

MODULAR PROGRAMME

COURSEWORK ASSESSMENT SPECIFICATION

Module Details

Module Code UFCFQ4-30-2	Run 16SEP/1	Module Title COMPUTER NETWORKS AND OPERATING SYSTEMS
Module Leader Ian Johnson/Ben Gaster	Module Coordinator	Module Tutors Benedict Gaster, Laurence O'Brien
Component and Element Number B: CW1		Weighting: (% of the Module's assessment) 25
Element Description PRACTICAL COURSEWORK (Practical Coursework)		Total Assignment time 12 + Lab time

Dates

Date Issued to Students 27/09/16	Date to be Returned to Students 12/01/2017
Submission Place Coursework Hub (Level 1 A Block Underpass)	Submission Date 08/12//2016
	Submission Time 2.00 pm

Deliverables

As per the attached specification

Module Leader Signature

Ian Johnson

Coursework 1 – Computer Networks and Operating Systems.

This coursework is concerned with developing and demonstrating skills in network programming.

Part A – INDIVIDUAL WORK

Complete the worksheets (found under module lab schedule) on blackboard. These need to be demonstrated to your lab tutor before 18/11/2015. **After this date lab tutors will concentrate on assisting students with part B.** The sign-offs have questions to test your understanding. These answers are less important than being able to demonstrate working code.

Worksheet 1 is worth up to 5%

Worksheets 2-4 successfully completed and demonstrated will be worth up to 10% each (Total 35%).

You need to submit the attached signoff sheets signed and dated by your tutor.

Sign-offs will only be done in the scheduled lab sessions.

Part B – IN PAIRS OR INDIVIDUAL WORK

Where work is done in pairs then a single hand-in containing both solo and collaborative work should be submitted with a **group cover sheet showing both student IDs**

The second part of the assignment is to develop an RFC compliant SNTP client and server. You can test your client against ntp.uwe.ac.uk if on campus. If working at home your ISP is likely to provide an NTP service.

Your work **MUST** be demonstrated on linux in the lab. Failing to demonstrate your work will cost you significant marks. Again, **demonstrations must be done during scheduled lab sessions.** RFC4330 is attached for your reference. This is the documentation for the protocol from which you will need to extract the process, packet format etc. that you intend to implement.

Up to 40% is available for the development of a unicast client & server, with a further 25% available for extending these to support the multicast paradigm.

You need to submit a design (a Finite State diagram is ideal), a test plan, and the results of your testing together with the signoff sheet and your source code for this part of the assignment. Marking will be based on functionality, code quality and the quality of your test plan. The test plan and results are intended to show that you have thought about the circumstances under which your code may fail and have thoroughly tested these.

You may consult <https://www.eecis.udel.edu/%7Emills/ntp/html/sntp.html> for suggestions for command line options for your client & server, together with the expected output format. Please note that you will need to be able to specify the port to send to for the client, and listen on for the server as you will not be able to start your server on port 123 in the lab due to privilege issues.

As an example, for the unicast client, minimally you are expected to:

- Use the talker code and command line options to initialise the NTP data frame, send packet and analyse response.
- Print response according to SNTP output given by Dave Mills
<https://www.eecis.udel.edu/~mills/ntp/html/sntp.html>
- Calculation using t1,t2,t3,t4 T1 etc.
- Error detection ie time-out, invalid host

This could then be improved by:

- Screen depiction of clock adjustment
- Timed repeat updates
- Use of a configuration file
- Timing out server and trying a different one on no response.

And then further by:

- Confidence checking of response
- Kiss-o'Death
- Status codes checks
- Use of threads
- Use of getopt()

For the test plan, you will need to:

- Compile requirements from RFC
- Derive tests from requirements
- Use network tools to view and verify data (e.g. wireshark or tcp dump)
- Provide evidence

Your code quality will be assessed on the following components

- Header
- Acknowledgements
- History
- Declarations
- Comments
- Functions with parameters and return values (talker() and listener() given)
- Use of headers
- Correlation with Design
- Makefile

Similarly for the unicast server and multicast client & server. Note that without well written, functional, demonstrated code you are unlikely to achieve good mark on that section.

The code you submit MUST be your own. **ALL submissions of code should be via the UWE Gitlab.** Your lab tutor may require you to demonstrate you understand the code before signing it off. **If you are using any code developed by others then you MUST ensure that the code you are submitting for marking is ONLY YOUR OWN.** Any code you wish to use that is not your own MUST be in a separate source file. **Source files containing other people's code will not receive any marks.**

NOTE: You are only expected to implement Unicast and Multicast. Other modes are not necessary. You can ignore Leap Second indicators.

SIGNOFFS will ONLY be completed in scheduled lab time by your own tutor. You can demo your client and server in lab time throughout this teaching block.

Component	Marks	Deliverables
Worksheets	35	Signoff sheets
Unicast SNTP client & server	40	Design, Test plan & results Demo sheet & Code
Manycast client & server	25	Design, Test plan & results Demo sheet & Code

UFCFQ4-30-2	Student ID....		
Worksheet 1	Answer	Tutor	Date
MTU is ..?			
Active close by?			
echo port number?			
daytime port number?			
TCP protocol No.?			
UDP protocol No.?			
Worksheet 2	Observations	Tutor	Date
Client Compiled?			
What happens when using names?			
dServer Compiled?			
PS & netstat -ap?			
telnet results? your server.			
telnet results? other server.			
dServName Compiled?			
PS & netstat -ap?			
telnet results? your server.			
telnet results? other server.			

UFCFQ4-30-2	Student ID....		
Worksheet 3	Observations	Tutor	Date
Server compiles cleanly			
Server runs, PID is?			
telnet to server			
Client compiles cleanly			
Client runs, PID is?			
Running multiple clients How many servers?			
What happens when .. original server killed			
using existing client			
starting new client			
starting new server			
further observations			

UFCFQ4-30-2	Student ID....		
Worksheet 4	Observations	Tutor	Date
Datagram size is?			
If Client starts first?			
Long message, 10 byte buffer?			
Telnet web8000?			
Browser connects to web8000?			
HTML formatted output displays correctly?			
Web8000 generates a 404?			

Assignment signoffs	Comments	Tutor	Date
Standalone client against standard snmp server			
Standalone server against client			
Working with other students clients & server?			
Manycast client			
Manycast server			

Appendix A – RFC4330

Network Working Group
Request for Comments: 4330
Obsoletes: 2030, 1769
Category: Informational

D. Mills
University of Delaware
January 2006

Simple Network Time Protocol (SNTP) Version 4 for IPv4, IPv6 and OSI

Status of This Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

This memorandum describes the Simple Network Time Protocol Version 4 (SNTPv4), which is a subset of the Network Time Protocol (NTP) used to synchronize computer clocks in the Internet. SNTPv4 can be used when the ultimate performance of a full NTP implementation based on RFC 1305 is neither needed nor justified. When operating with current and previous NTP and SNTP versions, SNTPv4 requires no changes to the specifications or known implementations, but rather clarifies certain design features that allow operation in a simple, stateless remote-procedure call (RPC) mode with accuracy and reliability expectations similar to the UDP/TIME protocol described in RFC 868.

This memorandum obsoletes RFC 1769, which describes SNTP Version 3 (SNTPv3), and RFC 2030, which describes SNTPv4. Its purpose is to correct certain inconsistencies in the previous documents and to clarify header formats and protocol operations for NTPv3 (IPv4) and SNTPv4 (IPv4, IPv6, and OSI), which are also used for SNTP. A further purpose is to provide guidance for home and business client implementations for routers and other consumer devices to protect the server population from abuse. A working knowledge of the NTPv3 specification, RFC 1305, is not required for an implementation of SNTP.

Table of Contents

1. Introduction	2
1.1. Specification of Requirements	5
2. Operating Modes and Addressing	5
3. NTP Timestamp Format	6
4. Message Format	8
5. SNTP Client Operations	13
6. SNTP Server Operations	16
7. Configuration and Management	19
8. The Kiss-o'-Death Packet	20
9. On Being a Good Network Citizen	21
10. Best Practices	21
11. Security Considerations	24
12. Acknowledgements	24
13. Contributors	24
14. Informative References	25

1. Introduction

The Network Time Protocol Version 3 (NTPv3), specified in RFC 1305 [MIL92], is widely used to synchronize computer clocks in the global Internet. It provides comprehensive mechanisms to access national time and frequency dissemination services, organize the NTP subnet of servers and clients, and adjust the system clock in each participant. In most places of the Internet of today, NTP provides accuracies of 1-50 ms, depending on the characteristics of the synchronization source and network paths.

RFC 1305 specifies the NTP protocol machine in terms of events, states, transition functions and actions, and engineered algorithms to improve the timekeeping quality and to mitigate several synchronization sources, some of which may be faulty. To achieve accuracies in the low milliseconds over paths spanning major portions of the Internet, these intricate algorithms, or their functional equivalents, are necessary. In many applications, accuracies on the order of significant fractions of a second are acceptable. In simple home router applications, accuracies of up to a minute may suffice. In such cases, simpler protocols, such as the Time Protocol specified in RFC 868 [POS83], have been used for this purpose. These protocols involve an RPC exchange where the client requests the time of day and the server returns it in seconds past a known reference epoch.

NTP is designed for use by clients and servers with a wide range of capabilities and over a wide range of network jitter and clock frequency wander characteristics. Many users of NTP in the Internet of today use a software distribution available from www.ntp.org. The distribution, which includes the full suite of NTP options,

mitigation algorithms, and security schemes, is a relatively complex, real-time application. Although the software has been ported to a wide variety of hardware platforms ranging from personal computers to supercomputers, its sheer size and complexity is not appropriate for many applications. Accordingly, it is useful to explore alternative strategies using simpler software appropriate for less stringent accuracy expectations.

This memo describes the Simple Network Time Protocol Version 4 (SNTPv4), which is a simplified access paradigm for servers and clients using current and previous versions of NTP and SNTP. The access paradigm is identical to the UDP/TIME Protocol, and, in fact, it should be easy to adapt a UDP/TIME client implementation, say for a personal computer, to operate using SNTP. Moreover, SNTP is also designed to operate in a dedicated server configuration including an integrated radio clock. With careful design and control of the various latencies in the system, which is practical in a dedicated design, it is possible to deliver time accurate on the order of microseconds.

The only significant protocol change in SNTPv4 from previous SNTP versions is a modified header interpretation to accommodate Internet Protocol Version 6 (IPv6) (RFC 2460) and OSI (RFC 1629) addressing. However, SNTPv4 includes certain optional extensions to the basic NTP Version 3 (NTPv3) model, including a multicast mode and a public-key-based authentication scheme designed specifically for broadcast and multicast applications. Although the multicast mode is described in this memo, the authentication scheme is described in another RFC to be submitted later. Until such time that a definitive NTPv4 specification is published, the multicast and authentication features should be considered provisional. In addition, this memo introduces the kiss-o'-death message, which can be used by servers to suppress client requests as circumstances require.

When operating with current and previous versions of NTP and SNTP, SNTPv4 requires no changes to the protocol or implementations now running or likely to be implemented specifically for future NTP or SNTP versions. The NTP and SNTP packet formats are the same, and the arithmetic operations to calculate the client time, clock offset, and roundtrip delay are the same. To an NTP or SNTP server, NTP and SNTP clients are indistinguishable; to an NTP or SNTP client, NTP and SNTP servers are indistinguishable. Like NTP servers operating in non-symmetric modes, SNTP servers are stateless and can support large numbers of clients; however, unlike most NTP clients, SNTP clients normally operate with only a single server at a time.

The full degree of reliability ordinarily expected of NTP servers is possible only using redundant sources, diverse paths, and the crafted

Mills	Informational	[Page 3]
RFC 4330	SNTPv4 for IPv4, IPv6 and OSI	January 2006

algorithms of a full NTP implementation. It is strongly recommended that SNTP clients be used only at the extremities of the synchronization subnet. SNTP clients should operate only at the leaves (highest stratum) of the subnet and in configurations where no NTP or SNTP client is dependent on another SNTP client for

synchronization. SNTP servers should operate only at the root (stratum 1) of the subnet, and then only in configurations where no other source of synchronization other than a reliable radio clock or telephone modem is available.

An important provision in this memo is the interpretation of certain NTP header fields that provide for IPv6 [DEE98] and OSI [COL94] addressing. The only significant difference between the NTP and SNTPv4 header formats is the four-octet Reference Identifier field, which is used primarily to detect and avoid synchronization loops. In all NTP and SNTP versions providing IPv4 addressing, primary servers use a four-character ASCII reference clock identifier in this field, whereas secondary servers use the 32-bit IPv4 address of the synchronization source. In SNTPv4 providing IPv6 and OSI addressing, primary servers use the same clock identifier, but secondary servers use the first 32 bits of the MD5 hash of the IPv6 or NSAP address of the synchronization source. A further use of this field is when the server sends a kiss-o'-death message, documented later in this memo.

NTP Version 4 (NTPv4), now in deployment, but not yet the subject of a standards document, uses the same Reference Identifier field as SNTPv4.

In the case of OSI, the Connectionless Transport Service (CLTS) is used as in [ISO86]. Each SNTP packet is transmitted as the TS-Userdata parameter of a T-UNITDATA Request primitive. Alternately, the header can be encapsulated in a Transport Protocol Data Unit (TPDU), which itself is transported using UDP, as described in RFC 1240 [DOB91]. It is not advised that NTP be operated at the upper layers of the OSI stack, such as might be inferred from RFC 1698 [FUR94], as this could seriously degrade accuracy. With the header formats defined in this memo, it is in principle possible to interwork between servers and clients of one protocol family and another, although the practical difficulties may make this inadvisable.

In the following, indented paragraphs such as this one contain information not required by the formal protocol specification, but considered good practice in protocol implementations.

This memo is organized as follows. Section 2 describes how the protocol works, the various modes, and how IP addresses and UDP ports are used. Section 3 describes the NTP timestamp format, and Section

Mills Informational [Page 4]

RFC 4330 SNTPv4 for IPv4, IPv6 and OSI January 2006

4 the NTP message format. Section 5 summarizes SNTP client operations, and Section 6 summarizes SNTP server operations. Section 7 summarizes operation and management issues. Section 8 describes the kiss-o'-death message, newly minted with functions similar to the ICMP Source Quench and ICMP Destination Unreachable messages. Section 9 summarizes design issues important for good network citizenry and presents an example algorithm designed to give good reliability while minimizing network and server resource demands.

1.1. Specification of Requirements

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 [BRA97].

2. Operating Modes and Addressing

Unless excepted in context, a reference to broadcast address means IPv4 broadcast address, IPv4 multicast group address, or IPv6 address of appropriate scope. Further information on the broadcast/multicast model is in RFC 1112 [DEE89]. Details of address format, scoping rules, etc., are beyond the scope of this memo. SNTPv4 can operate with either unicast (point to point), broadcast (point to multipoint), or anycast (multipoint to point) addressing modes. A unicast client sends a request to a designated server at its unicast address and expects a reply from which it can determine the time and, optionally, the roundtrip delay and clock offset relative to the server. A broadcast server periodically sends an unsolicited message to a designated broadcast address. A broadcast client listens on this address and ordinarily sends no requests.

Manycast is an extension of the anycast paradigm described in RFC 1546 [PAR93]. It is designed for use with a set of cooperating servers whose addresses are not known beforehand. The manycast client sends an ordinary NTP client request to a designated broadcast address. One or more manycast servers listen on that address. Upon receiving a request, a manycast server sends an ordinary NTP server reply to the client. The client then mobilizes an association for each server found and continues operation with all of them. Subsequently, the NTP mitigation algorithms operate to cast out all except the best three.

Broadcast servers should respond to client unicast requests, as well as send unsolicited broadcast messages. Broadcast clients may send unicast requests in order to measure the network propagation delay between the server and client and then continue operation in listen-only mode. However, broadcast servers may

Mills	Informational	[Page 5]
RFC 4330	SNTPv4 for IPv4, IPv6 and OSI	January 2006

choose not to respond to unicast requests, so unicast clients should be prepared to abandon the measurement and assume a default value for the delay.

The client and server addresses are assigned following the usual IPv4, IPv6 or OSI conventions. For NTP multicast, the IANA has reserved the IPv4 group address 224.0.1.1 and the IPv6 address ending :101 with appropriate scope. The NTP broadcast address for OSI has yet to be determined. Notwithstanding the IANA reserved addresses, other multicast addresses can be used that do not conflict with others assigned in scope. The scoping, routing, and group membership procedures are determined by considerations beyond the scope of this memo.

It is important to adjust the time-to-live (TTL) field in the IP

header of multicast messages to a reasonable value in order to limit the network resources used by this (and any other) multicast service. Only multicast clients in scope will receive multicast server messages. Only cooperating manycast servers in scope will reply to a client request. The engineering principles that determine the proper values to be used are beyond the scope of this memo.

In the case of SNTP as specified herein, there is a very real vulnerability that SNTP broadcast clients can be disrupted by misbehaving or hostile SNTP or NTP broadcast servers elsewhere in the Internet. It is strongly recommended that access controls and/or cryptographic authentication means be provided for additional security in such cases.

It is intended that IP broadcast addresses will be used primarily in IP subnets and LAN segments including a fully functional NTP server with a number of dependent SNTP broadcast clients on the same subnet, and that IP multicast group addresses will be used only in cases where the TTL is engineered specifically for each service domain. However, these uses are not integral to the SNTP specification.

3. NTP Timestamp Format

SNTP uses the standard NTP timestamp format described in RFC 1305 and previous versions of that document. In conformance with standard Internet practice, NTP data are specified as integer or fixed-point quantities, with bits numbered in big-endian fashion from 0 starting at the left or most significant end. Unless specified otherwise, all quantities are unsigned and may occupy the full field width with an implied 0 preceding bit 0.

Mills	Informational	[Page 6]
RFC 4330	SNTPv4 for IPv4, IPv6 and OSI	January 2006

Because NTP timestamps are cherished data and, in fact, represent the main product of the protocol, a special timestamp format has been established. NTP timestamps are represented as a 64-bit unsigned fixed-point number, in seconds relative to 0h on 1 January 1900. The integer part is in the first 32 bits, and the fraction part in the last 32 bits. In the fraction part, the non-significant low-order bits are not specified and are ordinarily set to 0.

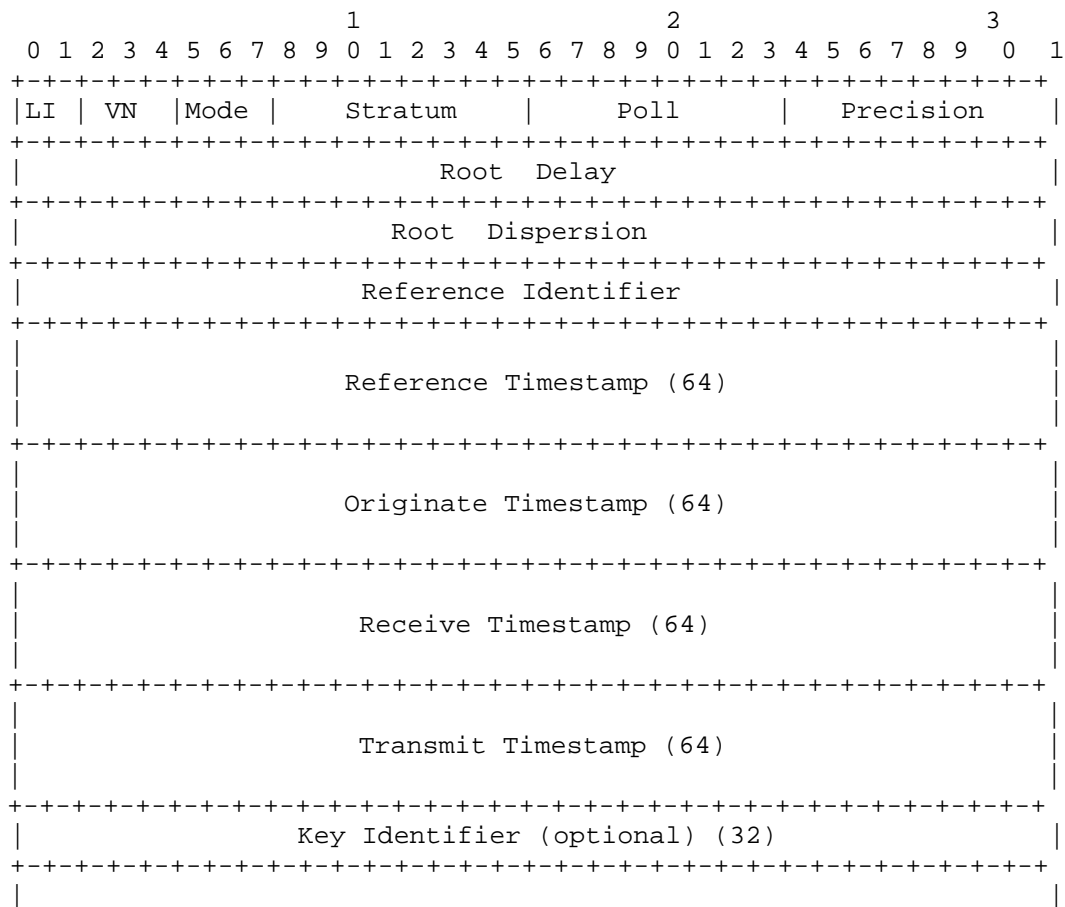
It is advisable to fill the non-significant low-order bits of the timestamp with a random, unbiased bitstring, both to avoid systematic roundoff errors and to provide loop detection and replay detection (see below). It is important that the bitstring be unpredictable by an intruder. One way of doing this is to generate a random 128-bit bitstring at startup. After that, each time the system clock is read, the string consisting of the timestamp and bitstring is hashed with the MD5 algorithm, then the non-significant bits of the timestamp are copied from the result.

The NTP format allows convenient multiple-precision arithmetic and conversion to UDP/TIME message (seconds), but does complicate the

Figure 1 is a description of the NTP and SNTP message format, which follows the IP and UDP headers in the message. This format is identical to the NTP message format described in RFC 1305, with the exception of the Reference Identifier field described below. For SNTP client messages, most of these fields are zero or initialized with pre-specified data. For completeness, the function of each field is briefly summarized below.

[Page 8]

January 2006



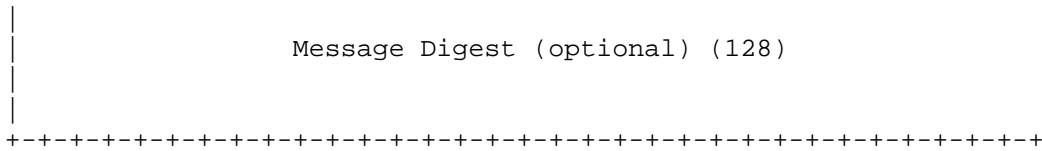


Figure 1. NTP Packet Header

Leap Indicator (LI): This is a two-bit code warning of an impending leap second to be inserted/deleted in the last minute of the current day. This field is significant only in server messages, where the values are defined as follows:

LI	Meaning

0	no warning
1	last minute has 61 seconds
2	last minute has 59 seconds
3	alarm condition (clock not synchronized)

On startup, servers set this field to 3 (clock not synchronized), and set this field to some other value when synchronized to the primary reference clock. Once set to a value other than 3, the field is never set to that value again, even if all synchronization sources become unreachable or defective.

Version Number (VN): This is a three-bit integer indicating the NTP/SNTP version number, currently 4. If necessary to distinguish between IPv4, IPv6, and OSI, the encapsulating context must be inspected.

Mode: This is a three-bit number indicating the protocol mode. The values are defined as follows:

Mode	Meaning

0	reserved
1	symmetric active
2	symmetric passive
3	client
4	server
5	broadcast
6	reserved for NTP control message
7	reserved for private use

In unicast and multicast modes, the client sets this field to 3 (client) in the request, and the server sets it to 4 (server) in the reply. In broadcast mode, the server sets this field to 5

(broadcast). The other modes are not used by SNTP servers and clients.

Stratum: This is an eight-bit unsigned integer indicating the stratum. This field is significant only in SNTP server messages, where the values are defined as follows:

Stratum	Meaning
0	kiss-o'-death message (see below)
1	primary reference (e.g., synchronized by radio clock)
2-15	secondary reference (synchronized by NTP or SNTP)
16-255	reserved

Mills

Informational

[Page 10]

RFC 4330

SNTPv4 for IPv4, IPv6 and OSI

January 2006

Poll Interval: This is an eight-bit unsigned integer used as an exponent of two, where the resulting value is the maximum interval between successive messages in seconds. This field is significant only in SNTP server messages, where the values range from 4 (16 s) to 17 (131,072 s -- about 36 h).

Precision: This is an eight-bit signed integer used as an exponent of two, where the resulting value is the precision of the system clock in seconds. This field is significant only in server messages, where the values range from -6 for mains-frequency clocks to -20 for microsecond clocks found in some workstations.

Root Delay: This is a 32-bit signed fixed-point number indicating the total roundtrip delay to the primary reference source, in seconds with the fraction point between bits 15 and 16. Note that this variable can take on both positive and negative values, depending on the relative time and frequency offsets. This field is significant only in server messages, where the values range from negative values of a few milliseconds to positive values of several hundred milliseconds.

Code	External Reference Source
LOCL	uncalibrated local clock
CESM	calibrated Cesium clock
RBDM	calibrated Rubidium clock
PPS	calibrated quartz clock or other pulse-per-second source
IRIG	Inter-Range Instrumentation Group
ACTS	NIST telephone modem service
USNO	USNO telephone modem service
PTB	PTB (Germany) telephone modem service
TDF	Allouis (France) Radio 164 kHz
DCF	Mainflingen (Germany) Radio 77.5 kHz
MSF	Rugby (UK) Radio 60 kHz
WWV	Ft. Collins (US) Radio 2.5, 5, 10, 15, 20 MHz
WWVB	Boulder (US) Radio 60 kHz
WWVH	Kauai Hawaii (US) Radio 2.5, 5, 10, 15 MHz
CHU	Ottawa (Canada) Radio 3330, 7335, 14670 kHz
LORC	LORAN-C radionavigation system

Figure 2. Reference Identifier Codes

Mills	Informational	[Page 11]
RFC 4330	SNTPv4 for IPv4, IPv6 and OSI	January 2006

Root Dispersion: This is a 32-bit unsigned fixed-point number indicating the maximum error due to the clock frequency tolerance, in seconds with the fraction point between bits 15 and 16. This field is significant only in server messages, where the values range from zero to several hundred milliseconds.

Reference Identifier: This is a 32-bit bitstring identifying the particular reference source. This field is significant only in server messages, where for stratum 0 (kiss-o'-death message) and 1 (primary server), the value is a four-character ASCII string, left justified and zero padded to 32 bits. For IPv4 secondary servers, the value is the 32-bit IPv4 address of the synchronization source. For IPv6 and OSI secondary servers, the value is the first 32 bits of the MD5 hash of the IPv6 or NSAP address of the synchronization source.

Primary (stratum 1) servers set this field to a code identifying the external reference source according to Figure 2. If the external reference is one of those listed, the associated code should be used. Codes for sources not listed can be contrived, as appropriate.

In previous NTP and SNTP secondary servers and clients, this field was often used to walk-back the synchronization subnet to the root (primary server) for management purposes. In SNTPv4 with IPv6 or OSI, this feature is not available, because the addresses are longer than 32 bits, and only a hash is available. However, a walk-back can be accomplished using the NTP control message and the reference identifier field described in RFC 1305.

Reference Timestamp: This field is the time the system clock was last set or corrected, in 64-bit timestamp format.

Originate Timestamp: This is the time at which the request departed the client for the server, in 64-bit timestamp format.

Receive Timestamp: This is the time at which the request arrived at the server or the reply arrived at the client, in 64-bit timestamp format.

Transmit Timestamp: This is the time at which the request departed the client or the reply departed the server, in 64-bit timestamp format.

Authenticator (optional): When the NTP authentication scheme is implemented, the Key Identifier and Message Digest fields contain the

5. SNTP Client Operations

An SNTP client can operate in unicast, broadcast, or multicast modes. In unicast mode, the client sends a request (NTP mode 3) to a designated unicast server and expects a reply (NTP mode 4) from that server. In broadcast client mode, it sends no request and waits for a broadcast (NTP mode 5) from one or more broadcast servers. In multicast mode, the client sends a request (NTP mode 3) to a designated broadcast address and expects a reply (NTP mode 4) from one or more multicast servers. The client uses the first reply received to establish the particular server for subsequent unicast operations. Later replies from this server (duplicates) or any other server are ignored. Other than the selection of address in the request, the operations of multicast and unicast clients are identical.

Client requests are normally sent at intervals depending on the frequency tolerance of the client clock and the required accuracy. However, under no conditions should requests be sent at less than one minute intervals. Further discussion on this point is in Section 9.

A unicast or multicast client initializes the NTP message header, sends the request to the server, and strips the time of day from the Transmit Timestamp field of the reply. For this purpose, all the NTP header fields shown above are set to 0, except the Mode, VN, and optional Transmit Timestamp fields.

NTP and SNTP clients set the mode field to 3 (client) for unicast and multicast requests. They set the VN field to any version number that is supported by the server, selected by configuration or discovery, and that can interoperate with all previous version NTP and SNTP servers. Servers reply with the same version as the request, so the VN field of the request also specifies the VN field of the reply. A prudent SNTP client can specify the earliest acceptable version on the expectation that any server of that or a later version will respond. NTP Version 3 (RFC 1305) and Version 2 (RFC 1119) servers accept all previous versions, including Version 1 (RFC 1059). Note that Version 0 (RFC 959) is no longer supported by current and future NTP and SNTP servers.

Although setting the Transmit Timestamp field in the request to the time of day according to the client clock in NTP timestamp format is not necessary in a conforming client implementation, it is highly recommended in unicast and multicast modes. This allows a simple calculation to determine the propagation delay between the server and client and to align the system clock generally within a few tens of milliseconds relative to the server. In addition, this provides a

simple method for verifying that the server reply is in fact a legitimate response to the specific client request and thereby for avoiding replays. In broadcast mode, the client has no information to calculate the propagation delay or to determine the validity of the server, unless one of the NTP authentication schemes is used.

To calculate the roundtrip delay d and system clock offset t relative to the server, the client sets the Transmit Timestamp field in the request to the time of day according to the client clock in NTP timestamp format. For this purpose, the clock need not be synchronized. The server copies this field to the Originate Timestamp in the reply and sets the Receive Timestamp and Transmit Timestamp fields to the time of day according to the server clock in NTP timestamp format.

When the server reply is received, the client determines a Destination Timestamp variable as the time of arrival according to its clock in NTP timestamp format. The following table summarizes the four timestamps.

Timestamp Name	ID	When Generated
-----	-----	-----
Originate Timestamp	T1	time request sent by client
Receive Timestamp	T2	time request received by server
Transmit Timestamp	T3	time reply sent by server
Destination Timestamp	T4	time reply received by client

The roundtrip delay d and system clock offset t are defined as:

$$d = (T4 - T1) - (T3 - T2) \quad t = ((T2 - T1) + (T3 - T4)) / 2.$$

Note that in general both delay and offset are signed quantities and can be less than zero; however, a delay less than zero is possible only in symmetric modes, which SNTP clients are forbidden to use. The following table summarizes the required SNTP client operations in unicast, multicast, and broadcast modes. The recommended error checks are shown in the Reply and Broadcast columns in the table. The message should be considered valid only if all the fields shown contain values in the respective ranges. Whether to believe the message if one or more of the fields marked "ignore" contain invalid values is at the discretion of the implementation.

Field Name	Unicast/Manycast Request	Reply	Broadcast
LI	0	0-3	0-3
VN	1-4	copied from request	1-4
Mode	3	4	5
Stratum	0	0-15	0-15
Poll	0	ignore	ignore
Precision	0	ignore	ignore
Root Delay	0	ignore	ignore
Root Dispersion	0	ignore	ignore
Reference Identifier	0	ignore	ignore
Reference Timestamp	0	ignore	ignore
Originate Timestamp	0	(see text)	ignore
Receive Timestamp	0	(see text)	ignore
Transmit Timestamp	(see text)	nonzero	nonzero
Authenticator	optional	optional	optional

Although not required in a conforming SNTP client implementation, it is wise to consider a suite of sanity checks designed to avoid various kinds of abuse that might happen as the result of server implementation errors or malicious attack. Following is a list of suggested checks.

1. When the IP source and destination addresses are available for the client request, they should match the interchanged addresses in the server reply.
2. When the UDP source and destination ports are available for the client request, they should match the interchanged ports in the server reply.
3. The Originate Timestamp in the server reply should match the Transmit Timestamp used in the client request.

Mills Informational [Page 15]

RFC 4330 SNTPv4 for IPv4, IPv6 and OSI January 2006

4. The server reply should be discarded if any of the LI, Stratum, or Transmit Timestamp fields is 0 or the Mode field is not 4 (unicast) or 5 (broadcast).

5. A truly paranoid client can check that the Root Delay and Root Dispersion fields are each greater than or equal to 0 and less than infinity, where infinity is currently a cozy number like one second. This check avoids using a server whose synchronization source has expired for a very long time.

6. SNTP Server Operations

A SNTP server operating with either an NTP or SNTP client of the same or previous versions retains no persistent state. Because an SNTP server ordinarily does not implement the full suite of grooming and mitigation algorithms intended to support redundant servers and diverse network paths, it should be operated only in conjunction with a source of external synchronization, such as a reliable radio clock or telephone modem. In this case it operates as a primary (stratum 1) server.

A SNTP server can operate with any unicast, multicast, or broadcast address or any combination of these addresses. A unicast or multicast server receives a request (NTP mode 3), modifies certain fields in the NTP header, and sends a reply (NTP mode 4), possibly using the same message buffer as the request. A multicast server listens on the designated broadcast address, but uses its own unicast IP address in the source address field of the reply. Other than the selection of address in the reply, the operations of multicast and unicast servers are identical. Broadcast messages are normally sent at intervals from 64 s to 1024 s, depending on the expected frequency tolerance of the client clocks and the required accuracy.

Unicast and multicast servers copy the VN and Poll fields of the request intact to the reply and set the Stratum field to 1.

Note that SNTP servers normally operate as primary (stratum 1) servers. Although operating at higher strata (up to 15) while synchronizing to an external source such as a GPS receiver is not forbidden, this is strongly discouraged.

If the Mode field of the request is 3 (client), the reply is set to 4 (server). If this field is set to 1 (symmetric active), the reply is set to 2 (symmetric passive). This allows clients configured in either client (NTP mode 3) or symmetric active (NTP mode 1) to interoperate successfully, even if configured in possibly suboptimal ways. For any other value in the Mode field, the request is

Mills	Informational	[Page 16]
RFC 4330	SNTPv4 for IPv4, IPv6 and OSI	January 2006

discarded. In broadcast (unsolicited) mode, the VN field is set to 4, the Mode field is set to 5 (broadcast), and the Poll field set to the nearest integer base-2 logarithm of the poll interval.

Note that it is highly desirable that a broadcast server also supports unicast clients. This is so a potential broadcast client can calculate the propagation delay using a client/server exchange prior to switching to broadcast client (listen-only) mode. By design, a multicast server is also a unicast server. There does

not seem to be a great advantage for a server to operate as both broadcast and multicast at the same time, although the protocol specification does not forbid it.

A broadcast or multicast server does not send packets if not synchronized to a correctly operating reference source. It may or may not respond to a client request if it is not synchronized, but the preferred option is to respond because this allows reachability to be determined regardless of synchronization state. If the server has never synchronized to a reference source, the LI field is set to 3 (unsynchronized). Once synchronized to a reference source, the LI field is set to one of the other three values and remains at the last value set even if the reference source becomes unreachable or turns faulty.

If the server is synchronized to a reference source, the Stratum field is set to 1, and the Reference Identifier field is set to the ASCII source identifier shown in Figure 2. If the server is not synchronized, the Stratum field is set to zero, and the Reference Identifier field is set to an ASCII error identifier described below.

The Precision field is set to reflect the maximum reading error of the system clock. For all practical cases it is computed as the negative base-2 logarithm of the number of significant bits to the right of the decimal point in the NTP timestamp format. The Root Delay and Root Dispersion fields are set to 0 for a primary server.

The timestamp fields in the server message are set as follows. If the server is unsynchronized or first coming up, all timestamp fields are set to zero, with one exception. If the message is a reply to a previously received client request, the Transmit Timestamp field of the request is copied unchanged to the Originate Timestamp field of the reply. It is important that this field be copied intact, as an NTP or SNTP client uses it to avoid bogus messages.

If the server is synchronized, the Reference Timestamp is set to the time the last update was received from the reference source. The Originate Timestamp field is set as in the unsynchronized case above. The Transmit Timestamp field is set to the time of day when the

Mills	Informational	[Page 17]
RFC 4330	SNTPv4 for IPv4, IPv6 and OSI	January 2006

message is sent. In broadcast messages the Receive Timestamp field is set to zero and copied from the Transmit Timestamp field in other messages. The following table summarizes these actions.

Field Name	Unicast/Multicast		Broadcast
	Request	Reply	
LI	ignore	as needed	as needed
VN	1-4	copied from request	4
Mode	3	4	5

be used for active operation and one of the others for backup should the active one fail or show an error condition. It is not normally useful to use more than one server at a time, as with millions of SNTP-enabled devices expected in the near future, such use would represent unnecessary drain on network and server resources.

Broadcast servers and multicast clients must be provided with the TTL and local broadcast or multicast group address. Unicast and multicast servers and broadcast clients may be configured with a list of address-mask pairs for access control, so that only those clients or servers known to be trusted will be accepted. Multicast servers and clients must implement the IGMP protocol and be provided with the local broadcast or multicast group address as well. The configuration data for cryptographic authentication is beyond the scope of this memo.

There are several scenarios that provide automatic server discovery and selection for SNTP clients with no pre-specified server configuration. For instance, a role server with CNAME such as pool.ntp.org returns a randomized list of volunteer secondary server addresses, and the client can select one or more as candidates. For an IP subnet or LAN segment including an NTP or SNTP server, SNTP clients can be configured as broadcast clients. The same approach can be used with multicast servers and clients. In both cases, provision of an access control list is a good way to ensure that only trusted sources can be used to set the system clock.

Mills

Informational

[Page 19]

RFC 4330

SNTPv4 for IPv4, IPv6 and OSI

January 2006

In another scenario suitable for an extended network with significant network propagation delays, clients can be configured for multicast addresses, both upon initial startup and after some period when the currently selected unicast source has not been heard. Following the defined protocol, the client binds to the server from which the first reply is received and continues operation in unicast mode.

8. The Kiss-o'-Death Packet

In the rambunctious Internet of today, it is imperative that some means be available to tell a client to stop making requests and to go somewhere else. A recent experience involved a large number of home/office routers all configured to use a particular university time server. Under some error conditions, a substantial fraction of these routers would send packets at intervals of one second. The resulting traffic spike was dramatic, and extreme measures were required to diagnose the problem and to bring it under control. The conclusion is that clients must respect the means available to targeted servers to stop them from sending packets.

According to the NTP specification RFC 1305, if the Stratum field in the NTP header is 1, indicating a primary server, the Reference Identifier field contains an ASCII string identifying the particular reference clock type. However, in RFC 1305 nothing is said about the

every Internet appliance has some kind of NTP support, including Windows XP, Cisco routers, embedded controllers, and software systems of all kinds. This is the first edition of the SNTP RFC where it has become necessary to lay down rules of engagement in the form of design criteria for SNTP client implementations. This is necessary to educate software developers regarding the proper use of Internet time server resources as the Internet expands and demands on time servers increase, and to prevent the recurrence of the sort of problem mentioned above.

10. Best Practices

NTP and SNTP clients can consume considerable network and server resources if they are not good network citizens. There are now consumer Internet commodity devices numbering in the millions that are potential customers of public and private NTP and SNTP servers. Recent experience strongly suggests that device designers pay particular attention to minimizing resource impacts, especially if large numbers of these devices are deployed. The most important

Mills	Informational	[Page 21]
RFC 4330	SNTPv4 for IPv4, IPv6 and OSI	January 2006

design consideration is the interval between client requests, called the poll interval. It is extremely important that the design use the maximum poll interval consistent with acceptable accuracy.

1. A client MUST NOT under any conditions use a poll interval less than 15 seconds.
2. A client SHOULD increase the poll interval using exponential backoff as performance permits and especially if the server does not respond within a reasonable time.
3. A client SHOULD use local servers whenever available to avoid unnecessary traffic on backbone networks.
4. A client MUST allow the operator to configure the primary and/or alternate server names or addresses in addition to or in place of a firmware default IP address.
5. If a firmware default server IP address is provided, it MUST be a server operated by the manufacturer or seller of the device or another server, but only with the operator's permission.
6. A client SHOULD use the Domain Name System (DNS) to resolve the server IP addresses, so the operator can do effective load balancing among a server clique and change IP address binding to canonical names.
7. A client SHOULD re-resolve the server IP address at periodic intervals, but not at intervals less than the time-to-live field in the DNS response.
8. A client SHOULD support the NTP access-refusal mechanism so that a server kiss-o'-death reply in response to a client request causes the client to cease sending requests to that server and to

switch to an alternate, if available.

The following algorithm can be used as a pattern for specific implementations. It uses the following variables:

Timer: This is a counter that decrements at a fixed rate. When it reaches zero, a packet is sent, and the timer is initialized with the timeout for the next packet.

Maximum timeout: This is the maximum timeout determined from the given oscillator frequency tolerance and the required accuracy.

Mills

Informational

[Page 22]

RFC 4330

SNTPv4 for IPv4, IPv6 and OSI

January 2006

Server Name: This is the DNS name of the server. There may be more than one of them, to be selected by some algorithm not considered here.

Server IP Address: This is the IPv4, IPv6, or OSI address of the server.

If the firmware or documentation includes specific server names, the names should be those the manufacturer or seller operates as a customer convenience or those for which specific permission has been obtained from the operator. A DNS request for a generic server name, such as ntp.mytimeserver.com, should result in a random selection of server IP addresses available for that purpose. Each time a DNS request is received, a new randomized list is returned. The client ordinarily uses the first address on the list.

When candidate SNTP or NTP servers are selected, it is imperative to respect the server operator's conditions of access. Lists of public servers and their conditions of access are available at www.ntp.org. A semi-automatic server discovery scheme using DNS is described at that site. Some ISPs operate public servers, although finding them via their help desks can be difficult.

A well-behaved client operates as follows (note that steps 2-4 constitute a synchronization loop):

1. Consider the specified frequency tolerance of the system clock oscillator. Define the required accuracy of the system clock, then calculate the maximum timeout. For instance, if the frequency tolerance is 200 parts per million (PPM) and the required accuracy is one minute, the maximum timeout is about 3.5 days. Use the longest maximum timeout possible given the system constraints to minimize time server aggregate load, but never make it less than 15 minutes.
2. When the client is first coming up or after reset, randomize the timeout from one to five minutes. This is to minimize shock when 3000 PCs are rebooted at the same time power is restored after a blackout. Assume at this time that the IP address is unknown and

that the system clock is unsynchronized. Otherwise, use the timeout value as calculated in previous loop steps. Note that it may be necessary to refrain from implementing the aforementioned random delay for some classes of International Computer Security Association (ICSA) certification.

Mills

Informational

[Page 23]

RFC 4330

SNTPv4 for IPv4, IPv6 and OSI

January 2006

3. When the timer reaches zero, if the IP address is not known, send a DNS query packet; otherwise, send an NTP request packet to that address. If no reply packet has been heard since the last timeout, double the timeout, but do not make it greater than the maximum timeout. If primary and secondary time servers have been configured, alternate queries between the primary and secondary servers when no successful response has been received.
4. If a DNS reply packet is received, save the IP address and continue at step 2. If a KoD packet is received, remove that time server from the list, activate the secondary time server, and continue at step 2. If a received packet fails the sanity checks, drop that packet and also continue at step 2. If a valid NTP packet is received, update the system clock, set the timeout to the maximum, and continue at step 2.

11. Security Considerations

Without cryptographic authentication, SNTPv4 service is vulnerable to disruption by misbehaving or hostile SNTP or NTP broadcast servers elsewhere in the Internet. It is strongly recommended that access controls and/or cryptographic authentication means be provided for additional security. This document includes protocol provisions for adding such security mechanisms, but it does not define the mechanisms themselves. A separate document [MIL03] in preparation will define a cryptographic security mechanism for SNTP.

12. Acknowledgements

Jeff Learman was helpful in developing the OSI model for this protocol. Ajit Thyagarajan provided valuable suggestions and corrections.

13. Contributors

D. Plonka

J. Montgomery

14. Informative References

- [BRA97] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [COL94] Colella, R., Callon, R., Gardner, E., and Y. Rekhter, "Guidelines for OSI NSAP Allocation in the Internet", RFC 1629, May 1994.
- [DEE89] Deering, S., "Host extensions for IP multicasting", STD 5, RFC 1112, August 1989.
- [DEE98] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.
- [DOB91] Shue, C., Haggerty, W., and K. Dobbins, "OSI connectionless transport services on top of UDP: Version 1", RFC 1240, June 1991.
- [FUR94] Furniss, P., "Octet Sequences for Upper-Layer OSI to Support Basic Communications Applications", RFC 1698, October 1994.
- [ISO86] International Standards 8602 - Information Processing Systems - OSI: Connectionless Transport Protocol Specification. International Standards Organization, December 1986.
- [MIL92] Mills, D., "Network Time Protocol (Version 3) Specification, Implementation and Analysis", RFC 1305, March 1992.
- [MIL03] Mills, D., "The Autokey Security Architecture, Protocol and Algorithms", <http://eecis.udel.edu/~mills/database/reports/stime/stime.pdf>, August 2003.
- [PAR93] Partridge, C., Mendez, T., and W. Milliken, "Host Anycasting Service", RFC 1546, November 1993.
- [POS80] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.
- [POS83] Postel, J. and K. Harrenstien, "Time Protocol", STD 26, RFC 868, May 1983.
- [SRI99] Srisuresh, P. and M. Holdrege, "IP Network Address Translator (NAT) Terminology and Considerations", RFC 2663, August 1999.

[SRI01] Srisuresh, P. and K. Egevang, "Traditional IP Network Address Translator (Traditional NAT)", RFC 3022, January 2001.

Author's Address

David L. Mills
Electrical and Computer Engineering Department
University of Delaware
Newark, DE 19716

Phone: (302) 831-8247
EMail: mills@udel.edu

Full Copyright Statement

Copyright (C) The Internet Society (2006).

This document is subject to the rights, licenses and restrictions contained in BCP 78 and at www.rfc-editor.org/copyright.html, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).