and show their NTP servers weight's CDFs in Figure 10. We choose these countries given size difference. On the y axis, we show the CDF of NTP servers in the zone (each zone size is in its legend). On the x axis, we show the cumulative normalized weight of the servers, *i.e.,* the cumulative individual weight divided by the sum of all weights in the zone. We see that for Germany (`de`) and the United States (`us`) zones, both with more than 500 servers, we see that a Pareto-like distribution: 78% of the servers account for only 20% of all occurrences (scores); and 22% of the servers account for 80% scores. We see a more homogeneous distribution for smaller zones, such as Japan (`jp`) and Taiwan (`tw`).

This distribution has to do with the nature of the servers in their zone and how much each volunteer is willing to share. For Germany's zone, the smallest weight for a NTP server is one, meaning it appeared only once in all 1.2M times records from the `de` zone were returned. In contrast, the server that appeared the most did it for 30k times (2.3% of all times), which in this case is one of Cloudflare's NTP server (162.159.200.1).

The reality for the `NTPPool`, however, is that the diversity that clients see from a zone is not necessarily linearly correlated with the size of the pool: a small fraction of the servers may dominate a zone, which is a direct result by how much each volunteer is able to donate as weights. As such, we recommend the `NTPPool` operators to evaluate, for each zone, how much each NTP server accounts for all answers in order to avoid significant concentrations in the hands of few time providers.

7 ETHICS AND PRIVACY

Our paper has two ethical concerns: avoiding negative consequences of our measurements in both clients (VPs) and servers, respecting the privacy of these VPs, and disclosing our findings to the `NTPPool` operators.

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 (§4), we use safe query rates (1 DNS query per 5 or 10min, depending on the experiment), and we limit the experiment duration for up to 24h.

We also crawl web pages related to each NTP server on the `NTPPool` website (§4.2) – fewer than 5k pages. To minimize impact, we rate-limit our crawler to 1 web page/s. Moreover, our Ripe Atlas experiments only send DNS queries – we only request data from the `NTPPool` authoritative servers, so we did not measure any NTP server. In our emulation experiments (§5), we use IP spoofing. To prevent that this impact the spoofed IPs on the Internet, we use a computer without Internet connectivity.

Privacy: we found cases of RIPE VPs that seem to use overseas DNS servers – which may be due trying to bypass government censorship or DNS hijack – DNS hijack in RIPE VPs has been known for years [32, 64]. While we cannot

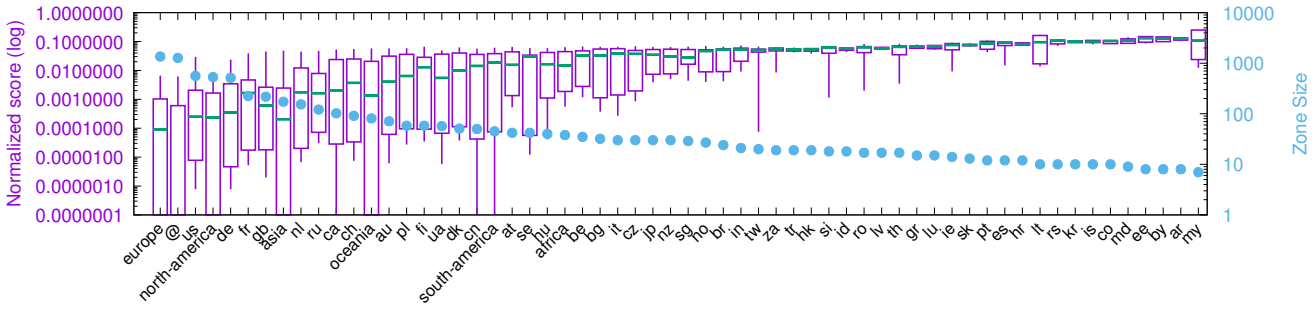

Figure 9: IPv4 NTP Servers: top 50 zones' weights distribution . Whiskers show *min* and *max* weight values, boxes shows IQRs, median values are shown as a green line.

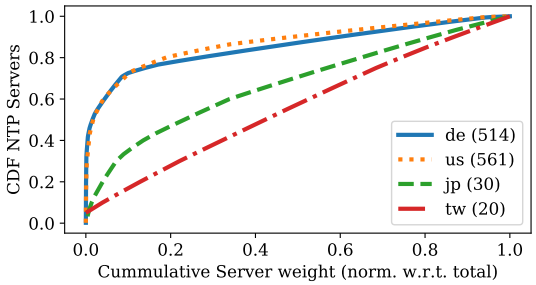

Figure 10: Cumulative weights and subzones - IPv4 servers (A Recs). Numbers is parentheses show zone size.

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.

Discussion with `NTPPool` operators: we shared multiple versions of this manuscript with the `NTPPool` operators, which upon reading the manuscript, did not oppose or question our findings and methods – they have, however, provided more that helped us clarity various parts of the paper. We thank them for their help.

8 RELATED WORK

A previous study to ours on NTP servers [57] has also crawled the `NTPPool` authoritative servers to enumerate them (similar to what we did in §3). They used a single vantage point, whereas we used 9k with RIPE Atlas (§3). Whereas they use their datasets to analyze the distribution of time providers, we use them to reverse engineer the `NTPPool` zones and scrutinize `GeoDNS` behavior, allowing us to determine how it works and ultimately emulate queries from all countries, covering all users everywhere.

While not directly related to ours, there are several other studies that focused on NTP security. They either covered the NTP protocol vulnerabilities [25], how NTP clients can be vulnerable to malicious time servers [13, 50] or how NTP servers can be used in distributed denial-of-service (DDoS) amplification attacks [12] (in which spoofed queries are sent with the source address of the target, which then receives unsolicited traffic), or off-path attacks using DNS cache poisoning [22]. While related to ours, they do not focus on the `NTPPool` itself, as we do.

With regards to NTP traffic characterization, a previous study has characterized traffic at the NIST's NTP servers [59] or running several NTP servers that are part of the `NTPPool` [57]. Our study, in turn, solely analyzes DNS traffic towards the `NTPPool` DNS servers (or emulated), which then is used to point clients to NTP servers.

9 CONCLUSIONS AND FUTURE WORK

The `NTPPool` project has been providing an invaluable service for millions of users, servers, and devices globally, for over 18 years (since Jan. 2003). Their operators and developers – all volunteers – deserve to be acknowledged for the continuous services they have been providing.

We have scrutinized its DNS authoritative service, which is responsible for deciding *who* gets to see what servers. Despite having ~ 4k NTP servers in the `NTPPool`, we see that most countries (60%) are served by 10 time providers only. Worse, many countries – including Nigeria, Israel and Pakistan – have a single provider – which in the case of Nigeria is about four times the population of California. Current `NTPPool` protection mechanisms cannot cope these services if they use anycast, as in the case of Cloudflare.

We have shared these results with the `NTPPool` operators, and we hope it can be used to improve their services. Overall, our study brings more transparency into who gets to tell time for clients all over the world. Our experiments do not depend no private data and can be reproduced.

As future work, develop montoring system that supports IP anycast for the `NTPPool` and investigate the quality of

the services provided by the NTP servers that the NTPPool share with its clients.

REFERENCES

[1] J. Abley and K. Lindqvist. 2006. *Operation of Anycast Services.* RFC 4786. IETF. http://tools.ietf.org/rfc/rfc4786.txt

[2] S. Alexander and R. Droms. 1997. *DHCP Options and BOOTP Vendor Extensions.* RFC 2132. IETF. http://tools.ietf.org/rfc/rfc2132.txt

[3] Apple. 2021. Apple NTPService. time.apple.com. (Aug. 2021).

[4] R. Arends, R. Austein, M. Larson, D. Massey, and S. Rose. 2005. *DNS Security Introduction and Requirements.* RFC 4033. IETF. http://tools.ietf.org/rfc/rfc4033.txt

[5] J. Arkko. 2019. Centralised Architectures in Internet Infrastructure. Internet Draft. (Nov. 2019). https://tools.ietf.org/html/draft-arkko-arch-infrastructure-centralisation-00

[6] 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

[7] 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. (July 2019). https://tools.ietf.org/html/draft-arkko-iab-internet-consolidation-02

[8] Ask Bjørn Hansen. 2012. GeoDNS sample zone file. http://tmp.askask.com/2012/08/dns/ntppool.org.json.big. (Aug. 2012).

[9] Ask Bjørn Hansen. 2021. GeoDNS servers. https://github.com/abh/geodns/. (July 2021).

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

[11] Cloudflare. 2021. Cloudflare Time Service. https://www.cloudflare.com/time/. (Aug. 2021).

[12] 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 (IMC).* ACM, 435–448. https://doi.org/10.1145/2663716.2663717

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

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

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

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

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

[18] Giovane C. M. Moura. 2021. setting ntp with dhcp. https://mailman.nanog.org/pipermail/nanog/2021-October/215616.html. (2021).

[19] Google. 2021. Google Public NTP. https://developers.google.com/time. (Aug. 2021).

[20] P. Hoffman, A. Sullivan, and K. Fujiwara. 2018. *DNS Terminology.* RFC 8499. IETF. http://tools.ietf.org/rfc/rfc8499.txt

[21] ISS. 2021. Zonemaster: pool.ntp.org Contains errors! https://zonemaster.iis.se/en/?resultid=007295f8ad2799a7. (30 June 2021).

[22] 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

[23] 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

[24] 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 (IMC '20).* Association for Computing Machinery, New York, NY, USA, 634–647.

[25] 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).*

[26] Maxmind. 2021. Maxmind. (2021). http://www.maxmind.com

[27] D. McPherson, D. Oran, D. Thaler, and E. Osterweil. 2014. *Architectural Considerations of IP Anycast.* RFC 7094. IETF. http://tools.ietf.org/rfc/rfc7094.txt

[28] Microsoft. 2021. Microsoft NTP Service. http://time.windows.com. (Aug. 2021).

[29] D. Mills, J. Martin, J. Burbank, and W. Kasch. 2010. *Network Time Protocol Version 4: Protocol and Algorithms Specification.* RFC 5905. IETF. http://tools.ietf.org/rfc/rfc5905.txt

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

[31] 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 (IMC '20).* Association for Computing Machinery, New York, NY, USA, 42–49. https://doi.org/10.1145/3419394.3423625

[32] 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

[33] 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

[34] C. Neuman, T. Yu, S. Hartman, and K. Raeburn. 2005. *The Kerberos Network Authentication Service (V5).* RFC 4120. IETF. http://tools.ietf.org/rfc/rfc4120.txt

[35] NTP Pool. 2021. All Pool Servers. https://www.ntppool.org/zone. (6 2021).

[36] NTP Pool. 2021. All Pool Servers. https://www.ntppool.org/zone. (June 2021).

[37] NTP Pool. 2021. Argentina — ar.pool.ntp.org. https://www.ntppool.org/zone/ar. (June 2021).

[38] NTP Pool. 2021. How do I use pool.ntp.org? https://www.ntppool.org/en/use.html. (June 2021).

[39] NTP Pool. 2021. List of underserved zones? . https://community.ntppool.org/t/list-of-underserved-zones/2133/9. (Aug. 2021).

[40] NTP Pool. 2021. NTP Pool DNS servers. https://www.pool.ntp.org/dns-server.html.. (July 2021).

[41] NTP Pool. 2021. NTP servers in Global, pool.ntp.org. https://www.ntppool.org/zone/@. (June 2021).

[42] NTP Pool. 2021. pool.ntp.org: statistics for 51.255.142.175 . https://www.ntppool.org/scores/51.255.142.175/. (Aug. 2021).

[43] NTP Pool. 2021. pool.ntp.org: Statistics for 95.217.188.206. https://www.ntppool.org/scores/95.217.188.206. (June 2021).

[44] NTP Pool. 2021. pool.ntp.org: the internet cluster of ntp servers. https://www.ntppool.org/en/. (June 2021).

[45] NTP Pool. 2021. The NTP Pool for vendors. https://www.ntppool.org/en/vendors.html. (June 2021).
[46] NTP Pool Community. 2021. DNS configuration errors/issues with pool.ntp.org. https://community.ntppool.org/t/dns-configuration-errors-issues-with-pool-ntp-org/2039. (June 2021).
[47] C. Partridge, T. Mendez, and W. Milliken. 1993. *Host Anycasting Service.* RFC 1546. IETF. http://tools.ietf.org/rfc/rfc1546.txt
[48] R. Penno, S. Perreault, M. Boucadair, S. Sivakumar, and K. Naito. 2016. *Updates to Network Address Translation (NAT) Behavioral Requirements.* RFC 7857. IETF. http://tools.ietf.org/rfc/rfc7857.txt
[49] Nicole Perlroth. 2016. Hackers Used New Weapons to Disrupt Major Websites Across U.S. *New York Times* (Oct. 22 2016), A1. http://www.nytimes.com/2016/10/22/business/internet-problems-attack.html
[50] 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).*
[51] Y. Rekhter, T. Li, and S. Hares. 2006. *A Border Gateway Protocol 4 (BGP-4).* RFC 4271. IETF. http://tools.ietf.org/rfc/rfc4271.txt
[52] RIPE NCC. 2021. RIPE Atlas Measurement IDS. https://atlas.ripe.net/measurements/ID. (June 2021). , where ID is the experiment ID: EnumV4: 32025718, EnumV6: 32058440, ArgV4: 31789516, ArgV4-Emul:31830680, ArgV4-Android: 31992051, DE-Android:31970486, ArgV6:32001506.
[53] RIPE NCC Staff. 2015. RIPE Atlas: A Global Internet Measurement Network. *Internet Protocol Journal (IPJ)* 18, 3 (Sep 2015), 2–26.
[54] RIPE Network Coordination Centre. 2020. RIPE Atlas. https://atlas.ripe.net. (2020).
[55] Root Server Operators. 2021. Root DNS. (May 2021). http://root-servers.org/.
[56] Root Zone file. 2020. Root. (Feb. 2020). http://www.internic.net/domain/root.zone.
[57] 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
[58] Bruce Schneier. 2018. Censorship in the Age of Large Cloud Providers. (2018). https://www.schneier.com/essays/archives/2018/06/censorship_in_the_ag.html
[59] Jeff A Sherman and Judah Levine. 2016. Usage analysis of the NIST internet time service. https://tf.nist.gov/general/pdf/2818.pdf. *Journal of Research of the National Institute of Standards and Technology* 121 (2016), 33. https://tf.nist.gov/general/pdf/2818.pdf
[60] Internet Society. 2019. Consolidation in the Internet Economy. (2019). https://future.internetsociety.org/2019/
[61] Raffaele Sommese. 2021. MAnycast² Anycast Census Data Repository. https://github.com/ut-dacs/Anycast-Census. (12 2021).
[62] Raffaele Sommese, Leandro Bertholdo, Gautam Akiwate, Mattijs Jonker, van Rijswijk-Deij, Roland, Alberto Dainotti, KC Claffy, and Anna Sperotto. 2020. MAnycast2—Using Anycast to Measure Anycast. In *Proceedings of the ACM Internet Measurement Conference.* ACM, Pittsburgh, PA, USA. https://doi.org/10.1145/3419394.3423646
[63] Raffaele Sommese, Leandro Bertholdo, Gautam Akiwate, Mattijs Jonker, Roland van Rijswijk-Deij, Alberto Dainotti, KC Claffy, and Anna Sperotto. 2020. MAnycast2: Using Anycast to Measure Anycast. In *Proceedings of the ACM Internet Measurement Conference (IMC '20).* Association for Computing Machinery, New York, NY, USA, 456–463. https://doi.org/10.1145/3419394.3423646
[64] 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/. (Dec. 2015).

Server	a	b	c	d	e	f	g	h	i
# IPv4	6	2	3	4	3	3	1	1	1
# IPv6	3	1	1	3	3	2	0	1	1

Table 11: DNS infrastructure of `[a–i].ntpns.org` (2021-06-29).

[65] S. Thomson, C. Huitema, V. Ksinant, and M. Souissi. 2003. *DNS Extensions to Support IP Version 6.* RFC 3596. IETF. http://tools.ietf.org/rfc/rfc3596.txt
[66] Adrian von Bidder. 2003. ntp DNS round robin experiment. https://groups.google.com/g/comp.protocols.time.ntp/c/cShrN7imCJ0. (Jan. 2003).
[67] Lan Wei and John Heidemann. 2017. Does Anycast Hang Up On You?. In *IEEE Network Traffic Monitoring and Analysis Conference.* IEEE, Dublin, Ireland, 9. https://doi.org/10.23919/TMA.2017.8002905
[68] Wikiepdia. 2021. List of countries by number of Internet users. (08 2021). https://en.wikipedia.org/wiki/List_of_countries_by_number_of_Internet_users
[69] Chris Williams. 2019. Bezos DDoS'd: Amazon Web Services' DNS systems knackered by hours-long cyber-attack. https://www.theregister.co.uk/2019/10/22/aws_dns_ddos/. (10 2019).

A NTPPOOL DNS INFRASTRUCTURE

NTP Servers under the NTPPool are available under *.pool.ntp.org and pool.ntp.org. The zone pool.ntp.org has, in turn, [a--i].ntpns.org as authoritative servers (Figure 1). The NTPPool also uses volunteer authoritative DNS servers [40]. These are the servers that are ultimately responsible for answering resolver queries about the NTPPool, which in turn allows clients to synchronize their clocks. Given that, it is vital that the NTPPool DNS infrastructure is highly redundant, which we investigate next.

To do that, we query directly one of their parent's authoritative servers (anyns.pch.net, *i.e.,* one of the authoritative servers for [a--i].ntpns.org) over the IPv4 address for the A and AAAA records of [a--i].ntpns.org. Table 11 shows the results. We see that in total, there are 24 A records and 15 AAAA records associated with the [a--i].ntpns.org domains.

Taking as whole, Table 12 shows the number of unique addresses for all authoritative servers of the NTPPool. We see that 3 IPv4/ASes and 2 IPv6/ASes are not responsive (we keep a snapshot of the measurement here [21]). We have reported this to the operators (2021-06-30) [46], but it has not been fixed by time of this writing (2021-07-07).

Another layer of redundancy can be added by using IP anycast [27, 47], which is heavily used in the Root DNS zone and multiple other zones as well. We use IPv4 anycast census the 2021 dataset from [62] and match the A records used by

	IPv4	IPv6	ASes(IPv6)	ASes(IPv6)
Unique	24	15	20	14
Resp.	21	12	17	12
Not Resp.	3	2	3	2

Table 12: Aggregated data from `NTPPool` authoritative servers (2021-06-30) [21].

the authoritative servers for the `NTPPool`. We see that none of them use IP anycast according to the 2021 census.

Regardless of these errors and lack of anycast, we can conclude that the AS and IP diversity of the addresses used by the parent DNS servers of the pool is highly diverse (Table 12) – likely to the diversity in volunteering organizations willing provide the service. For comparison, the Top-level domains (TLDs) from Root DNS zone [55, 56] have, on average, 2.7 ASes for IPv4 and IPv6 addresses (2021-07-05).

B 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[4]. 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 `NTPPool` 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" : {
      "" : {  "" 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
            ],
            ...
         ]
      },
    "tr" : { #subzone: tr.pool.ntp.org
         "a" : [
            [
               "77.243.184.65",
               "1000000"
            ],
            [
               "212.175.18.126",
               "100000"
            ],
    ...}
```

Listing 2: GeoDNS demo zone file for `pool.ntp.org`. # marks our comments.

[4]Traditional authoritative DNS server use standardized text zone file formats [30], but `GeoDNS` uses JSON zone files instead [9].

C GEODNS SAMPLE LOG

Listing 3 shows a sample log from `GeoDNS` for a single query/response. In this log, an Israel-based client (`RemoteAddr` field) can be mapped to three subzones (`Targets: il, asia, @`), but it is ultimately mapped to Israel's country zone (`il`).

```
{  "Time": 1626941639825507800,
  "Origin": "zweig.welt",
  "Name": "zweig.welt.",
  "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 3: Sample Log from GeoDNS

D DYNAMIC BEHAVIOR FROM THE NTPPOOL AUTHORITATIVE SERVERS

Listing 4 shows the `NTPPool` score with respect the 51.255.142.175 server, measured from the monitoring system from the `NTPPool`, obtained from [42].

```
ts_epoch,ts,offset,step,score,monitor_id,monitor_name,
    leap,error
1630012924,"2021-08-26 21:22:04",0.000396957,1,10.1,9,"
    San Jose, CA, US",0,
1630012924,"2021-08-26 21:22:04",0.000396957,1,10.1,,,0,
1630012190,"2021-08-26 21:09:50",0.001251177,1,9.6,9,"San
     Jose, CA, US",0,
1630012190,"2021-08-26 21:09:50",0.001251177,1,9.6,,,0,
1630011332,"2021-08-26 20:55:32",0,-5,9,9,"San Jose, CA,
    US",,"i/o timeout"
1630011332,"2021-08-26 20:55:32",0,-5,9,,,,"i/o timeout"
1630010513,"2021-08-26 20:41:53",0.002216952,1,14.8,9,"
    San Jose, CA, US",0,
1630010513,"2021-08-26 20:41:53",0.002216952,1,14.8,,,0,
1630013498,|51.255.142.175
```

Listing 4: `NTPPool` score files for server 51.255.142.175 .

E ZONE FILE ARGV4-EMUL EXPERIMENT

```
{
  "ttl": 390,
  "serial": 2,
  "data": {
    "": {
      "ns": [
        "ns.zeit1.org",
        "ns.zeit2.org"
      ],
      "a": [
        [
          "162.159.200.1",
          "37580"
        ],
        [
          "168.96.251.227",
          "31142"
        ],
        [
          "170.210.222.10",
          "28599"
        ],
        [
          "168.96.251.226",
          "25707"
        ],
        [
          "181.93.10.58",
          "24878"
        ],
        [
          "168.96.251.195",
          "24731"
        ],
        [
          "168.96.251.197",
          "17223"
        ],
        [
          "162.159.200.123",
          "14838"
        ]
      ]
    }
  },
  "max_hosts": 2
}
```

Listing 5: ArgV4-Emul zone file

F EXTRA FIGURES

Figure 11 shows the top AAAA 50 zones and the weight distributions of the servers within each zone.

Figure 12 show the countries visibility of the NTPPool, for for AAAA records.

G VENDOR ZONES AND IPV6 CLIENTS

The NTPPool operators encourage vendors to ask for their own DNS subzones [45]. However, we did not find any vendor zones while reverse engineering in NTPPool 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 NTPPool operators with a easy way to *remove* problematic vendors from service without affecting other users.

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 13: Query distribution for ArvgV4-Android and ArgV6 experiments. Datasets: [52].

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 [52]), and found only 7 distinct IP addresses,as shown in Table 13, 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 [37] .

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 NTPPool 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.

G.1 IPv6 clients

Clients can send queries over IPv4 and IPv6 to the NTPPool 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 NTPPool, we configure 12 RIPE Atlas probes to send queries over IPv6 from Argentina to the NTPPool 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 13 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.

H EXTRA TABLES

Table 14 show shows the list of countries with fewer than 3 NTP IPv4 servers (A records). Table 15 the list of countries with fewer than 3 NTP IPv6 servers (AAAA).

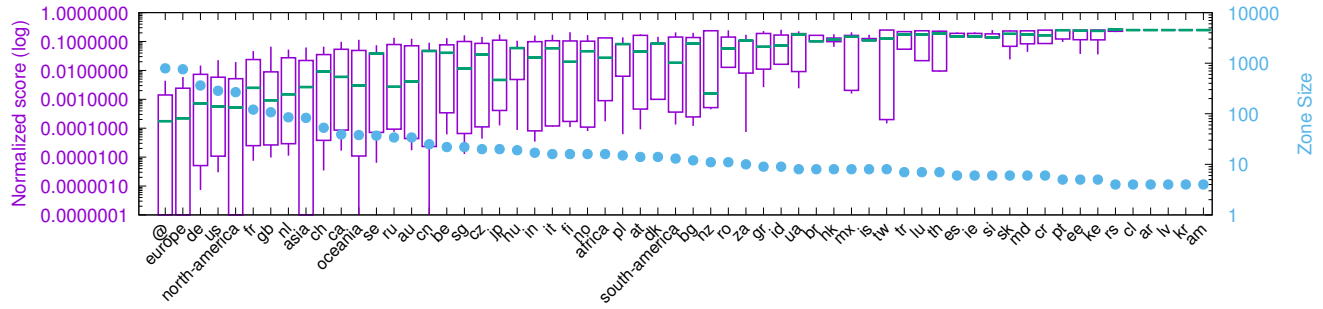

Figure 11: IPv6 NTP severs: Top 50 zones records weights distribution . Whiskers show *min* and *max* weight values, boxes shows IQRs, median shown as a green line.

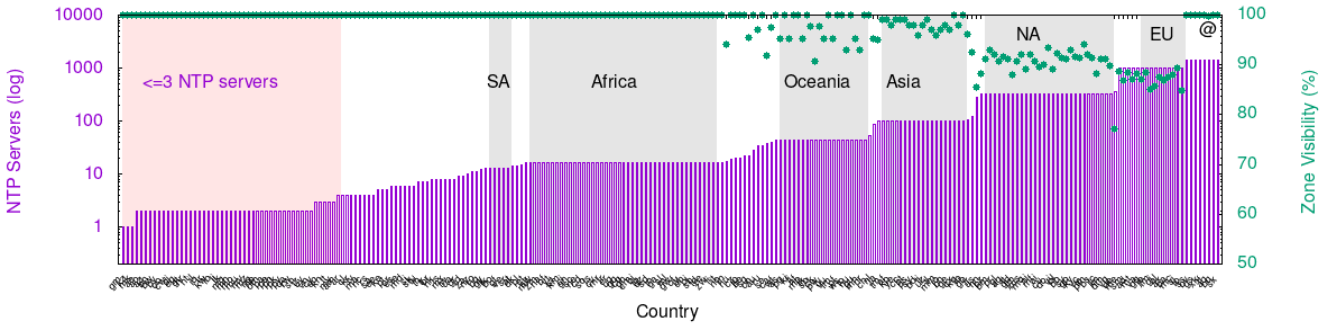

Figure 12: CCv6: Countries visibility of the `NTPPool`.

Table 16 shows the country codes, country names, and the IP addresses we used for each country on §5.

	countryIP	zoneMapped	subZoneSize
0	la	la	1
1	gg	gg	1
2	gh	gh	1
3	cm	cm	1
4	im	im	1
5	sy	sy	1
6	kg	kg	1
7	re	re	1
8	sn	sn	2
9	gt	gt	2
10	kw	kw	2
11	il	il	2
12	dj	dj	2
13	eg	eg	2
14	mo	mo	2
15	qa	qa	2
16	kh	kh	2
17	bh	bh	2
18	cw	cw	2
19	gi	gi	2
20	pa	pa	2
21	rw	rw	2
22	bd	bd	2
23	ht	ht	2
24	az	az	2
25	ge	ge	2
26	mz	mz	2
27	mn	mn	2
28	lb	lb	2
29	om	om	2
30	ng	ng	2
31	tz	tz	3
32	iq	iq	3
33	mu	mu	3
34	pe	pe	3
35	lk	lk	3
36	uy	uy	3
37	ph	ph	3
38	np	np	3
39	ba	ba	3
40	li	li	3

Table 14: Countries with fewer than 3 NTP servers (A records).

	countryIP	zoneMapped	subZoneSize
0	gm	gm	1
1	kz	kz	1
2	mk	mk	1
3	ae	ae	2
4	ao	ao	2
5	az	az	2
6	bh	bh	2
7	bw	bw	2
8	by	by	2
9	co	co	2
10	cw	cw	2
11	dj	dj	2
12	eg	eg	2
13	gt	gt	2
14	hr	hr	2
15	ht	ht	2
16	il	il	2
17	iq	iq	2
18	ir	ir	2
19	kw	kw	2
20	lb	lb	2
21	li	li	2
22	lk	lk	2
23	mg	mg	2
24	mn	mn	2
25	mo	mo	2
26	mu	mu	2
27	mv	mv	2
28	mz	mz	2
29	ng	ng	2
30	np	np	2
31	om	om	2
32	pa	pa	2
33	pe	pe	2
34	ph	ph	2
35	pk	pk	2
36	py	py	2
37	qa	qa	2
38	rw	rw	2
39	sn	sn	2
40	tz	tz	2
41	uy	uy	2
42	vn	vn	2
43	cy	cy	3
44	ec	ec	3
45	kh	kh	3
46	lt	lt	3
47	my	my	3

Table 15: Countries with fewer than 3 NTP servers (AAAA records).

CC	Country Name	IP address
ad	Andorra	34.99.136.1
ae	United Arab Emirates	2.16.158.1
af	Afghanistan	27.116.56.1
ag	Antigua and Barbuda	23.132.144.1
ai	Anguilla	69.57.224.1
al	Albania	2.58.82.1
am	Armenia	2.17.249.1
ao	Angola	41.63.160.1
aq	Antarctica	146.75.179.1
ar	Argentina	2.17.202.1
as	American Samoa	38.101.164.1
at	Austria	2.16.10.1
au	Australia	1.0.0.1
aw	Ecuador	45.4.88.1
ax	Åland	79.133.0.1
az	Azerbaijan	5.10.240.1
ba	Bosnia and Herzegovina	5.43.64.1
bb	Barbados	23.236.0.1
bd	Bangladesh	5.182.185.1
be	Belgium	2.17.107.1
bf	Burkina Faso	41.78.48.1
bg	Bulgaria	2.20.45.1
bh	Bahrain	3.5.220.1
bi	Burundi	2.18.11.1
bj	Benin	41.74.0.1
bl	Saint Barthélemy	90.31.74.1
bm	Bermuda	45.42.144.1
bn	Brunei	5.182.197.1
bo	Bolivia	34.100.4.1
bq	Bonaire_ Sint Eustatius_ and Saba	143.0.32.1
br	Brazil	2.16.15.1
bs	Bahamas	23.232.250.1
bt	Bhutan	5.182.196.1
bv	Bouvet Island	46.29.219.1
bw	Botswana	41.74.48.1
by	Belarus	5.100.192.1
bz	Belize	2.56.44.1
ca	Canada	2.22.72.1
cc	Cocos [Keeling] Islands	45.42.156.1
cd	DR Congo	5.175.77.1
cf	Central African Republic	41.78.120.1
cg	Congo Republic	41.75.64.1
ch	Switzerland	2.16.12.1
ci	Ivory Coast	41.61.12.1
ck	Cook Islands	14.137.40.1
cl	Chile	2.18.236.1
cm	Cameroon	2.16.134.1
cn	China	1.0.1.1
co	Colombia	4.33.232.1
cr	Costa Rica	8.242.196.1

cu	Cuba	57.74.110.1
cv	Cabo Verde	41.74.128.1
cw	Curaçao	23.209.95.1
cx	Christmas Island	104.224.54.1
cy	Cyprus	2.56.140.1
cz	Czechia	2.16.2.1
de	Germany	2.16.6.1
dj	Djibouti	41.189.224.1
dk	Denmark	2.16.63.1
dm	Dominica	45.74.22.1
do	Dominican Republic	23.140.80.1
dz	Algeria	5.180.66.1
ec	Ecuador	45.4.88.1
ee	Estonia	2.57.220.1
eg	Egypt	2.21.128.1
eh	Western Sahara	154.150.4.1
er	Eritrea	57.82.120.1
es	Spain	1.178.224.1
et	Ethiopia	45.59.144.1
fi	Finland	2.16.144.1
fj	Fiji	14.137.38.1
fk	Falkland Islands	80.73.208.1
fm	Federated States of Micronesia	43.248.156.1
fo	Faroe Islands	37.120.252.1
fr	France	1.179.112.1
ga	Gabon	41.72.224.1
gb	United Kingdom	2.16.14.1
gd	Grenada	23.208.167.1
ge	Georgia	2.57.60.1
gf	French Guiana	45.169.164.1
gg	Guernsey	5.62.84.1
gh	Ghana	2.16.77.1
gi	Gibraltar	2.58.8.1
gl	Greenland	37.18.44.1
gm	Gambia	41.76.8.1
gn	Guinea	41.77.184.1
gp	Guadeloupe	5.187.96.1
gq	Equatorial Guinea	41.79.48.1
gr	Greece	2.16.19.1
gs	South Georgia and the South Sandwich Islands	154.65.48.1
gt	Guatemala	24.152.52.1
gu	Guam	8.3.122.1
gw	Guinea-Bissau	45.42.213.1
gy	Guyana	57.74.246.1
hk	Hong Kong	1.32.205.1
hn	Honduras	45.4.84.1
hr	Croatia	2.58.32.1
ht	Haiti	45.74.24.1
hu	Hungary	2.58.168.1
id	Indonesia	3.5.35.1
ie	Ireland	2.16.138.1
il	Israel	2.22.233.1

im	Isle of Man	5.62.80.1
in	India	1.6.0.1
io	British Indian Ocean Territory	202.44.112.1
iq	Iraq	2.56.36.1
ir	Iran	2.144.0.1
is	Iceland	5.23.64.1
it	Italy	2.16.17.1
je	Jersey	5.35.160.1
jm	Jamaica	23.156.32.1
jo	Jordan	2.17.24.1
jp	Japan	1.0.16.1
ke	Kenya	14.137.168.1
kg	Kyrgyzstan	5.57.8.1
kh	Cambodia	1.32.252.1
ki	Kiribati	57.70.164.1
km	Comoros	41.194.33.1
kn	St Kitts and Nevis	23.131.208.1
kp	North Korea	175.45.176.1
kr	South Korea	1.11.0.1
kw	Kuwait	5.104.66.1
ky	Cayman Islands	23.188.0.1
kz	Kazakhstan	2.57.96.1
la	Laos	5.253.184.1
lb	Lebanon	5.8.128.1
lc	Saint Lucia	23.189.192.1
li	Liechtenstein	5.34.248.1
lk	Sri Lanka	23.49.160.1
lr	Liberia	41.57.80.1
ls	Lesotho	23.4.87.1
lt	Lithuania	5.20.0.1
lu	Luxembourg	2.17.201.1
lv	Latvia	2.58.16.1
ly	Libya	5.63.0.1
ma	Morocco	23.232.251.1
mc	Principality of Monaco	34.99.172.1
md	Moldova	2.56.0.1
me	Montenegro	31.204.192.1
mf	Saint Martin	38.99.116.1
mg	Madagascar	41.63.128.1
mh	Marshall Islands	45.59.139.1
mk	North Macedonia	5.32.176.1
ml	Mali	41.73.96.1
mm	Myanmar	23.199.72.1
mn	Mongolia	14.0.59.1
mo	Macao	17.91.136.1
mp	Northern Mariana Islands	8.3.112.1
mq	Martinique	41.77.244.1
mr	Mauritania	41.138.128.1
ms	Montserrat	104.224.6.1
mt	Malta	2.59.128.1
mu	Mauritius	41.72.213.1
mv	Maldives	5.62.61.1

mw	Malawi	41.70.0.1
mx	Mexico	8.14.224.1
my	Malaysia	1.9.0.1
mz	Mozambique	41.76.0.1
na	Namibia	41.63.192.1
nc	New Caledonia	27.122.0.1
ne	Niger	41.78.116.1
nf	Norfolk Island	103.43.204.1
ng	Nigeria	8.35.57.1
ni	Nicaragua	45.5.216.1
nl	Netherlands	2.16.1.1
no	Norway	2.18.172.1
np	Nepal	14.137.53.1
nr	Nauru	43.230.6.1
nu	Niue	49.156.48.1
nz	New Zealand	5.181.67.1
om	Oman	5.21.0.1
pa	Panama	5.252.152.1
pe	Peru	8.24.244.1
pf	French Polynesia	43.249.176.1
pg	Papua New Guinea	14.137.32.1
ph	Philippines	1.37.0.1
pk	Pakistan	14.1.104.1
pl	Poland	2.16.172.1
pm	Saint Pierre and Miquelon	70.36.0.1
pr	Puerto Rico	12.205.64.1
ps	Palestine	2.58.132.1
pt	Portugal	2.16.65.1
pw	Palau	57.70.176.1
py	Paraguay	24.152.40.1
qa	Qatar	2.23.168.1
re	Réunion	5.57.96.1
ro	Romania	2.17.116.1
rs	Serbia	5.22.160.1
ru	Russia	1.2.9.1
rw	Rwanda	41.74.160.1
sa	Saudi Arabia	2.59.54.1
sb	Solomon Islands	14.137.34.1
sc	Seychelles	2.56.10.1
sd	Sudan	41.67.0.1
se	Sweden	2.16.66.1
sg	Singapore	1.32.128.1
sh	Saint Helena	104.224.22.1
si	Slovenia	5.32.136.1
sj	Svalbard and Jan Mayen	45.59.151.1
sk	Slovakia	2.57.64.1
sl	Sierra Leone	41.78.84.1
sm	San Marino	31.193.32.1
sn	Senegal	41.82.0.1
so	Somalia	41.78.72.1
sr	Suriname	45.74.20.1
ss	South Sudan	41.79.24.1

st	São Tomé and Príncipe	45.42.228.1
sv	El Salvador	45.5.12.1
sx	Sint Maarten	131.161.84.1
sy	Syria	5.0.0.1
sz	Eswatini	5.175.76.1
tc	Turks and Caicos Islands	63.130.249.1
td	Chad	41.74.32.1
tf	French Southern Territories	45.59.175.1
tg	Togo	41.78.136.1
th	Thailand	1.0.128.1
tj	Tajikistan	37.98.152.1
tk	Tokelau	27.96.24.1
tl	East Timor	43.254.56.1
tm	Turkmenistan	45.59.159.1
tn	Tunisia	41.62.0.1
to	Tonga	14.137.33.1
tr	Turkey	2.16.150.1
tt	Trinidad and Tobago	23.3.72.1
tv	Tuvalu	14.137.42.1
tw	Taiwan	1.32.194.1
tz	Tanzania	2.17.250.1
ua	Ukraine	2.21.89.1
ug	Uganda	2.17.248.1
us	United States	2.19.128.1
uy	Uruguay	2.18.64.1
uz	Uzbekistan	5.101.221.1
va	Vatican City	45.42.143.1
vc	Saint Vincent and the Grenadines	23.170.80.1
ve	Venezuela	8.242.232.1
vg	British Virgin Islands	2.56.144.1
vi	U.S. Virgin Islands	8.26.16.1
vn	Vietnam	1.52.0.1
vu	Vanuatu	14.137.37.1
wf	Wallis and Futuna	27.125.192.1
ws	Samoa	43.241.164.1
xk	Kosovo	185.244.25.87
ye	Yemen	5.100.160.1
yt	Mayotte	41.242.116.1
za	South Africa	2.16.140.1
zm	Zambia	41.60.0.1
zw	Zimbabwe	41.57.64.1

Table 16: List of Country codes, Country Names, and IP addresses.

Table 17 shows the number of time providers per country.

country	nASesV4	nASesV6
@	974	403
ae	2	1
africa	25	11

am	3	3
ao	3	1
ar	4	2
asia	122	51
at	29	11
au	28	16
az	1	1
ba	2	0
bd	1	0
be	13	10
bg	22	8
bh	1	1
br	15	6
bw	1	1
by	5	2
ca	40	18
ch	34	20
cl	3	2
cm	1	0
cn	23	15
co	5	1
cr	4	2
cw	1	1
cy	2	2
cz	16	10
de	81	63
dj	1	1
dk	21	9
ec	2	2
ee	5	3
eg	1	1
es	6	2
europe	640	274
fi	13	6
fr	35	21
gb	68	40
ge	1	0
gg	1	0
gh	1	0
gi	1	0
gm	0	1
gr	10	6
gt	1	1
hk	12	5
hr	4	1
ht	1	1
hu	13	4
id	11	8
ie	7	4
il	1	1
im	1	0
in	11	6

iq	2	1
ir	4	2
is	5	4
it	14	9
jp	18	12
ke	2	2
kg	1	0
kh	1	2
kr	6	4
kw	1	1
kz	5	1
la	1	0
lb	1	1
li	1	1
lk	2	1
lt	7	3
lu	9	5
lv	9	3
ma	2	2
md	5	3
mg	3	1
mk	3	1
mn	1	1
mo	1	1
mu	2	1
mv	3	1
mx	6	2
my	6	2
mz	1	1
nc	2	2
ng	1	1
nl	57	35
no	16	10
north-america	212	85
np	2	1
nz	19	7
oceania	45	22
om	1	1
pa	1	1
pe	2	1
ph	2	1
pk	2	1
pl	34	7
pt	6	3
py	2	1
qa	1	1
re	1	0
ro	11	6
rs	8	2
ru	82	17
rw	1	1
sa	2	2

se	17	10
sg	15	12
si	5	1
sk	10	5
sn	1	1
south-america	27	8
sy	1	0
th	13	4
tj	2	0
tr	11	4
tw	9	5
tz	2	1
ua	35	4
us	177	73
uy	2	1
uz	2	0
vn	4	1
za	14	8

Table 17: Time providers per country.