

Internet Engineering Task Force
Internet-Draft
Intended status: Best Current Practice
Expires: September 10, 2016

D. Reilly
Spectracom Corporation
H. Stenn
Network Time Foundation
D. Sibold
PTB
March 9, 2016

Network Time Protocol Best Current Practices
draft-reilly-ntp-bcp-01

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

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on September 10, 2016.

Copyright Notice

Copyright (c) 2016 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in [Section 4](#).e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	2
1.1.	Requirements Language	3
2.	Keeping NTP up to date	3
3.	General Network Security Best Practices	3
3.1.	BCP 38	4
4.	NTP Configuration Best Practices	4
4.1.	Use enough time sources	4
4.2.	Use a diversity of Reference Clocks	5
4.3.	Mode 6 and 7	5
4.4.	Monitoring	6
4.5.	Security	6
4.5.1.	Pre-Shared Key Approach	7
4.5.2.	Autokey	8
4.5.3.	External Security Means	8
4.6.	Using Pool Servers	8
4.7.	Starting, Cold-Starting, and Re-Starting NTP	9
4.8.	Leap Second Handling	9
4.8.1.	Leap Smearing	9
5.	NTP in Embedded Devices	10
5.1.	Updating Embedded Devices	10
5.2.	KISS Packets	10
5.3.	Server configuration	11
5.3.1.	Get a vendor subdomain for pool.ntp.org	11
6.	NTP Deployment Examples	11
6.1.	Client-Only configuration	11
6.2.	Server-Only Configuration	11
6.3.	Anycast	11
7.	Acknowledgements	12
8.	IANA Considerations	12
9.	Security Considerations	13
10.	References	13
10.1.	Normative References	13
10.2.	URIs	13
	Authors' Addresses	14

[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

No software (not even NTP) is perfect. Bugs can be present in any software. Even if software is thoroughly tested and "all" the bugs are discovered and fixed, users continuously find new ways to use software that their authors did not conceive of, which can uncover more bugs. Thousands of individual bugs have been found and fixed in the NTP Project's reference implementation since the first NTPv4 release in 1997.

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. Many security mechanisms rely on time, either directly or indirectly, 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.

In general, the best way to protect yourself and your networks against these bugs and security threats is to make sure that you keep your NTP implementation up-to-date. There are multiple versions of the NTP protocol in use and multiple implementations and versions of NTP software also 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. The NTP software can be downloaded from [ntp.org](#) [[1](#)] and also from NTF's github page [[2](#)].

3. General Network Security Best Practices

NTP deployments are only as secure as the networks they are running over.

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 your network is sending out packets claiming to be from an address that is not on your network, this could be the first indication that you have a machine that has been cracked, and is being used abusively. If packets are arriving on an external interface with a source address that should only be seen on an internal network, that's a strong indication that an attacker is trying to inject spoofed packets into your network. More information is available at <http://www.bcp38.info> .

4. NTP Configuration Best Practices

NTP can be made more secure by making a few simple changes to the ntp.conf file.

4.1. Use enough time sources

NTP 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 may 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. This gets easier if there are 3 sources of time. But if one of those 3 sources becomes unreachable or unusable, we're back to only having 2 time sources. 4 sources of time is another interesting choice, assuming things go well. If one of these sources develops a problem there are still 3 others. Seems good. Until the leap second we had in June of 2015, where several operators implemented leap smearing while others did not. See [Section 4.8.1](#) 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.

Monitor your ntpd instances. If your times 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

If you are using reference clocks, it is recommended that you use several different types. Having a diversity of sources means that any one issue is less likely to cause a service interruption.

Are all your clocks from the same vendor? 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? 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. A chipset problem can happen. There may be a bug in the vendor's firmware.

4.3. Mode 6 and 7

NTP Mode 6 (ntpq) and Mode 7 (ntpd) 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. Used improperly, these facilities can be an abuse vector.

Mode 7 queries have been disabled by default since 4.2.7p230, released on 2011/11/01. Unless you have a good reason for using ntpdc, do not enable Mode 7.

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

Older versions of the reference implementation of NTP could be abused to participate in high-bandwidth DDos attacks. 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
```

Users can prevent their NTP servers from participating 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. The access to this service is controlled by the restrict statement in NTP's configuration file (ntp.conf). The syntax reads:

```
restrict address mask address_mask nomodify
```

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

If your system starts getting unexpected time replies from its time servers, that can be an indication that the IP address of your server is being forged in requests to that time server, and these abusers are trying to convince your time servers to stop serving time to you.

If your system is a broadcast client and your syslog shows that you are receiving "early" time messages from your server, that is an indication that somebody may be forging packets from a broadcast server.

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

If you are using broadcast mode and have ntp-4.2.8p6 or later, use the 4th field of the ntp.keys file to identify the IPs of machines that are allowed to serve time to the group.

4.5. Security

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 security measures 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.

[4.5.1](#). 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 is 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

4.5.2. 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. Work was begun on an enhanced replacement for Autokey, which was called Network Time Security (NTS) [3]. NTS was published in the summer of 2013. As of February 2016, this effort was at draft #13, 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 the summer of 2016.

We recommend 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 must 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.

4.5.3. External Security Means

TBD

4.6. 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.

Set up or sponsor one or more pool servers.

4.7. Starting, Cold-Starting, and Re-Starting NTP

Only use -g on cold-start. Other things TBD.

Editor's Note: I think I'd like to expand this a bit to cover how to deal with NTP stopping, when to restart it, and under what circumstances to not restart it!

4.8. Leap Second Handling

The UTC timescale is kept in sync with the rotation of the earth through the use of leap seconds. NTP time is based on the UTC timescale, and the protocol has the capability to broadcast leap second information. Some GNSS systems (like GPS) 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.

The International Earth Rotation and Reference Systems Service (IERS) is responsible to announce the introduction of a leap second. It maintains a leap second list at <https://hpiers.obspm.fr/iers/bul/bulc/ntp/leap-seconds.list> for NTP users who are not receiving leap second information through an automatic source. 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 will need to restart to apply the changes.

4.8.1. Leap Smearing

Some NTP installations may instead make use of a technique called "Leap Smearing". 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 around the leap second event. The amount of the slew should be small enough that clients will follow the smeared time without objecting. During the adjustment window, the NTP server's time may be offset from UTC by as much as .5 seconds. This is done to enable software that doesn't deal with minutes that have more or less than 60 seconds to function correctly, at the expense of fidelity to UTC during the smear window.

Leap Smearing was introduced in ntpd versions 4.2.8.p3 and 4.3.47. Support is not configured by default and must be 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>.

Leap Smearing is not recommended for public-facing NTP servers, as they will disagree with non-smearing servers during the leap smear interval. However, some public-facing servers may be configured this way anyway. Users are advised to be aware of impending leap seconds and how the servers (inside and outside their organization) they are using deal with them.

5. NTP in Embedded Devices

Readers of this BCP already understand how important accurate time is for network computing. And as computing becomes more ubiquitous, there will be many small "Internet of Things" devices that require accurate time. These embedded devices may 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.

5.1. Updating Embedded Devices

Vendors of embedded devices have a special responsibility to pay attention to the current state of NTP bugs and security issues, because their customers usually don't have the ability to update their NTP implementation on their own. Those devices may 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 the NTP server address.

(Note: do we find specific historical instances of devices behaving badly and cite them here?)

5.2. 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 might unintentionally increase their poll rate and simulate a 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.

5.3. 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 may 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.

5.3.1. 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> .

6. NTP Deployment Examples

A few examples of interesting NTP Deployments

6.1. Client-Only configuration

TBD

6.2. Server-Only Configuration

TBD

6.3. Anycast

Anycast is described in [BCP 126 \[RFC4786\]](#). (Also see [RFC 7094 \[RFC7094\]](#)). With anycast, a 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 can also be used in large organizations to simplify configuration of a large number of NTP clients. 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 temporary loss of service while a server is taken down, these additions can be 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 may arbitrarily enter and leave the network, the server a particular client is connected to may change. This may cause a small shift in time from the perspective of the client when the server it is connected to changes. 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 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 a server. In the event the 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 may have several Stratum 2 servers with a unicast interface and an anycast 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.

7. Acknowledgements

The author wishes to acknowledge the contributions of Sue Graves, Samuel Weiler, Lisa Perdue, and Karen O'Donoghue.

8. IANA Considerations

This memo includes no request to IANA.

9. Security Considerations

TBD

10. References

10.1. 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>>.

10.2. URIs

- [1] <http://www.ntp.org/downloads.html>
- [2] <https://github.com/ntp-project/ntp>
- [3] <https://tools.ietf.org/html/draft-ietf-ntp-network-time-security-00>

Authors' Addresses

Denis Reilly
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@nwttime.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

