Internet Engineering Task Force
Internet-Draft
Intended status: Best Current Practice
Expires: December 15, 2017

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

June 13, 2017

Network Time Protocol Best Current Practices
draft-ietf-ntp-bcp-05

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 detect and protect computers and networks against undefined behavior and security threats related to time is to keep their NTP implementations current, use an appropriate number of trustworthy time sources, and properly monitor their time infrastructure.

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 may turn out to be insecure after some time. By keeping NTP implementations current, having "enough" trustworthy time sources, and properly monitoring their time infrastructure, network operators can make sure that their time infrastructure is operating correctly and within specification, and is not being attacked or misused.

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 NTP Project's BitKeeper repository [2] or github page [3].

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 ingress and egress filtering. 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 should only be 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 [4] .

4. NTP Configuration Best Practices

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

4.1. Use enough time sources

An NTP implementation (as opposed to an SNTP implementation) takes the available sources of time and submits this timing data to sophisticated intersection, clustering, and combing algorithms to get the best estimate of the correct time. The description of these algorithms is beyond the scope of this document. Interested readers should read RFC 5905 [RFC5905] or the detailed description of NTP in MILLS 2006 [MILLS2006]

o 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 source of time that can be considered. Any issue with the time at the source will be passed on to the client.

- If there are 2 sources of time and they agree well enough, then the best "time" can be calculated easily. But if one source fails, then your solution degrades to the single-source solution outlined above. And if the two sources don't agree, then it's impossible to know which one is correct by simply looking at the time.

- If there are 3 sources of time, there is more data available to converge on a "best" time, and this time is more likely to be accurate. And you can tolerate one of the sources becoming unreachable or unusable. But at that point, you are back down to 2 sources.

- 4 or more sources of time is better. If one of these sources develops a problem there are still at least 3 other time sources.

But even with 4 or more sources of time, systemic problems can happen. During the leap second of June of 2015, several operators implemented leap smearing while others did not, and many NTP end nodes could not determine an accurate time source because 2 of their 4 sources of time gave them consistent UTC/POSIX time, while the other 2 gave them consistent leap-smeared time. 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. If you have good reason, you may use the 'minclock' and 'maxclock' options of the 'tos' command to override the default values of how many servers are discovered through the 'pool' directive.

Monitor your ntpd 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 may 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 is 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 must be correct close to 100% of the time, then even if you are using a satellite-based system, you must plan for those rare instances when the system 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. 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 can be limited to authenticated sessions that provide a 'controlkey'. 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, 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's configuration 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 considering query/configuration traffic by default 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 generally controlled by the "noquery" directive in NTP's configuration file (ntp.conf) via a "restrict" statement. The syntax reads:

```
restrict address mask address_mask noquery
```

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

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

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 group of volunteers who have donated their computing and bandwidth resources to freely distribute time from primary time sources to others on the Internet. 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 are a vendor who wishes to provide time service to your customers or clients, consider joining the pool and providing a "vendor zone" thru the pool project.

If you want to synchronize many computers, consider running your own NTP servers that are synchronized by 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 [7] to within +/- 0.9 seconds by the insertion (or possibly a 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 [8] 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 are synced to an ntp server that is ultimately synced to a source that provides leap second notification you will get advance notification of impending leap seconds automatically.

Since the length of the UT1 day is generally slowly increasing [9], 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, should that ever become necessary (or useful).

While earlier versions of NTP contained some ambiguity regarding when a leap second that is broadcast by a server should be 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 may 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 NTPv4 allows a maximum poll interval of 17, or 131,072 seconds, which is longer than a day.

The IETF maintains a leap second list [10] for NTP users who are not receiving leap second information through an automatic source. The use of leap second files requires ntpd 4.2.6 or later. After fetching the leap seconds file onto the server, add this line to ntpd.conf to apply and use the file:

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

You may need to restart ntpd to apply this change.

Files are 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 ntp-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 a single upstream server of a group of configured servers provided a leap second notification. This would lead to misbehavior if single NTP servers sent an invalid leap second warning, e.g. due to a faulty GPS receiver in one server, but this behavior was once chosen because

in the "early days" there was a greater chance that leap second information would be available from a very limited number of sources.

4.6.1. 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 (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. Periods ranging from 2 to 24 hours have been used sucessfully. During the adjustment window, all the NTP clients' times may be offset from UTC by as much as a full second, depending on the implementation. But at least all clients will generally agree on what time they think it is!

NOTE WELL that using a leap-smear can cause your reported time to be "legally indefensible" and/or be a breach of compliance regulations.

The purpose of Leap Smearing is to enable systems that don't deal with the leap second event properly to function consistently, 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 OS's) are known to have problems with that.

Leap Smearing was introduced in ntpd versions 4.2.8.p3 and 4.3.47, in response to CLIENT requests. Support for leap smearing 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://bk.ntp.org/ntp-stable/README.leapsmear?PAGE=anno [11].

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 never have a leap second file loaded, and the smearing servers must never advertise to clients 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 may 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 never 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

Configuration access to ntpd is controlled by the "modify" status in NTP's configuration file (ntp.conf), which is controlled by a "restrict" statement. The syntax is:

```
restrict address mask address_mask nomodify
```

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 this payload information 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, and an AES-128-CMAC hash is expected to soon be allowed as well. If the NTP daemon is built against an OpenSSL library, NTP can also base the calculation of the MAC upon 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. A key is deemed trusted when its keyid is added to the 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. However, security researchers have identified vulnerabilities in the Autokey protocol, which make the protocol "useless". [13]

Autokey SHOULD NOT BE USED.

5.3. Network Time Security

Work has begun on an enhanced replacement for Autokey, which is called Network Time Security (NTS) [NTS]. NTS was published 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 may 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 may 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 limit 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 authenticated sources via authorized IP addresses.

A host that is not supposed to act as an NTP server that provides timing information to other hosts may additionally log and drop incoming mode 3 timing queries from unexpected sources. Note well that the easiest way to monitor ntpd's status is to send it a mode 3 query. A much better appropach might be to filter mode 3 queries at the edge, or make sure mode 3 queries are allowed from trusted systems or networks.

An "leaf-node host" is host that is using NTP solely for the purpose of adjusting its own system time. Such a host is not expected to provide time to other hosts, and relies exclusively on NTP's basic mode to take time from a set of servers. (That is, the host sends mode 3 queries to its servers and receives mode 4 responses from these servers containing timing information.) To minimize information leakage, leaf-node hosts should drop all incoming NTP packets except mode 4 response packets that come from unknown sources. Note well that proper monitoring of an ntpd instance includes checking the time of that ntpd instance.

6.2. Avoiding Daemon Restart Attacks

RFC 5905 [RFC5905] says NTP clients should not accept time shifts greater than the panic threshold. Specifically, RFC 5905 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 can be exploited by attackers [NDSS16], when the following two conditions hold:

1. The operating system automatically restarts the NTP daemon when it quits. (Modern *NIX operating systems are replacing traditional init systems with process supervisors, such as systemd, which can be configured 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 systems as they migrate legacy init scripts to process supervisors such as systemd.)

2. If, against long-standing recommendation, ntpd is always started with the -g option, it will ignore the panic threshold when it is restarted. The -g option SHOULD only be provided in cold-start situations.

In such cases, if the attacker can send the target an offset that exceeds the panic threshold, the client will 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. The recommended and natural solution is not to run hosts with these conditions. Specifically, only use the -g flag in cold-start situations, or when sufficient oversight and checking is in place to make sure that the use of -g is appropriate.

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

o 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.

o Request manual intervention when a timestep larger than the panic threshold is detected.

o Prevent the NTP daemon from taking time steps that set the clock to a time earlier than the compile date of the NTP daemon.

o Add "minsane" and "minclock" parameters to the ntp.conf file so ntpd waits until "enough" trusted sources of time agree on the correct time.

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 could be sign that the client is under attack. See Section 6.4 for more information.

6.4. 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.

That said, a client must only accept a KoD packet if it has a valid origin timestamp. Once a RATE packet is accepted, the client should increase its poll interval value (thus decreasing its polling rate) up to a reasonable maximum. This maximum can vary by implementation but should not exceed a poll interval value of 13 (2 hours). The mechanism to determine how much to increase the poll interval value is undefined in RFC 5905 [RFC5905]. If the client uses the poll interval value sent by the server in the KoD packet, it must not simply accept any value. Using large interval values may open a vector for a denial-of-service attack that causes the client to stop querying its server [NDSS16].

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.

6.5. Broadcast Mode Should Only Be Used On Trusted Networks

Per RFC 5905 [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 the broadcast server uses this key to append a message authentication code (MAC) to the broadcast packets it sends.

Importantly, all broadcast clients that listen to this server must know the 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 broadcast client can use its knowledge of this key to attack the other broadcast clients.

For this reason, an NTP broadcast server and all its client must 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. Note, however, that an adversarial client that knows the symmetric broadcast key could still easily spoof its source IP to an IP that is allowed to serve time. (This is easy to do because the origin timestamp on broadcast mode packets is not validated by the client. By contrast, client/server and symmetric modes do require origin timestamp validation, making it more difficult to spoof packets [CCR16].

6.6. 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 per peer 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.

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.

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 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.

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, because their customers 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 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 [14].

7.2. 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 to connect to.

7.2.1. 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]. (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.

NOTE WELL: Using a single anycast address for NTP should be done with care. It means each client will likely use a single time server source. A key element of a robust NTP deployment is each client using multiple sources of time. With multiple time sources, a client will analyze the various time sources, selecting good ones, and disregarding poor ones. If a single Anycast address is used, this analysis will not happen.

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 only 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. If 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 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, Miroslav Lichvar, Daniel Fox Franke, and Robert Nagy.

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 may assume that system time cannot be changed and is always monotonic, and vulnerabilites may 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 encourages 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>.

[MILLS2006] Mills, D., "Computer network time synchronization: the Network Time Protocol", CRC Press , 2006.

[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] http://bk.ntp.org/

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

[4] http://www.bcp38.info

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

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

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

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

[11] http://bk1.ntp.org/ntp-stable/README.leapsmear?PAGE=anno

[13] https://lists.ntp.org/pipermail/ntpwg/2011-August/001714.html

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

Authors' Addresses

Denis Reilly (editor)
Spectracom
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