Discovery of Network Time Servers through Domain Names

ROLF VAN KLEEF, University of Twente, The Netherlands
PIETER-TJERK DE BOER, University of Twente, The Netherlands
PASCAL HUPPERT, University of Twente, The Netherlands

Having a secure, stable, and reliable infrastructure for distributing time across the Internet is critical to its overall security. Therefore, we wish to discover and analyze Network Time Protocol (NTP) servers that are available and/or used. The method propsed and executed in this research is to scan the Domain Name System for common places references to NTP servers might exist, and then validating those references, resulting in a list of NTP servers. Some initial analysis of this list of servers is also done in order to draw some conclusions about, for example, IPv6-readiness of the NTP ecosystem.

CCS Concepts: • **Networks** → **Time synchronization protocols**; **Network measurement**; **Naming and addressing**; *Public Internet*; Network monitoring.

Additional Key Words and Phrases: DNS; NTP; IPv6; Measurements

1 INTRODUCTION

Many security mechanisms, distributed systems, and authentication protocols depend on synchronized clocks to function correctly [21, 26]. The Network Time Protocol (NTP) is the de-facto standard protocol used to achieve this synchronization, and it plays a foundational role in maintaining trust and reliability across the internet.

Despite the importance of NTP infrastructure, not all of it is well-documented or well-managed. Misconfigurations can lead to vulnerabilities such as DDoS amplification attacks [1, 32], while outdated deployments may lack support for modern technologies like IPv6 or Network Time Security (NTS). As a result, there is ongoing interest in measuring and monitoring the availability, configuration, and distribution of NTP servers.

Most previous efforts to discover NTP servers have relied on scanning the IP address space [30]. However, this approach is increasingly impractical in the IPv6 era due to the vastly larger address space as compared to IPv4 [29]. Another method for discovering NTP servers is the use of peering information that can be obtained via mode 6 control methods in order to locate other NTP servers [12], though this method is not future-proof as this control should be disabled for public use as it is a possible DDoS amplification vector. An alternative and largely underexplored approach is to leverage the Domain Name System (DNS) as a discovery mechanism.

This research investigates the feasibility of using DNS zone enumeration to discover publicly accessible NTP servers. By probing large domain datasets and scanning for well-known time-related subdomains[1], we aim to build an NTP server dataset that complements datasets created through traditional IP scanning, and provides insights on NTP servers running IPv6. Furthermore, this research evaluates how DNS-based discovery methods can reveal otherwise hidden relationships and improve insight into IPv6 support, naming practices, and deployment patterns within the NTP ecosystem.

2 BACKGROUND

This section provides the necessary background on key technologies related to our research: the Network Time Protocol (NTP), the Domain Name System (DNS), and the importance of the transition from the Internet Protocol (IP) version 4 (IPv4) to IP version 6 (IPv6).

2.1 The Network Time Protocol

NTP [22] servers are a critical part of internet infrastructure, as accurate time-keeping is essential for the internet to function. Malfunctioning NTP infrastructure can lead to security issues, operational issues, and functional issues [15, 21, 32]. NTP serves functions beyond informing users of the current time. It is primarily designed as a reliable time reference for synchronization within computer systems, for safeguarding against attacks in network security protocols, to support time-based authentication protocols such as TOTP [26], and to aid in synchronization in distributed systems.

NTP can also be a significant source of problems on the internet. Misconfigured NTP servers can be used as a so-called DDoS amplification vector [15]. A good understanding of the NTP infrastructure is important for both the appreciation of its critical role in modern internet infrastructure, and to identify potential problems and misuse.

NTP obsoletes the earlier Time Protocol [28], a very simple, but severely limited protocol meant for dissemination of time. The primary problem with the Time Protocol was that it was not accurate and precise, as the maximum precision was a second, and the protocol did not correct for network transit time. NTP corrects for that, but also adds additional functionality like peer-to-peer synchronization between two servers, optional encryption to add an additional layer of security, and provides additional metadata. NTP can usually provide time accurate to several milliseconds, or better if care is taken in the configuration and the network conditions are sufficient.

There are three operational modes of NTP that are relevant for this research. Most important are the mode 3, named “Client” mode and mode 4, named “Server” mode.

TScIT 43, July 4, 2025, Enschede, The Netherlands

[1]We use the term “subdomains” in the case where we talk about a fully-qualified domain name (FQDN) consisting of a domain prefix, a registered domain (which, as mentioned later, is usually a second-level domain), and a top-level domain

An NTP client will periodically send mode 3 queries to an NTP server, which then returns a mode 4 "Server" response. The mode 4 response will then allow the client to determine the precise time offset of the NTP server that was queried. NTP mode 6 is is also mentioned in this research. It is a mode reserved for control of the NTP server, intended to allow clients to determine operational information about an NTP server, and to allow clients to alter settings and state of an NTP server.

Each NTP server has an attribute called "Stratum". This is an integer that indicates the distance between it and the closest reference time source that is used. For example, a stratum 1 NTP server is directly connected to a reference time source like a GPS receiver or a high-precision atomic clock, while a stratum 2 servers use a stratum 1 server as a time reference. A higher stratum time server may indicate a lower accuracy time compared to a stratum 1 server with a properly deployed and configured time reference, though higher-stratum time servers are easier to deploy as deploying a reference time source for them is unnecessary.

As assigned by IANA, NTP by default operates on its assigned port `123` on UDP [6].

2.2 The Domain Name System

The Domain Name System [24] is a distributed, hierarchical directory system for the whole internet. The primary purpose of the DNS is to convert names that humans can remember and spell into machine-readable IP addresses and associated metadata.

The DNS does name translation through a hierarchical structure of zones. Each zone has one or more nameservers that provide the information for the zone. Resolving starts at the DNS root zone, which is maintained by IANA in their Root Zone Database [5]. This root zone contains pointers to delegated zones like `com.`[2] and `net.`. This continues recursively, each zone delegating to their child zones where necessary. At second level zones, the nameservers also start containing particular information that is relevant for this research, as some of that information is relating to specific services that we may want to identify.

The information in the DNS is divided up into Resource Records (RRs) of different types. These types are maintained by IANA in the DNS Parameters Registry [2], most interestingly for us being:

- **A and AAAA Resource Records**: Address records map a domain name to an IP address. A records return IPv4 addresses, and AAAA records return IPv6 addresses.
- **CNAME Resource Records**: Canonical Name records alias one fully-qualified domain name to another fully-qualified domain name. They require another full lookup for the new domain name in order to resolve the final IP address.
- **SRV Resource Records**: Service records are records that provide information about a specific service type. They specify the target domain name and port number where this service should be available. For example, a service record located at `_ntp._udp.utwente.nl.` provides information about the NTP service for the `utwente.nl` domain.
- **NS Resource Records**: Name Server records are records that contain the zone delegation information. This record indicates to resolvers that queries into the zone this record is located at, and the sub-zones of that, must be delegated to the nameserver that this record indicates.
- **PTR Resource Records**: Pointer records are used for reverse DNS lookups. They perform exactly the opposite of A and AAAA in the DNS, that being translating IP addresses back into domain names. PTR records are placed in specific zones, called reverse lookup zones, depending on the IP family they service. For IPv4, this is `in-addr.arpa.`, and for IPv6, this is `in6.arpa.`. For IPv4, each byte is converted to decimal, and placed in right-to-left order in the FQDN. For example, IP address `1.2.3.4` would be associated with the reverse DNS name `4.3.2.1.in-addr.arpa.`. For IPv6, the process is similar, however, the hierarchical steps are split on nibbles, instead of bytes, resulting in an IPv6 address of, for example `fe80::598a:eed0:7327:10c1` to be associated with the reverse DNS name `1.c.0.1.7.2.3.7.0.d.e.e.a.8.9.5.0.0.0.0.0.0.0.0.0.0.0.0.0.0.8.e.f.in6.arpa.`.
- **NSEC3 Resource Records**: Next Secure Level 3 records contain information for securing the DNS infrastructure through a mechanism called DNSSEC [19]. They can cryptographically prove the non-existence of specific ranges of DNS names in a zone [8].

The DNS has a sub-protocol for transferring zones between different nameservers [20]. This process is called zone-transferring. If available, this allows clients to enumerate zones that are running on a nameserver. This sub-protocol is, however usually heavily restricted in accessibility as it is often not desirable for clients to be able to enumerate entire zones. Some reasons for disabling this is written in RFC 3833 devoted to Threat Analysis of the DNS [10].

A domain prefix[3] is the part of a domain name that is prefixed a higher-level domain, usually appearing as direct children of a registered domain. Some conventions have informally developed that assign common domain prefixes to well-known services. For instance, websites are usually available under the `www`[4] domain prefix, SMTP servers are often available under `smtp`, or `mail`. Of particular interest to us are the domain prefixes typically allocated to time servers. These are commonly available under `time`, `ntp`, `tick`, `tock`, and several others.

This consistent naming allows us to do a more targeted search for specific services, by querying the prefixes of registered domains that are conventionally used for that service type.

[2]We use the convention of a trailing dot denoting an absolute DNS name, and no trailing dot being a relative DNS name.

[3]We use the term "domain prefix" here, where commonly the term subdomain is used. This is because the term subdomain is ambiguous regarding whether a domain component or an FQDN is meant. In this case, a domain prefix is a single name component of an FQDN.

[4]Note that we have no trailing dot here, so this is, as explained before, a relative name, a prefix.

2.3 IPv6 transition

The transition from IPv4 to IPv6 is one of the most significant ongoing changes to the internet's underlying infrastructure. IPv4, with its 32-bit addressing space, provides approximately 4.3 billion unique addresses, an insufficient number given the scale of modern internet-connected devices [11, 16]. IPv6 solves this limitation by expanding the address space to 128 bits, enabling an effectively inexhaustible number of unique IP addresses [18].

This transition has far-reaching consequences for internet services. As IPv6 adoption increases, services like NTP should also transition more to using IPv6, to make optimal use of the advantages that IPv6 offers.

However, IPv6 deployment of many services is far from available. While some providers and networks have embraced IPv6, others remain IPv4-only or only partially support IPv6. This disparity presents challenges in ensuring consistent service availability across protocol versions. Investigating how well essential services like NTP support IPv6 is therefore crucial in assessing the readiness and robustness of this ongoing evolution.

3 PROBLEM STATEMENT

No recent studies have looked into finding a large amount of NTP servers, and certainly not through the DNS.

The motivation for discovering a large number of NTP servers is to enable further research into their deployment, configuration, security, and operational characteristics. Some initial applicable information can also be obtained with this initial discovery of NTP servers to, for example, analyze the IPv6-readiness of NTP infrastructure, and to determine some operational and ownership information of individual NTP servers.

3.1 Research Questions

Main RQ: How is scanning for NTP servers through DNS probing advantageous to more traditional scanning methods like IP-based host scans?

3.1.1 Sub-questions.

RQ1: How do we enumerate the DNS in order to find NTP servers?

RQ2: Can we find NTP servers that we expect won't be found through scanning without using the DNS?

RQ3: What fraction of NTP servers support IPv4 versus IPv6, and what fraction of domains that operate NTP support IPv4 versus IPv6?

4 RELATED WORK

There has been some work on discovering NTP servers. Minar has done a search for NTP servers through issuing peering queries to several known NTP servers [23], though as mentioned before, this is no longer a viable option. Among many others, Cao and Veich have also done a qualitative comparison between several already well-known NTP servers [12]. Rytilahti et al. have analyzed some interesting information on how the NTP pool by `ntppool.org` is being used, what servers are available in the NTP pool and how clients behave toward this pool [30]. The research on the NTP pool [25] by Moura et al. is also relevant, especially for the pool exploration component of our research.

There are several large datasets of DNS information. A very large one is OpenINTEL [3, 34], which is a dataset based on knowledge of delegated zones[5], so this datset exclusively contains information about the delegated zones and registered domains[6]. A dataset that is more immediately available, and has some other useful information like more subdomains is Tranco [4, 27], however this dataset, is problematic due to the way it is collected and due to the that it is, compared to OpenINTEL, miniscule.

5 METHODOLOGY

The primary objective being the development of a method for the discovery of NTP servers through the probing of the DNS, the research method and the results are tightly intertwined. This section outlines the steps taken to collect, filter, and analyze DNS and NTP data, along with specific strategies for addressing each research question. The source code for the scanning method is publicly available in a git repository hosted at https://codeberg.org/rhbvkleef/finding-ntp.

5.1 Method for answering RQ1

How do we enumerate the DNS in order to find NTP servers?

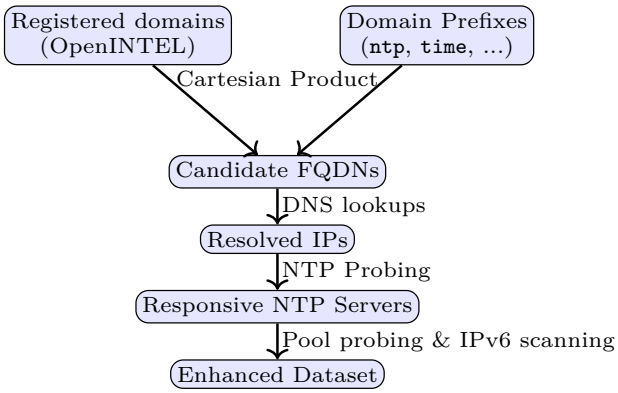

Fig. 1. Compact pipeline for DNS-based NTP server discovery, showing data state per step

The DNS has a limitation that will be very important for this research: DNS lookups can only be done for exact domain names, except for when domain transfers are enabled. As DNS servers typically have domain transfers (through the `AXFR` and `IXFR` lookups) disabled, we cannot query DNS servers for their entire zone file, or even for a list of domain prefixes for a particular domain. A server could leave this capability of the DNS enabled, but that is widely considered unsafe [33]. Due to the widespread deactivation of zone transfers, this research has not attempted the use this method.

[5]We use the term "delegated zone" for the zone associated with a delegated domain. We use the term "delegated domain" for domains that are directly below a public suffix (or as we refer to them, top-level domains).

[6]We use the term "registered domain", as it is the most generally applicable terms, even though they are uncommonly used.

Because of this limitation of the DNS, only two strategies remain, namely to guess domains and domain prefixes, or to find some other source of this information. For top-level domains[7] and registered domains, there are databases we can use. Top-level domains are kept track of by the Internet Assigned Numbers Authority in their Root Zone Database [5]. For delegated domains, there are data sources such as OpenINTEL [3, 34]. OpenINTEL is a project that works together with registry operators like SIDN and VeriSign to obtain large sets of delegated zone information for the purposes of research.

We can easily query all registered domains for presence of NTP servers. This is, however, not enough as many NTP servers are expected to be located at prefixes of registered domains as described in section 2.2. We therefore need to create a list of domain prefixes where these NTP servers may reside.

Some methods that have been used to create this list are:

- to look at public lists of NTP servers and their domains, and extract the domain prefixes; and
- to make educated guesses of logical domain prefixes.

The DNS scanner uses publicly available resolver software (The Internet System Consortium's Bind9 in our case [14]) to do DNS resolution. For each subdomain and for the registered domain, we scanned several resource record types:

- **A** and **AAAA** records to identify IP addresses;
- **CNAME** records, whose targets we treat the same as the delegated domains in initial input list, namely that we query them with our list of domain prefixes too; and
- **SRV** records, specifically `_ntp._udp.domain.`, to discover NTP servers that are pointed to by these records.

The process of CNAME and SRV recursion is expressed in simplified code in Listing 3.

The NTP scanner iterates through a list of IP address and port pairs generated by the DNS scanner, and queries each distinct IP address and port pair using an NTP mode 3 query. If the scanner received a mode 4 response from an IP address and port pair, an NTP server is detected on that IP address and port pair. If after 1 second, no response has been received, no NTP server is detected on that IP address and port pair. Due to the simplicity of NTP, these mode 3 queries can be sent at a very high rate, so this step of the scan can be executed very swiftly.

This method required almost all of the individual components of them to be automated, as large-scale probing of the DNS is infeasible to do by hand. Specifically, all DNS probing and all NTP probing had to be fully automated in order to feasibly generate a large dataset.

Despite the automation, the process of scanning large zones takse much too long for this research. It is expected that with the current set-up, scanning the entire `com.` zone would take over a hundred days. Due to this, the `com.` zone has been randomly sampled in order to still be able to create a useful dataset. This random sample consists of 40,000,000 registered domain and prefix pairs, which amounts to approximately 2.7% of the entire `com.` zone. The `se.` zone has been scanned in it's entirety, consisting of approximately 51,400,000 registered domain and prefix pairs.

Once all the the initial scan was done, some enhancement scans were done, like the exploration of suspected pools of NTP servers, further analysis as explained later in this chapter. Pools of NTP servers, like the NTP pool at `ntppool.org` [9] are fully qualified domain names that resolve a large amount of IP addresses of NTP servers. In the case of `pool.ntp.org`, this is done with a piece of software called GeoDNS [17], which can, depending on the client location, return different geo-local results. Moreover, it can return different results for similar locales of clients as a way to do some load-balancing for NTP servers. This allows such pools to distribute server load and provide low latency connections to clients, so that a large amount of clients can be serviced efficiently.

All in total, this described method closely resembles the process as drawn in Figure 1, namely, to start with obtaining a list of registered domains, then create a cartesian product with that list and the list of domain prefixes we wish to query, then resolve all of those for the 4 types of resource records, and then query all of the discovered IP address and port pairs for the presence of an NTP server by sending them an NTP mode 3 query and awaiting a mode 4 response.

5.2 Method for answering RQ2

Can we find NTP servers that we expect won't be found through scanning without using DNS?

In order to determine whether NTP servers that we find might not be found through IP scans, we looked at the IP-address and port at which an NTP server resides. If an NTP server resides on a port other than `123`, IP scans will likely not discover them, as they generally only query port `123`, which, as pointed out in Section 2.1, is NTP's default assigned port. These NTP servers are only likely to be discovered through `SRV` lookups through the DNS, or through peering analysis of an NTP network.

Another way that IP scans might not find NTP servers is if NTP servers are only available on a very small IPv6 address range. IPv6 services may respond to many IPv6 addresses, potentially even 2^{64} addresses, as IPv6 subnets with a 64-bit prefix are commonly handed out [7]. Even then, scanning 2^{64} addresses is already infeasible. It is therefore useful to try to determine on how many IPv6 addresses an NTP server might respond. Because these IPv6 address ranges are likely to be contiguous, we can scan IP addresses that are contiguous to the already known IP address as returned by the DNS in an effort to determine on how many addresses an NTP server might be listening. This part of the process is later referred to as IPv6 lookaround. Listing 1 shows an approximate procedure that could be used to do this IPv6 lookaround.

5.3 Method for answering RQ3

What fraction of NTP servers support IPv4 versus IPv6, and what fraction of domains that operate NTP support IPv4 versus IPv6?

[7] We use the term "top-level domain" even though this is potentially incorrect in the general case. The more general term would be "public suffix"

A result of RQ1 is a list of NTP servers, and all discovered registered domains and subdomains associated with them. Querying this data in order to obtain the data required for answering this question is trivial, simply requiring us to:

- select all discovered NTP servers, and counting how many of them support IPv4 versus IPv6; and
- select all domains pointing to at least one NTP server, and counting how many of them contain IPv4 and IPv6 servers.

6 PRACTICAL CONSIDERATIONS

6.1 Query reduction

As querying the DNS is the slowest part of our scans, and potentially disruptive to authoritative nameservers, we needed to develop a strategy for reducing the amount of DNS queries we execute. We do this by employing a step-wise querying strategy:

(1) Query a subset of domain prefixes, specifically excluding the prefixes `ntp2`–`ntp4`, `time2`–`time4`, and `time-b`–`time-f`.
(2) For the domains that resolve the initial list, also query the excluded prefixes.

Moreover, DNS servers often return AAAA and CNAME resource records when present, even when a lookup for the A record type is done, so performing a lookup for an A record first, and skipping the AAAA and CNAME lookups when we get AAAA or CNAME results reduces the amount of executed queries too. Listing 2 shows an approximate procedure for skipping lookups that have been returned by previous lookups for other resource record types.

Lastly, we employ as much caching as possible, to reduce the need for re-executing queries at a later time for resources that can be re-used, like CNAME targets and nameserver locations.

6.2 DNS Recursion is Slow

As observed in section 6.1 about query reduction, DNS scanning is the slowest part of the overall scanning process. This has been solved by relying on industry-standard resolver software (namely, Bind9 [14]) to offload the querying, recursion, and caching of results, as well as running many instances of our scanner to scan concurrently.

6.3 Aggressive DNS Rate Limiting is Common

The next problem that was encountered was DNS rate limiting. Even though the initial scanning already employed some primitive methods for avoiding overloading servers, the discovery was made that many authoritative DNS servers have very low rate limits. This problem was solved by thoroughly shuffling the input domains and prefix pairs, so that domain prefixes to a single registered domain are not queried in rapid succession.

7 RESULTS

It has turned out that large-scale probing of the DNS is very slow and difficult. This has had a significant impact in the development of the scanning method, but also on the scale of the measurements that have been done. The results shown below were obtained using scans done between the 4th and the 13th of June 2025 for the `com.` zone, and between the 24th and 28th of May 2025 for the `se.` zone.

7.1 Results regarding Research Question 1

Research Question 1 has yielded the most results, as it was the most fundemental and involved question of them. Below are components of the answer to the question, that combined with the methodology give an answer to the question "How do we enumerate the DNS in order to find NTP servers?".

7.1.1 Scanning performance. The scan rate measured with the current scan tooling is 10,000,000 registered-domain and prefix pairs per approximately 15 hours. The whole `com.` zone that was used, which was measured by OpenINTEL on June 2, 2025, and combined with our own prefix set, contains 1,859,798,448 registered domain and prefix pairs, so scanning the entire sone would take approximately 2790 hours, or 116 days. The scanning was primarily CPU limited, so increasing the amount of CPU resources available should significantly increase the speed of scanning. We used 4 cores from a AMD EPYC 9534 CPU, and if we were to allocate all of its 64 cores, we would expect the process to go approximately 16 times as fast, resulting in an expected measurement duration of just over 7 days, which is a very reasonable duration to be measuring for. Optimizing the DNS scanner is unlikely to speed up the scanning more than an order of 2, as a large part of the CPU usage was taken up by the Bind9 resolver, which is a factor which is largely out of our control.

7.1.2 Encountered Rate Limits. During the DNS probing, rate limits were encountered. Unfortunately, no measurement was made on how many rate limits were encountered, however by observing the logs, we can determine that rate limits were primarily encountered in the CNAME and SRV recursion stages, and were even then very rare.

Prefix	Count	Prefix	Count	Prefix	Count
(none)	16,142	ntp0	3,300	time0	3,251
www	14,123	time-a	3,298	tock	3,212
ntp	3,435	tick	3,297	ntp1	3,189
time	3,339	time1	3,280	pool	1,836

Table 1. Server counts of NTP servers for prefixes in the `com.` zone

Prefix	Count	Prefix	Count	Prefix	Count
(none)	3,352	time4	608	time-f	598
www	3,036	tick	607	ntp4	597
ntp	673	time1	606	time3	594
ntp1	631	time-d	606	time-e	593
time	628	time-a	605	time-c	591
ntp3	616	ntp2	605	time0	585
tock	614	time-b	603	ntp0	584
pool	613	time2	601		

Table 2. Same as Table 1, but for the `se.` zone

7.1.3 Domain Prefixes. We measured a set of 23 domain prefixes, listed in Table 3. The NTP servers discovered for

Prefixes			
ntp	time	time-a	*none*
ntp0	time0	time-b	www
ntp1	time1	time-c	tick
ntp2	time2	time-d	tock
ntp3	time3	time-e	pool
ntp4	time4	time-f	

Table 3. List of prefixes that were scanned

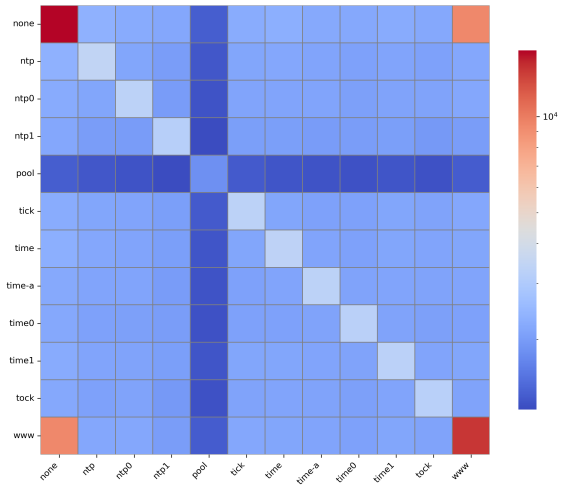

Fig. 2. Correlation of `com.` prefixes discovering the same NTP servers

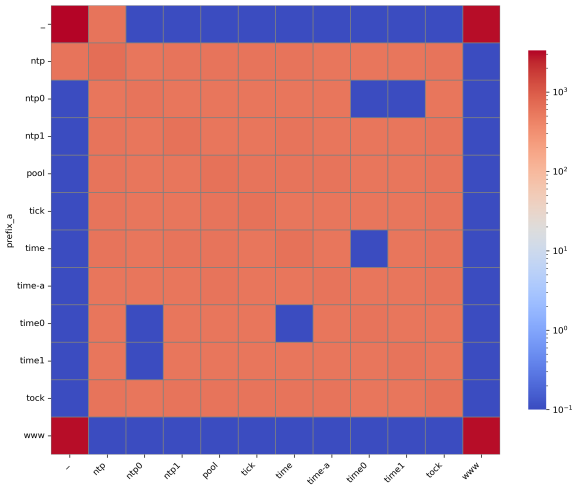

Fig. 3. Same as Figure 2, but for the `se.` zone

each domain prefix for each zone are listed in Table 1 and Table 2. They have significant amount of duplication in the detected NTP servers, as Figure 2 and Figure 3 show, though as we can clearly tell, the base registered domain and the `www.` domain prefix yield the most NTP servers, though also have a very significant overlap with each other. The `pool` prefix being less popular in the `com.` zone is an interesting result. What the practical reason for this is, is unknown and merits further research.

7.2 Results for Research Question 2

Two avenues were explored that might make traditional IP scanning impractical for the discovery of some NTP servers. Those were either IPv6 NTP servers that are impractical to discover due to the extremely large amount of IPv6 addresses that need scanning, and NTP servers that run on ports other than the standard port `123`.

The look-around procedure as described in the methodology has determined that none of the NTP servers running IPv6 that were discovered listen to more than 8 consecutive IPv6 addresses. Sampling some servers that appeared to be listening to multiple IPv6 addresses showed that, at least from the sample, all of them appear to actually be different servers listening to the different IP addresses, telling by their stratum, root delay and root dispersion values, so it is likely that none of the servers actually listen to more than 1 IPv6 address. Such a small range, or even a range of just 1, would mean that a scan through IPv6 would be entirely infeasible.

None of the `SRV` resource records that have been queried yielded NTP servers that listened to a non-standard UDP port, so on the IPv4 side, it is entirely feasible to discover all NTP servers through IP scanning.

7.3 Results for Research Question 3

Research Question 3 asks how ready the NTP ecosystem is to support IPv6. In the course of our measurements, a significant amount of information was obtained that allows us to give some insight into this. The answer is two-fold, analyzing both the raw counts of IPv4 versus IPv6 capable NTP servers, and analyzing the amount of domains operating either IPv4 or IPv6 NTP servers, or NTP servers on both IPv4 and IPv6.

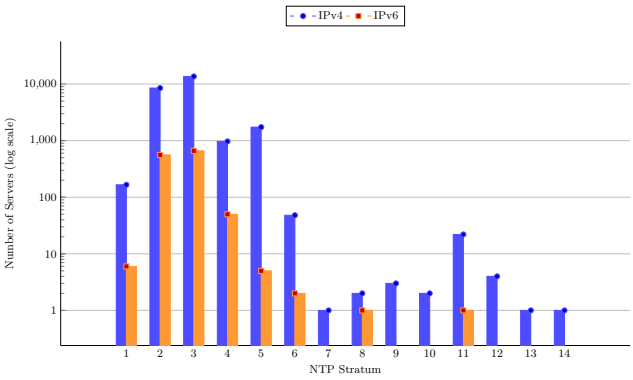

Fig. 4. Discovered NTP servers through the `com.` zone by stratum and IP family

As can be seen in Figure 5 and Table 4, the number of discovered IPv6 NTP servers through the `se.` and `com.` TLDs is vastly lower than the amount of IPv4 servers amounting to about 5% of all discovered NTP servers in both cases. Moreover, Table 5 and Table 4 show that an even larger proportion of domains do not support IPv6.

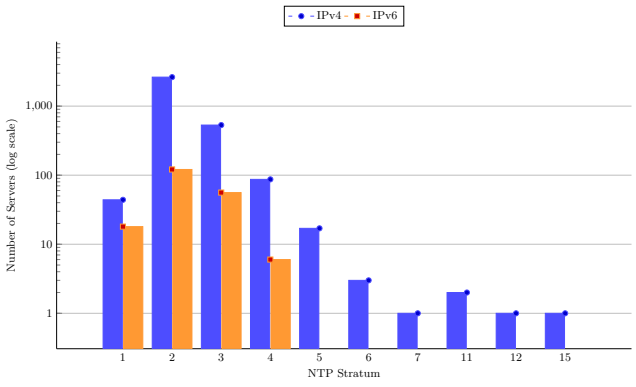

Fig. 5. Same as Figure 4, but for the `se.` zone

IP family	Count	Proportion
IPv4 only	292,157	98.4%
IPv6 only	898	0.3%
Both	3,819	1.3%
Total	296,874	100%

Table 4. Domains supporting NTP by IP family in the `com.` zone

IP family	Count	Proportion
IPv4 only	62,013	98.5%
IPv6 only	131	0.2%
Both	915	1.5%
Total	63,059	100%

Table 5. Same as Table 4, but for the `se.` zone

7.4 Strata

Unrelated to any research questions, we also took a look at the NTP stratum distribution. According to our measurements as shown in Figure 4 and Figure 5, stratum 2 and 3 are by far the most common strata in the measured part of the ecosystem. In the `com.` zone, 88.5% of the NTP servers found are either stratum 2 or stratum 3, while for the `se.` domain, this fraction is even higher, at 94.9%. The stratum distribution on IPv4 and IPv6 is roughly similar, with both IPv4 and IPv6 peaking on stratum 2 and 3.

8 DISCUSSION

The results show that DNS-based discovery of NTP servers is a viable approach, albeit with several caveats and challenges. This section discusses key findings and methodological concerns that might have impacted the quality of the obtained data.

8.1 CNAME Resolution Potential Bias

We follow CNAME and SRV records by recursing using their target and immediately resolving all known prefixes consecutively for each recursion step. This differs from how we sample the registered domains from our input list, where we try to space out resolving of domain prefixes for each domain in order to avoid rate-limits. Due to this difference, it is possible that for the domains that we recurse into, we encounter more rate limits, and thus cause our data to skew.

8.2 Inaccuracies in IPv6 Lookaround Estimation

The lookaround method for estimating how many IPv6 addresses an NTP server listens on is prone to overestimation. The scans only look for whether a response is returned, and does not attempt to identify whether the same server is responding. This causes the scan to indicate that several servers that appear at consecutive IP addresses are one server listening to multiple IP addresses. Moreover, it is possible that NTP servers are listening to non-consecutive IP addresses, possibly causing both over- and underestimation of the amount of IPv6 addresses an NTP server is listening on.

9 ETHICAL CONSIDERATIONS

This research involved scanning publicly accessible DNS resource records and probing discovered IP addresses for NTP service availability. While all activities were limited to data that is generally considered public or non-sensitive, several ethical considerations were taken into account.

- In order to minimize disruptions to the DNS ecosystem, domains were not scanned in sequence, but in a random order. This spreads out the load on individual authoritative nameservers so that they do not experience high peaks of traffic due to our scans.
- Some thought had to be put into whether the NTP scanning could be disruptive to the NTP servers being scanned, but as NTP queries are limited to one per IP address, and NTP queries are very light-weight, this was not considered problematic.
- If, despite the measures taken to minimize disruptions, disruptions were caused, a contact point was clearly advertised on the scanner machine. No complaints were submitted.
- Releasing the list of discovered NTP servers without restrictions might expose previously unknown and vulnerable NTP servers to the scrutiny of bad actors, so doing so should not be done.

10 FUTURE WORK

This research has primarily explored methods for DNS scanning. It creates a significant amount of potential avenues of research as a new kind of dataset is generated that has not been explored to its fullest in this research.

10.1 Enhancement of Domain Prefix List

Minimal work has been done to create a complete domain prefix list for scanning. Except for looking through public lists of NTP servers, and extracting the most common prefixes, and some guessing and thinking, no other methods have been used to generate more prefixes.

Future research could look into other methods of enhancements, like:

- querying the current dataset for prefixes discovered through CNAME and SRV resource records;
- querying discovered NTP servers for their reverse DNS PTR resource records to attempt to extract prefixes from hostnames;

- inspecting Certificate Transparency logs for potential prefixes for NTP servers [31]; and
- analyzing more lists of public NTP servers.

10.2 NTP server relationship mapping

One of our research goals was to map ownership of NTP servers through DNS-level and BGP-level knowledge. Due to time constraints, this has not been done, but this is worth exploring in the future.

10.3 Reverse-DNS IPv6 enumeration

Reverse-zone-based IPv6 host enumeration using PTR and NSEC3 resource records appears to be potentially feasible [13]. If this were to be feasible, this could provide a way to not only generate a larger dataset of IPv6 NTP servers, but also a way to validate the results of this research against a more complete dataset.

10.4 Query NTP with other versions

This research has exclusively queried using NTP version 4 packets, and therefore likely misses many NTP servers that do not support this NTP version. Future measurements could be done using older NTP versions, or a combination of new and old versions, in order to attempt to discover more NTP servers.

10.5 Speed up scanning by resolving delegated zones locally

It may be significantly faster to store the zone data for delegated zones on the resolver used by the probing software, in addition to the already stored root-zone, as making requests to the nameservers hosting these zones is no longer necessary. This may reduce round-trip time of the DNS lookups, causing the DNS scanner to be able to progress through domains quicker. This data is available to OpenINTEL, so this is entirely possible to attempt. Moreover, this reduces the load on the nameservers hosting theses zones, which is a significant advantage when doing faster scanning, as the load placed on these servers by the scan is significant.

10.6 Large-scale scanning

Due to time constraints, only the `se.` zone and approximately 2.7% of the `com.` zone have been scanned. Scanning of more zones, and scanning of a larger fraction of the `com.` zone may yield more insights.

As mentioned in the results, the measuring phase is not so slow as to make the scanning of entire large zones infeasible, though it is likely some more optimization work may make the scanning even faster.

10.7 Further exploration of the obtained dataset

As the main intention of this work is to build groundwork for future research, we can suggest a significant amount of further work based on the data that the method developed by this research can obtain.

Doing research on the security of discovered NTP servers, like their vulnerability to becoming a DDoS amplification agent will likely be interesting.

Cao and Veitch have done some qualitative research on NTP servers [12], though this has been done on a relatively small dataset. It would be potentially valuable to apply these methods to larger datasets of NTP servers that we can generate.

A colleague student is currently doing research into the security of public NTP servers, and has only done this for IPv4 servers. His research could be enhanced by doing similar measurements on NTP servers exclusively running IPv6, as discovered by this research.

11 CONCLUSION

This work proposed and demonstrated a method for discovering publicly accessible Network Time Protocol (NTP) servers by leveraging the Domain Name System (DNS) as a discovery mechanism. Unlike traditional host-scanning approaches that are infeasible for the discovery of IPv6-based internet services, DNS-based probing enables scalable, targeted discovery of time services and potentially many other networked services.

We developed a high-performance, multi-stage scanning system that queried millions of domain names and subdomains for NTP-related subdomains. Despite challenges like DNS rate limiting and query volume, the scanning pipeline produced a dataset of thousands of operational NTP servers, including many IPv6-capable hosts that would likely be missed through conventional IP scans.

Our results show that:

- The DNS is a viable and underutilized resource for service discovery, particularly for protocols like NTP that use predictable naming.
- NTP deployment remains heavily skewed toward IPv4, with relatively few domains offering service over IPv6, with 98.5% of the domains only offering IPv4 service.
- We have laid groundwork for future research that goes more in-depth into the analysis of discovered NTP servers and the resilience of the NTP ecosystem.

We used a limited set of domain prefixes in our scanning, and our results show that it is likely worth it to re-evaluate the list of domain prefixes, as that may in future either speed up the scanning, or provide a larger set of NTP servers. Despite this, the current set of domain prefixes already yielded useful results, as can be shown by the small amount of analysis of the scanned data has shown.

While the discovered IPv6 NTP infrastructure remains limited, this method represents a scalable and complementary approach to service enumeration in the modern internet.

It has also been discovered that the large majority of NTP servers operate in stratum 2 or stratum 3, which indicates that the NTP ecosystem is likely deployed in a way very similar to the original intention of the design of the Network Time Protocol. It may, however, indicate a high degree of centralization of the stratum 1 NTP servers, which future research could attempt to quantify.

All in total, we have developed a method to do large-scale DNS probing to discover NTP servers, providing tools for future research into more aspects of the NTP ecosystem, and we have discovered potential problems in the NTP ecosystem with regards to IPv6-readiness and centralization of root “stratum 1” time references.

REFERENCES

[1] 2013. CVE - CVE-2013-5211. https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2013-5211

[2] 2025. DNS Parameters. https://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml

[3] 2025. OpenINTEL: Active DNS Measurement Project. https://openintel.nl/

[4] 2025. A research-oriented top sites ranking hardened against manipulation - Tranco. https://tranco-list.eu/

[5] 2025. Root Zone Database. https://www.iana.org/domains/root/db

[6] 2025. Service Name and Transport Protocol Port Number Registry. https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

[7] 2025. Understanding IP Addressing and CIDR Charts. https://www.ripe.net/about-us/press-centre/understanding-ip-addressing/

[8] Roy Arends, Geoffrey Sisson, David Blacka, and Ben Laurie. 2008. *DNS Security (DNSSEC) Hashed Authenticated Denial of Existence*. Request for Comments RFC 5155. Internet Engineering Task Force. https://doi.org/10.17487/RFC5155

[9] Ask Bjørn Hansen. 2025. pool.ntp.org: The Internet Cluster of NTP Servers. https://www.ntppool.org/

[10] Derek Atkins and Rob Austein. 2004. *Threat Analysis of the Domain Name System (DNS)*. Request for Comments RFC 3833. Internet Engineering Task Force. https://doi.org/10.17487/RFC3833

[11] Jailendrasingh Beeharry and Bhissum Nowbutsing. 2016. Forecasting IPv4 Exhaustion and IPv6 Migration. In *2016 IEEE International Conference on Emerging Technologies and Innovative Business Practices for the Transformation of Societies (EmergiTech)*. 336–340. https://doi.org/10.1109/EmergiTech.2016.7737362

[12] Yi Cao and Darryl Veitch. 2019. Where on Earth Are the Best-50 Time Servers?. In *Passive and Active Measurement*, David Choffnes and Marinho Barcellos (Eds.). Springer International Publishing, Cham, 101–115. https://doi.org/10.1007/978-3-030-15986-3_7

[13] Clinton Carpene. 2016. *An Investigation into Off-Link IPv6 Host Enumeration Search Methods*. Ph. D. Dissertation. https://ro.ecu.edu.au/theses/1772

[14] Internet Systems Consortium. 2025. BIND 9. https://www.isc.org/bind/

[15] 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 Conference on Internet Measurement Conference*. ACM, Vancouver BC Canada, 435–448. https://doi.org/10.1145/2663716.2663717

[16] Greg Goth. 2012. The End of IPv4 is Nearly Here — Really. *IEEE Internet Computing* 16, 2 (March 2012), 7–11. https://doi.org/10.1109/MIC.2012.37

[17] Ask Bjørn Hansen. 2025. GeoDNS servers. https://github.com/abh/geodns

[18] Bob Hinden and Steve E. Deering. 1998. *Internet Protocol, Version 6 (IPv6) Specification*. Request for Comments RFC 2460. Internet Engineering Task Force. https://doi.org/10.17487/RFC2460

[19] Paul E. Hoffman. 2023. *DNS Security Extensions (DNSSEC)*. Request for Comments RFC 9364. Internet Engineering Task Force. https://doi.org/10.17487/RFC9364

[20] Edward P. Lewis and Alfred Hoenes. 2010. *DNS Zone Transfer Protocol (AXFR)*. Request for Comments RFC 5936. Internet Engineering Task Force. https://doi.org/10.17487/RFC5936

[21] Michiel. 2024. Manipulating Time Through (S)NTP. https://tweedegolf.nl/en/blog/142/manipulating-time-through-sntp

[22] D. Mills, J. Martin, J. Burbank, and W. Kasch. 2010. *Network Time Protocol Version 4: Protocol and Algorithms Specification*. Technical Report RFC5905. RFC Editor. RFC5905 pages. https://doi.org/10.17487/rfc5905

[23] Nelson Minar. 1999. A Survey of the NTP Network. https://api.semanticscholar.org/CorpusID:17133789

[24] P.V. Mockapetris. 1987. *Domain names - Concepts and Facilities*. Technical Report RFC1034. RFC Editor. RFC1034 pages. https://doi.org/10.17487/rfc1034

[25] Giovane C. M. Moura, Marco Davids, Caspar Schutijser, Cristian Hesselman, John Heidemann, and Georgios Smaragdakis. 2024. Deep Dive into NTP Pool's Popularity and Mapping. *Proc. ACM Meas. Anal. Comput. Syst.* 8, 1 (Feb. 2024), 15:1–15:30. https://doi.org/10.1145/3639041

[26] David M'Raihi, Johan Rydell, Mingliang Pei, and Salah Machani. 2011. *TOTP: Time-Based One-Time Password Algorithm*. Request for Comments RFC 6238. Internet Engineering Task Force. https://doi.org/10.17487/RFC6238

[27] Victor Le Pochat, Tom Van Goethem, Samaneh Tajalizadehkhoob, Maciej Korczyński, and Wouter Joosen. 2019. Tranco: A Research-Oriented Top Sites Ranking Hardened Against Manipulation. In *Proceedings 2019 Network and Distributed System Security Symposium*. https://doi.org/10.14722/ndss.2019.23386 arXiv:1806.01156 [cs]

[28] John Postel and Ken Harrenstien. 1983. *Time Protocol*. Request for Comments RFC 868. Internet Engineering Task Force. https://doi.org/10.17487/RFC0868

[29] Philipp Richter, Oliver Gasser, and Arthur Berger. 2022. Illuminating Large-scale IPv6 Scanning in the Internet. In *Proceedings of the 22nd ACM Internet Measurement Conference (IMC '22)*. Association for Computing Machinery, New York, NY, USA, 410–418. https://doi.org/10.1145/3517745.3561452

[30] 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

[31] Quirin Scheitle, Oliver Gasser, Theodor Nolte, Johanna Amann, Lexi Brent, Georg Carle, Ralph Holz, Thomas C. Schmidt, and Matthias Wählisch. 2018. The Rise of Certificate Transparency and Its Implications on the Internet Ecosystem. In *Proceedings of the Internet Measurement Conference 2018 (IMC '18)*. Association for Computing Machinery, New York, NY, USA, 343–349. https://doi.org/10.1145/3278532.3278562

[32] SIDN. 2020. Security flaws in Network Time Protocol make other (security) protocols vulnerable. https://www.sidn.nl/en/news-and-blogs/security-flaws-in-network-time-protocol-make-other-security-protocols-vulnerable

[33] Marcin Skwarek, Maciej Korczynski, Wojciech Mazurczyk, and Andrzej Duda. 2019. Characterizing Vulnerability of DNS AXFR Transfers with Global-Scale Scanning. In *2019 IEEE Security and Privacy Workshops (SPW)*. IEEE, San Francisco, CA, USA, 193–198. https://doi.org/10.1109/SPW.2019.00044

[34] Roland van Rijswijk-Deij, Mattijs Jonker, Anna Sperotto, and Aiko Pras. 2015. The Internet of Names: A DNS Big Dataset. *SIGCOMM Comput. Commun. Rev.* 45, 4 (Aug. 2015), 91–92. https://doi.org/10.1145/2829988.2789996

A USAGE OF AI STATEMENT

During the preparation of this work, I used ChatGPT 4o to give hints on the state of the thesis and help with debugging of code written for the collection of data. After using this tool, I thoroughly reviewed and edited the content as needed, so I (Rolf van Kleef) take full responsibility for the final outcome.

B LISTINGS

Listing 1. IPv6 lookaround

```
def find_max_offset(base_ip, port):
    offset = 1
    while offset < (1 << 64):
        if not send_ntp_request(base_ip ^ offset, port):
            return offset
        offset <<= 1
    return offset

def binary_search_range(base_ip, low, high, port):
    while low < high:
        mid = (low + high) // 2
        if send_ntp_request(base_ip ^ mid, port):
            low = mid + 1
        else:
            high = mid
    return high - 1

def estimate_subnet_size(ipv6_addr, port):
    if not send_ntp_request(ipv6_addr, port):
        return 0
    first_fail = find_max_offset(base_int, port)
    return 1 + binary_search_range(base_int, 0, first_fail, port)

servers = get_ipv6_ntp_servers()

for server in servers:
  print("{}, {}, {}",
    server.ip, server.port,
    estimate_subnet_size(server.ip, server.port))
```

Listing 2. DNS Scan

```
pub async fn probe_name(domain: Name, prefix: Option<String>) {
  let mut found_aaaa = false;
  let mut found_cname = false;
  found_aaaa, found_cname = probe_a(domain, prefix).await;
  if !found_aaaa {
    found_cname |= probe_aaaa(domain, prefix).await;
  }
  if !found_cname {
    probe_cname(domain, prefix).await;
  }
  probe_srv(domain, prefix).await;
}
```

Listing 3. DNS CNAME and SRV recursion

```
async fn handle_record(record: Record, domain: Name, prefix: String) {
  match record {
    Record::A(a) => store_record(a.ip, domain, prefix, port: 123),
    Record::AAAA(aaaa) => store_record(aaaa.ip, domain, prefix, port: 123),
    Record::CNAME(cname) => {
      let ip = probe_name(cname.target).await;
      store_record(ip, domain, prefix, port: 123);
    },
```

```
      Record::SRV(srv) => {
        let ip = probe_name(srv.target).await;
        store_record(ip, domain, prefix, srv.port);
      },
    }
}
```