Internet Engineering Task Force
Internet-Draft
Intended status: Best Current Practice
Expires: May 4, 2017

D. Reilly, Ed.
Spectracom Corporation
H. Stenn
Network Time Foundation
D. Sibold
PTB

October 31, 2016

Network Time Protocol Best Current Practices
draft-ietf-ntp-bcp-02

Abstract

NTP Version 4 (NTPv4) has been widely used since its publication as RFC 5905 [RFC5905]. This documentation is a collection of Best Practices from across the NTP community.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

NTP Version 4 (NTPv4) has been widely used since its publication as RFC 5905 [RFC5905]. This documentation is a collection of Best Practices from across the NTP community.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Keeping NTP up to date

Many network security mechanisms rely on time as part of their operation. If an attacker can spoof the time, they may be able to bypass or neutralize other security elements. For example, incorrect time can disrupt the ability to reconcile logfile entries on the affected system with events on other systems. The best way to protect computers and networks against undefined behavior and security threats related to time is to keep their NTP implementations current.

There are always new ideas about security on the Internet, and an application which is secure today could be insecure tomorrow once an unknown bug (or a known behavior) is exploited in the right way. Even our definition of what is secure has evolved over the years, so code which was considered secure when it was written can be considered insecure after some time. By keeping NTP implementations current, network operators can make sure that older behaviors are not exploited.

Thousands of individual bugs have been found and fixed in the NTP Project’s ntpd since the first NTPv4 release in 1997. Each version release contains at least a few bug fixes. The best way to stay in front of these issues is to keep your NTP implementation current.

There are multiple versions of the NTP protocol in use, and multiple implementations in use, on many different platforms. It is recommended that NTP users actively monitor wherever they get their software to find out if their versions are vulnerable to any known attacks, and deploy updates containing security fixes as soon as practical.

The reference implementation of NTP Version 4 from Network Time Foundation (NTF) continues to be actively maintained and developed by NTF’s NTP Project, with help from volunteers and NTF’s supporters.

This NTP software can be downloaded from ntp.org [1] and also from NTF's github page [2].

3. General Network Security Best Practices

3.1. BCP 38

Many network attacks rely on modifying the IP source address of a packet to point to a different IP address than the computer which originated it. This modification/abuse vector has been known for quite some time, and BCP 38 [RFC2827] was approved in 2000 to address this. BCP 38 [RFC2827] calls for filtering outgoing and incoming traffic to make sure that the source and destination IP addresses are consistent with the expected flow of traffic on each network interface. It is recommended that all networks (and ISP's of any size) implement this. If a machine on a network is sending out packets claiming to be from an address that is not on that network, this could be the first indication that there is a machine that has been compromised, and is being used abusively. If packets are arriving on an external interface with a source address that is normally only seen on an internal network, that's a strong indication that an attacker is trying to inject spoofed packets into the network. More information is available at the BCP38 Info page [3] .

4. NTP Configuration Best Practices

These Best Practices, while based on the ntpd reference implementation developed and maintained by Network Time Foundation, may be applicable to other implementations as well.

4.1. Use enough time sources

ntpd takes the available sources of time and submits their timing data to intersection and clustering algorithms, looking for the best idea of the correct time.

- If there is only 1 source of time, the answer is obvious. It might not be a good source of time, but it's the only one.

- If there are 2 sources of time and they agree well enough, that's good. But if they don't, then ntpd has no way to know which source to believe.

- If there are 3 sources of time, you can tolerate one of those sources becoming unreachable or unusable. But at that point, we are back down to 2 sources.

- 4 sources of time is better. If one of these sources develops a problem there are still 3 others.

But even with 4 or more sources of time, systemic issues can happen. During the leap second of June of 2015, several operators implemented leap smearing while others did not, and some NTP clients follwed 2 servers that offered UTC time (with leap second announcements) and 2 servers that offered leap smeared time (with no leap second announcements). See Section 4.6.1 for more information.

Starting with ntp-4.2.6, the ’pool’ directive will spin up "enough" associations to provide robust time service, and will disconnect poor servers and add in new servers as-needed. The default values built in to ntpd should be correct. If you have good reasons, you may use the ’minclock’ and ’maxclock’ options of the ’tos’ command to override the default values of how many servers are discovered and used through the ’pool’ directive.

Properly monitor your NTP instances. If your time sources do not generally agree, find out why and either correct the problems or stop using defective servers. See Section 4.4 for more information.

4.2. Use a diversity of Reference Clocks

When using servers with attached hardware reference clocks, it is recommended that several different types of reference clocks be used. Having a diversity of sources means that any one issue is less likely to cause a service interruption.

Are all clocks on a network from the same vendor? They might have the same bugs. Are they using the same base chipset, regardless of whether or not the finished products are from different vendors? Are they all running the same version of firmware? Chipset and firmware bugs can happen, but are often more difficult to diagnose than a standard software bug.

A systemic problem with time from any satellite navigation service is possible and has happened. Sunspot activity can render satellite or radio-based time source unusable. If the time on your network needs to be correct close to 100% of the time, then even if you are using a satellite-based time source you must plan for those rare instances when the time source is unavailable or wrong.

4.3. Mode 6 and 7

NTP Mode 6 (ntpq) and Mode 7 (ntpdc) packets are designed to permit monitoring and optional authenticated control of ntpd and its configuration. Used properly, these facilities provide vital debugging and performance information, and control facilities. Used improperly, these facilities can be an abuse vector.

Mode 7 queries have been disabled by default in ntpd since 4.2.7p230, released on 2011/11/01. Do not enable Mode 7 unless there is a compelling reason to do so.

The ability to use Mode 6 beyond its basic monitoring capabilities is limited by default to authenticated sessions that provide and use a 'controlkey'. Similarly, if Mode 7 has been explicitly enabled its use for more than basic monitoring is limited by default to authenticated sessions that provide and use a 'requestkey'.

Older versions of the reference implementation of NTP likely do not have the protections listed above and could be abused to participate in high-bandwidth DDoS attacks, if the above restrictions are not applied. Starting with ntp-4.2.7p26, released in April of 2010, ntpd requires the use of a nonce before replying with potentially large response packets.

As mentioned above, there are two general ways to use Mode 6 and Mode 7 requests. One way is to query ntpd for information, and this mode can be disabled with:

```
restrict ... noquery
```

The second way to use Mode 6 and Mode 7 requests is to modify ntpd's behavior. Modification of ntpd ordinarily requires an authenticated session. By default, if no authentication keys have been specified no modifications can be made. For additional protection, the ability to perform these modifications can be controlled with:

```
restrict ... nomodify
```

Adminitstrators can prevent their NTP servers from responding to these directive in the general case by adding the following to their ntp.conf file:

```
restrict default -4 nomodify notrap nopeer noquery

restrict default -6 nomodify notrap nopeer noquery

restrict source nomodify notrap noquery
# nopeer is OK if you don't use the 'pool' directive
```

4.4. Monitoring

The reference implementation of NTP allows remote monitoring. Access to this service is controlled by the restrict statement in ntpd's configuration file (ntp.conf). The syntax is:

```
restrict address mask address_mask nomodify
```

Monitor ntpd instances so machines that are "out of sync" can be quickly identified. Monitor system logs for messages from ntpd so abuse attempts can be quickly identified.

If a system starts getting unexpected time replies from its time servers, that is a likely indication that an abuser is forging your server's IP in time requests to your time server in an attempt to convince your time servers to stop serving time to your system.

If a system is a broadcast client and its syslog shows that it is receiving "early" time messages from its server, that is an indication that somebody might be forging packets from a broadcast server. Broadcast time should only be used in trusted networks.

If a server's syslog shows messages that indicates it is receiving timestamps that are earlier than the current system time, then either the system clock is unusually fast or somebody is trying to launch a replay attack against that server.

If a system is using broadcast mode and is running ntp-4.2.8p6 or later, use the 4th field of the ntp.keys file to specify the IPs of machines that are allowed to serve time to the group.

4.5. Using Pool Servers

It only takes a small amount of bandwidth and system resources to synchronize one NTP client, but NTP servers that can service tens of thousands of clients take more resources to run. Users who want to synchronize their computers should only synchronize to servers that they have permission to use.

The NTP pool project is a collection of volunteers who have donated their computing and bandwidth resources to provide time on the Internet for free. The time is generally of good quality, but comes with no guarantee whatsoever. If you are interested in using the pool, please review their instructions at http://www.pool.ntp.org/en/use.html .

If you want to synchronize many computers using the pool, consider running your own NTP servers, synchronizing them to the pool, and synchronizing your clients to your in-house NTP servers. This reduces the load on the pool.

If you would like to contribute a server with a static IP address and a permanent Internet conenction to the pool, please consult the instructions at http://www.pool.ntp.org/en/join.html .

4.6. Leap Second Handling

UTC is kept in agreement with the astronomical time UT1 [6] to within +/- 0.9 seconds by the insertion or deletion of a leap second. UTC is an atomic time scale whereas UT1 is based on the rotational rate of the earth. Leap seconds are not introduced at a fixed rate. They are announced by the IERS (International Earth rotation and Reference systems Service) in its Bulletin C [7] when necessary to keep UTC and UT1 aligned.

NTP time is based on the UTC timescale, and the protocol has the capability to broadcast leap second information. Some GNSS systems (like GPS) or radio transmitters (like DCF77) broadcast leap second information, so if you have a Stratum-1 server synced to GNSS or you are synced to a lower stratum server that is ultimately synced to GNSS, you will get advance notification of impending leap seconds automatically.

Since the length of the UT1 day is generally slowly increasing [8], all leap seconds that have been introduced since the practice started in 1972 have been "positive" leap seconds, where a second is added to UTC. NTP also supports a "negative" leap second, where a second is removed from UTC, in the event that the IERS announces one.

While earlier versions of NTP contained some ambiguity regarding when a leap second that is broadcast by a server is applied by a client, RFC 5905 is clear that leap seconds are only applied on the last day of a month. However, because some older clients might apply it at the end of the current day, it is recommended that NTP servers wait until the last day of the month before broadcasting leap seconds. Doing this will prevent older clients from applying a leap second at the wrong time. Note well that in NTPv4 the maximum allowed poll interval is 17, or about 1.5 days’ time. In this situation, it’s possible that a client will miss the leap second announcement.

The IETF maintains a leap second list [9]. The use of a leap second list requires ntpd 4.2.6 or later. After fetching the leap seconds file onto the server, add this line to ntpd.conf to apply the file:

```
leapfile "/path/to your/leap-file"
```

You may need to restart ntpd to apply this change.

The leap second list file is also available from other sources:

NIST: ftp://time.nist.gov/pub/leap-seconds.list

US Navy (maintains GPS Time): ftp://tycho.usno.navy.mil/pub/ntp/leap-seconds.list

IERS (announces leap seconds):
https://hpiers.obspm.fr/iers/bul/bulc/ntp/leap-seconds.list

ntpd servers with a manually configured leap second file will ignore leap second information broadcast from upstream NTP servers until the leap second file expires.

If no valid leap second file is available then a leap second notification from an attached reference clock is always accepted by ntpd.

If no valid leap second file is available, a leap second notification may be accepted from upstream NTP servers. As of ntpd 4.2.6, a majority of servers must provide the notification before it is accepted. Before 4.2.6, a leap second notification would be accepted if only a single upstream server of a group of configured servers provided a leap second notification. While this was useful behavior in the past, when information about pending leap seconds was less available. Now, we have the combination of 1) easy availablilty of the leap second file and software to use it, and 2) a greater awareness of the potential for hostile behavior. In this new light, we have shifted from "believe a single warning" to "follow the best authoritative source". The best authoritative source is the leap seconds list file, The next best source would be an attached refclock that can provide notification of leap second adjustments, and the next best source would be the majority consensus from upstream servers. There is still a risk of misbehavior if a "master" NTP server gets an invalid leap second warning, e.g. due to an incorrect leap second list file, or a faulty GPS receiver.

4.6.1. Leap Smearing

Some NTP installations may make use of a technique called "Leap Smearing" to propagate a leap second correction. With this method, instead of introducing an extra second (or eliminating a second), NTP time will be slewed in small increments over a comparably large window of time (called the smear interval) around the leap second event. The smear interval should be large enough to make the rate that the time is slewed small, so that clients will follow the smeared time without objecting. A smear interval of 86400 seconds, which is 1 day, has been used sucessfully. During the adjustment window, all the NTP clients' times could be offset from UTC by as much as a full second, depending on the implementation. But at least all clients will agree on what time they think it is!

The purpose of Leap Smearing is to enable systems that don't deal with the leap second event properly to function smoothly, at the expense of fidelity to UTC during the smear window. During a standard leap second event, that minute will have 61 (or possibly 59) seconds in it, and some applications (and even some OSes) are known to have problems with that.

Leap Smearing was introduced in ntpd versions 4.2.8.p3 and 4.3.47. Support is not availabled by default and has to be specifically added at compile time. In addition, no leap smearing will occur unless a leap smear interval is specified in ntpd.conf . For more information, refer to http://bk1.ntp.org/ntp-stable/README.leapsmear?PAGE=anno .

Clients that are connected to leap smearing servers must not apply the "standard" NTP leap second handling. So if they are using ntpd, these clients must not have a leap second file loaded, and the smearing servers must not advertise that a leap second is pending.

Leap Smearing must not be used for public-facing NTP servers, as they will disagree with non-smearing servers (as well as UTC) during the leap smear interval. However, be aware that some public-facing servers might be configured this way anyway in spite of this guidance.

System Administrators are advised to be aware of impending leap seconds and how the servers (inside and outside their organization) they are using deal with them. Individual clients must not be configured to use a mixture of smeared and non-smeared servers. If a client uses smeared servers, the servers it uses must all have the same leap smear configuration.

4.7. Configuring ntpd

See https://support.ntp.org/bin/view/Support/ConfiguringNTP for additional information on configuring ntpd.

5. NTP Security Mechanisms

In the standard configuration NTP packets are exchanged unprotected between client and server. An adversary that is able to become a Man-In-The-Middle is therefore able to drop, replay or modify the content of the NTP packet, which leads to degradation of the time synchronization or the transmission of false time information. A profound threat analysis for time synchronization protocols are given in RFC 7384 [RFC7384]. NTP provides two internal security mechanisms to protect authenticity and integrity of the NTP packets. Both measures protect the NTP packet by means of a Message Authentication Code (MAC). Neither of them encrypts the NTP's payload, because it is not considered to be confidential.

5.1. Pre-Shared Key Approach

This approach applies a symmetric key for the calculation of the MAC, which protects authenticity and integrity of the exchanged packets for a association. NTP does not provide a mechanism for the exchange of the keys between the associated nodes. Therefore, for each association, keys have to be exchanged securely by external means. It is recommended that each association be protected by its own unique key. NTP does not provide a mechanism to automatically refresh the applied keys. It is therefore recommended that the participants periodically agree on a fresh key. The calculation of the MAC may always be based on an MD5 hash. If the NTP daemon is built against an OpenSSL library, NTP can also base the calculation of the MAC upon the SHA-1 or any other digest algorithm supported by each side's OpenSSL library.

To use this approach the communication partners have to exchange the key, which consists of a keyid with a value between 1 and 65534, inclusive, and a label which indicates the chosen digest algorithm. Each communication partner adds this information to their key file in the form:

```
keyid label key
```

The key file contains the key in clear text. Therefore it should only be readable by the NTP process. Different keys are added line by line to the key file.

A NTP client establishes a protected association by appending the option "key keyid" to the server statement in the NTP configuration file:

```
server address key keyid
```

Note that the NTP process has to trust the applied key. An NTP process explicitly has to add each key it want to trust to a list of trusted keys by the "trustedkey" statement in the NTP configuration file.

```
trustedkey keyid_1 keyid_2 ... keyid_n
```

5.2. Autokey

Autokey was designed in 2003 to provide a means for clients to authenticate servers. By 2011, security researchers had identified computational areas in the Autokey protocol that, while secure at the time of its original design, were no longer secure.

It is recommended that Autokey not be used. Know that as of the fall of 2011, a common(?) laptop computer could crack the security cookie used in the Autokey protocol in 30 minutes' time. If you want to use Autokey, know that your session keys should be set to expire in under 30 minutes' time.

If you have reason to believe your autokey-protected associations will be attacked, you should read https://lists.ntp.org/pipermail/ntpwg/2011-August/001714.html and decide what resources your attackers might be using, and adjust the session key expiration time accordingly.

5.3. Network Time Security

Work has begun on an enhanced replacement for Autokey, which is called Network Time Security (NTS) [NTS]. NTS was first published as an Internet-Draft in the summer of 2013. As of October 2016, this effort was at draft #15, and about to begin 'final call'. The first unicast implementation of NTS was started in the summer of 2015 and is expected to be released in early 2017.

6. NTP Security Best Practices

6.1. Minimizing Information Leakage

The base NTP packet leaks important information (including reference ID and reference time) that can be used in attacks [NDSS16], [CVE-2015-8138], [CVE-2016-1548]. A remote attacker can learn this information by sending mode 3 queries to a target system and inspecting the fields in the mode 4 response packet. NTP control queries also leak important information (including reference ID, expected origin timestamp, etc.) that can be used in attacks [CVE-2015-8139]. A remote attacker can learn this information by sending control queries to a target system and inspecting the response.

As such, access control should be used to prevent the exposure of this information to inappropriate third parties.

Hosts should only respond to NTP control queries from authorized parties. One way to do this is to only allow control queries from authorized IP addresses.

A host that is not supposed to act as an NTP server that provides timing information to other hosts should additionally drop incoming mode 3 timing queries.

A "leaf client" is a host that is using NTP solely for the purpose of adjusting its own time. A leaf client should not be a time server to other hosts. That is, a leaf client sends mode 3 queries to its servers and receives mode 4 responses from these servers containing timing information. To minimize information leakage, leaf clients should drop all incoming NTP packets except for packets coming from trusted monitoring systems and mode 4 response packets that come from its configured time sources.

6.2. Avoiding Daemon Restart Attacks

[RFC5905] says NTP clients should not accept time shifts greater than the panic threshold. Specifically, RFC5905 says "PANIC means the offset is greater than the panic threshold PANICT (1000 s) and should cause the program to exit with a diagnostic message to the system log."

However, this behavior is designed to be used only in cold-start situations. If it is used in more general situations it can be exploited by attackers [NDSS16] when ntpd is restarted with a disabled panic gate check.

If your operating system has init scripts, these scripts should not disable panic gate checking on restarts.

A growing number of operating systems use process supervisors such as systemd to automatically restart any daemons that quit. This behavior is the default in CoreOS and Arch Linux. It is likely to become the default behavior in other Linux-based systems as they migrate legacy init scripts to systemd. These scripts should not disable panic gate checking on restarts.

If, against long-standing recommendations, a system disables panic gate checking on all restarts, an attacker can send the target an offset that exceeds the panic threshold, causing the client to quit. Then, when the client restarts, it ignores the panic threshold and accepts the attacker’s large offset.

Hosts running with the above two conditions should be aware that the panic threshold does not protect them from attacks. A natural solution is not to run hosts with these conditions.

As an alternative, the following steps could be taken to mitigate the risk of attack.

- Monitor NTP system log to detect when the NTP daemon has quit due to a panic event, as this could be a sign of an attack.
- Request manual intervention when a timestep larger than the panic threshold is detected.
- Prevent the NTP daemon from taking time steps that set the clock to a time earlier than the compile date of the NTP daemon.
- Modify the NTP daemon so that it "hangs" (ie does not quit, but just waits for a better timing samples but does not modify the local clock) when it receives a large offset.

6.3. Detection of Attacks Through Monitoring

Users should monitor their NTP instances to detect attacks. Many known attacks on NTP have particular signatures. Common attack signatures include:

1. "Bogus packets" - A packet whose origin timestamp does not match the value that expected by the client.
2. "Zero origin packet" - A packet with a origin timestamp set to zero [CVE-2015-8138].
3. A packet with an invalid cryptographic MAC [CCR16].

The observation of many such packets could indicate that the client is under attack.

Also, Kiss-o'-Death (KoD) packets can be used in denial of service attacks. Thus, the observation of even just one KoD packet with a high poll value (e.g. poll>10) could be sign that the client is under attack.

6.4. Broadcast Mode Should Only Be Used On Trusted Networks

Per [RFC5905], NTP's broadcast mode is authenticated using symmetric key cryptography. The broadcast server and all of its broadcast clients share a symmetric cryptographic key, and this key is used by the broadcast server to build and by the broadcast clients to authenticate the Message Authentication Code (MAC) that protects NTP broadcast packets.

Put another way, all broadcast clients that listen to broadcast servers know and share the same cryptographic key. This mean that any client can use this key to send valid broadcast messages that look like they come from the broadcast server. Thus, a rogue with knowledge of this key cab attack broadcast clients.

For this reason, all NTP broadcast servers and clients need to trust each other. Broadcast mode should only be run from within a trusted network.

Starting with ntp-4.2.8p7 the ntp.keys file accepts an optional 4th column, a comma-separated list of IPs that are allowed to serve time. Use this feature.

Updated NTP broadcast clients are protected against and detect these attacks by reporting unexpected or inconsistent broadcast packets, and by ignoring broadcast packets that arrive "too early". Monitor your NTP log files.

6.5. Symmetric Mode Should Only Be Used With Trusted Peers

In symmetric mode, two peers Alice and Bob can both push and pull synchronization to and from each other using either ephemeral symmetric passive (mode 2) or persistent symmetric active (NTP mode 1) packets. The persistent association is preconfigured and initiated at the active peer but not preconfigured at the passive peer (Bob). Upon receipt of a mode 1 NTP packet from Alice, Bob mobilizes a new ephemeral association if he does not have one already. This is a security risk for Bob because an arbitrary attacker can attempt to change Bob’s time by asking Bob to become its symmetric passive peer.

For this reason, a host (Bob) should only allow symmetric passive associations to be established with trusted peers. Specifically, Bob should require each of its symmetric passive association to be cryptographically authenticated. Each symmetric passive association should be authenticated under a different cryptographic key.

The use of a different cryptographic key for each peer association prevents a Sybil attack, where a single malicious peer uses the same cryptographic key to set up multiple symmetric associations a target, and thus bias the results of the target’s Byzantine fault tolerant peer selection algorithms.

7. NTP in Embedded Devices

Readers of this BCP likely already understand how important accurate time is for network computing. And as computing becomes more ubiquitous, there will be many "Internet of Things" devices that require accurate time. These embedded devices might not have a traditional user interface, but if they connect to the Internet they will be subject to the same security threats as traditional deployments.

7.1. Updating Embedded Devices

Vendors of embedded devices have a special responsibility to pay attention to the current state of NTP bugs and security issues and fix them quickly, because their customers don't have the ability to update their NTP implementation on their own. Those devices might have a single firmware upgrade, provided by the manufacturer, that updates all capabilities at once. This means that the vendor assumes the responsibility of making sure their devices have the latest NTP updates applied.

This should also include the ability to update any NTP server addresses on these devices.

There is a catalog of NTP server abuse incidents, some of which involve embedded devices, on the Wikipedia page for NTP Server Misuse and Abuse [12].

7.2. KISS Packets

The "Kiss-o'-Death" (KoD) packet is a rate limiting mechanism where a server can tell a misbehaving client to "back off" its query rate. It is important for all NTP devices to respect these packets and back off when asked to do so by a server. It is even more important for an embedded device, which may not have exposed a control interface for NTP.

The KoD mechanism relies on clients behaving properly in order to be effective. Some clients ignore the KoD packet entirely, and other poorly-implemented clients erroneously and destructively increase their poll rate and create a low-level denial of service attack. Server administrators should be prepared for this and take measures outside of the NTP protocol to drop packets from misbehaving clients.

7.3. Server configuration

Vendors of embedded devices that need time synchronization should also carefully consider where they get their time from. There are several public-facing NTP servers available, but they might not be prepared to service requests from thousands of new devices on the Internet.

Vendors are encouraged to invest resources into providing their own time servers for their devices to connect to.

7.3.1. Get a vendor subdomain for pool.ntp.org

The NTP Pool Project offers a program where vendors can obtain their own subdomain that is part of the NTP Pool. This offers vendors the ability to safely make use of the time distributed by the Pool for their devices. Vendors are encouraged to support the pool if they participate. For more information, visit http://www.pool.ntp.org/en/vendors.html .

8. NTP over Anycast

Anycast is described in BCP 126 [RFC4786], with additional information at RFC 7094 [RFC7094]. With anycast, single IP address is assigned to multiple interfaces, and routers direct packets to the closest active interface.

Anycast is often used for Internet services at known IP addresses, such as DNS. Anycast could be used in large organizations to simplify configuration of a large number of NTP clients, but note well this simplification comes at the cost of degraded NTP behavior and performance. Each client can be configured with the same NTP server IP address, and a pool of anycast servers can be deployed to service those requests. New servers can be added to or taken from the pool, and other than a possible brief loss of service immediately after server is taken down (and before packets are directed to a new server), these additions are transparent to the clients.

If clients are connected to an NTP server via anycast, the client does not know which particular server they are connected to. As anycast servers are allowed to arbitrarily enter and leave the network or as the routing behavior changes, the server a particular client is connected to could change. This can cause a shift in the delay and symmetry between the client and the server.

NOTE WELL: Using anycast for NTP is likely to be a bad idea, because it means there is likely to be an apparent single time server source for the client population. A key element of a robust NTP deployment is multiple sources of time. With multiple time servers a client can analyze the various time sources, selecting good ones, and disregarding poor ones. In an Anycast scenario, this analysis is likely impossible.

If clients are connected to an NTP server via anycast, the client does not know which particular server they are connected to. As anycast servers are allowed to arbitrarily enter and leave the network, the server any given client is connected to could change. It is recommended that anycast be deployed in environments where these small shifts can be tolerated.

Configuration of an anycast interface is independent of NTP. Clients will always connect to the closest anycast server, even if that server is having NTP issues. It is recommended that anycast NTP implementations have an independent method of monitoring the performance of NTP on all servers and clients. If a server is not performing to specification, it should remove itself from the Anycast network. It is also recommended that each Anycast NTP server have at least one Unicast interface so its performance can be checked independently of the anycast routing scheme.

One useful application in large networks is to use a hybrid unicast/ anycast approach. Stratum 1 NTP servers can be deployed with unicast interfaces at several sites. Each site could have several Stratum 2 servers with two ethernet interfaces. One interface has a unique unicast IP address. The second has an anycast IP interface (with a shared IP address per location). The unicast interfaces can be used to obtain time from the Stratum 1 servers globally (and perhaps peer with the other Stratum 2 servers at their site). Clients at each site can be configured to use the shared anycast address for their site, simplifying their configuration. Keeping the anycast routing restricted on a per-site basis will minimize the disruption at the client if its closest anycast server changes. Each Stratum 2 server can be uniquely identified on their unicast interface, to make monitoring easier.

9. Acknowledgements

The authors wish to acknowledge the contributions of Sue Graves, Samuel Weiler, Lisa Perdue, Karen O’Donoghue, David Malone, Sharon Goldberg, Martin Burnicki, and Miroslav Lichvar.

10. IANA Considerations

This memo includes no request to IANA.

11. Security Considerations

Time is a fundamental component of security on the internet. Credentials and certificates can expire. Logins and other forms of access can be revoked after a period of time, or at a scheduled time. And some applications might assume that system time cannot be changed and is always monotonic, and vulnerabilites could be exposed if a time in the past is forced into a system. Therefore, any system adminstrator concerned with security should be concerned with how the current time gets into their system.

[NTS] is an Internet-Draft of a collection of methods to secure time transfer over networks. [NTSFORNTP] is an Internet-Draft that applies the methods in [NTS] specifically to NTP. At the time of this writing, these are still drafts. Readers are encouraged to check the status of these drafts, and make use of the methods they describe.

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2827] Ferguson, P. and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", BCP 38, RFC 2827, DOI 10.17487/RFC2827, May 2000, <http://www.rfc-editor.org/info/rfc2827>.

[RFC4786] Abley, J. and K. Lindqvist, "Operation of Anycast Services", BCP 126, RFC 4786, DOI 10.17487/RFC4786, December 2006, <http://www.rfc-editor.org/info/rfc4786>.

[RFC5905] Mills, D., Martin, J., Ed., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, DOI 10.17487/RFC5905, June 2010, <http://www.rfc-editor.org/info/rfc5905>.

[RFC7094] McPherson, D., Oran, D., Thaler, D., and E. Osterweil, "Architectural Considerations of IP Anycast", RFC 7094, DOI 10.17487/RFC7094, January 2014, <http://www.rfc-editor.org/info/rfc7094>.

[RFC7384] Mizrahi, T., "Security Requirements of Time Protocols in Packet Switched Networks", RFC 7384, DOI 10.17487/RFC7384, October 2014, <http://www.rfc-editor.org/info/rfc7384>.

12.2. Informative References

[CCR16] Malhotra, A. and S. Goldberg, "Attacking NTP’s Authenticated Broadcast Mode", SIGCOMM Computer Communications Review (CCR) , 2016.

[CVE-2015-8138] Van Gundy, M. and J. Gardner, "NETWORK TIME PROTOCOL ORIGIN TIMESTAMP CHECK IMPERSONATION VULNERABILITY", 2016, <http://www.talosintel.com/reports/TALOS-2016-0077>.

[CVE-2015-8139] Van Gundy, M., "NETWORK TIME PROTOCOL NTPQ AND NTPDC ORIGIN TIMESTAMP DISCLOSURE VULNERABILITY", 2016, <http://www.talosintel.com/reports/TALOS-2016-0078>.

[CVE-2016-1548] Gardner, J. and M. Lichvar, "Xleave Pivot: NTP Basic Mode to Interleaved", 2016, <http://blog.talosintel.com/2016/04/vulnerability-spotlight-further-ntpd_27.html>.

[NDSS16] Malhotra, A., Cohen, I., Brakke, E., and S. Goldberg, "Attacking the Network Time Protocol", NDSS’16, San Diego, CA. , 2016, <https://eprint.iacr.org/2015/1020.pdf>.

[NTS] Sibold, D., Roettger, S., and K. Teichel, "Network Time Security", draft-ietf-ntp-network-time-security-15 (work in progress), September 2016.

[NTSFORNTP] Sibold, D., Roettger, S., and K. Teichel, "Using the Network Time Security Specification to Secure the Network Time Protocol", draft-ietf-ntp-using-nts-for-ntp-06 (work in progress), September 2016.

12.3. URIs

[1] http://www.ntp.org/downloads.html

[2] https://github.com/ntp-project/ntp

[3] bcp38.info">http://www.bcp38.info

[6] https://en.wikipedia.org/wiki/Solar_time#Mean_solar_time

[7] https://www.iers.org/IERS/EN/Publications/Bulletins/bulletins.html

[8] https://en.wikipedia.org/wiki/Solar_time#Mean_solar_time

[9] https://www.ietf.org/timezones/data/leap-seconds.list

[12] https://en.wikipedia.org/wiki/NTP_server_misuse_and_abuse

Authors' Addresses

Denis Reilly (editor)
Spectracom Corporation
1565 Jefferson Road, Suite 460
Rochester, NY 14623
US

Email: denis.reilly@spectracom.orolia.com

Harlan Stenn
Network Time Foundation
P.O. Box 918
Talent, OR 97540
US

Email: stenn@nwtime.org

Dieter Sibold
Physikalisch-Technische Bundesanstalt
Bundesallee 100
Braunschweig D-38116
Germany

Phone: +49-(0)531-592-8420
Fax: +49-531-592-698420
Email: dieter.sibold@ptb.de