Network Working Group   D. L. . Mills
Internet-Draft   University of Delaware
Intended status: Informational   K.F. O'Donoghue, Ed.
Expires: April 02, 2014   Internet Society
                                                          D. L. . Hart
                                                         H. M. . Stenn
                                         Network Time Foundation, Inc.
                                                    P.F. Chimento, Ed.
                                          Johns Hopkins University/APL
                                                         October 2013

Control Messages Protocol for Use with Network Time Protocol Version 4
draft-odonoghue-ntpv4-control-02

Abstract

   This document describes the structure of the control messages used
   with the Network Time Protocol. These control messages can be used
   to monitor and control the Network Time Protocol application running
   on any IP network attached computer. The information in this
   informational RFC was originally described in Appendix B of RFC 1305.

Status of this Memo

   This Internet-Draft is submitted in full conformance with the
   provisions of BCP 78 and BCP 79.

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

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

   This Internet-Draft will expire on April 02, 2014.

Copyright Notice

and restrictions with respect to this document. Code Components
extracted from this document must include Simplified BSD License text
as described in Section 4.e of the Trust Legal Provisions and are
provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF
Contributions published or made publicly available before November
10, 2008. The person(s) controlling the copyright in some of this
material may not have granted the IETF Trust the right to allow
modifications of such material outside the IETF Standards Process.
Without obtaining an adequate license from the person(s) controlling
the copyright in such materials, this document may not be modified
outside the IETF Standards Process, and derivative works of it may
not be created outside the IETF Standards Process, except to format
it for publication as an RFC or to translate it into languages other
than English.

Table of Contents

1. Introduction

Editor's Note (to be removed prior to publication): The text below is
taken directly from RFC 1305. Input is requested to update the text
to reflect current practice. This is required to fully obsolete RFC
1305.

In a comprehensive network-management environment, facilities are
presumed available to perform routine NTP control and monitoring
functions, such as setting the leap-indicator bits at the primary
servers, adjusting the various system parameters and monitoring
regular operations. Ordinarily, these functions can be implemented
using a network-management protocol such as SNMP and suitable
extensions to the MIB database. However, in those cases where such
facilities are not available, these functions can be implemented
using special NTP control messages described herein. These messages
are intended for use only in systems where no other management
facilities are available or appropriate, such as in dedicated-
function bus peripherals. Support for these messages is not required
in order to conform to this specification.

The NTP Control Message has the value 6 specified in the mode field
of the first octet of the NTP header and is formatted as shown below.
The format of the data field is specific to each command or response;
however, in most cases the format is designed to be constructed and
viewed by humans and so is coded in free-form ASCII. This facilitates
the specification and implementation of simple management tools in
the absence of fully evolved network-management facilities. As in
ordinary NTP messages, the authenticator field follows the data
field. If the authenticator is used the data field is zero-padded to
a 32-bit boundary, but the padding bits are not considered part of
the data field and are not included in the field count.

IP hosts are not required to reassemble datagrams larger than 576
octets; however, some commands or responses may involve more data
than will fit into a single datagram. Accordingly, a simple
reassembly feature is included in which each octet of the message
data is numbered starting with zero. As each fragment is transmitted
the number of its first octet is inserted in the offset field and the
number of octets is inserted in the count field. The more-data (M)
bit is set in all fragments except the last.

Most control functions involve sending a command and receiving a
response, perhaps involving several fragments. The sender chooses a
distinct, nonzero sequence number and sets the status field and R and
E bits to zero. The responder interprets the opcode and additional
information in the data field, updates the status field, sets the R
bit to one and returns the three 32-bit words of the header along
with additional information in the data field. In case of invalid
message format or contents the responder inserts a code in the status
field, sets the R and E bits to one and, optionally, inserts a
diagnostic message in the data field.

Some commands read or write system variables and peer variables for
an association identified in the command. Others read or write
variables associated with a radio clock or other device directly
connected to a source of primary synchronization information. To
identify which type of variable and association a 16-bit association
identifier is used. System variables are indicated by the identifier
zero. As each association is mobilized a unique, nonzero identifier
is created for it. These identifiers are used in a cyclic fashion,
so that the chance of using an old identifier which matches a newly
created association is remote. A management entity can request a
list of current identifiers and subsequently use them to read and
write variables for each association. An attempt to use an expired
identifier results in an exception response, following which the list
can be requested again.

Some exception events, such as when a peer becomes reachable or
unreachable, occur spontaneously and are not necessarily associated
with a command. An implementation may elect to save the event
information for later retrieval or to send an asynchronous response
(called a trap) or both. In case of a trap the IP address and port
number is determined by a previous command and the sequence field is
set as described below. Current status and summary information for
the latest exception event is returned in all normal responses. Bits
in the status field indicate whether an exception has occurred since
the last response and whether more than one exception has occurred.

Commands need not necessarily be sent by an NTP peer, so ordinary
access-control procedures may not apply; however, the optional mask/
match mechanism suggested in [RFC5905] provides the capability to
limit mode 6 processing to selected address ranges.

The Network Time Protocol reference implementation maintained by the
University of Delaware and ntp.org provides a utility program, ntpq
which enables management and configuration of the ntpd daemon using
NTP Control Messages (mode 6). A related utility program, ntpdc, uses
an earlier, deprecated implementation-specific binary management
protocol using NTP mode 7 datagrams. Due to the implementation
complexity of the earlier protocol, the reference implementation has
added support for all operations that previously were exposed only
via mode 7 to the preferred mode 6 interface. Support for mode 7
requests is likely to be disabled by default in the reference
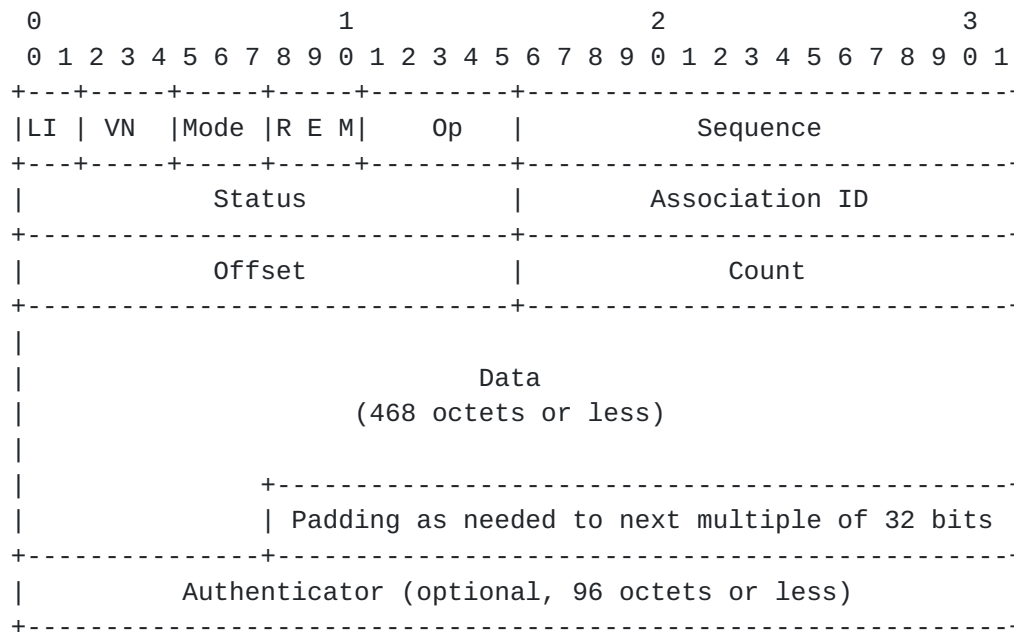implementation's daemon.

2. NTP Control Message Format

The format of the NTP Control Message header, which immediately
follows the UDP header, is shown below. Following is a description
of its fields. Bit positions marked as zero are reserved and should

always be transmitted as zero.

```
     0                   1                   2                   3
     0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
    +---+-----+-----+-----+---------+-------------------------------+
    |LI | VN  |Mode |R E M|   Op    |           Sequence            |
    +---+-----+-----+-----+---------+-------------------------------+
    |             Status            |         Association ID        |
    +-------------------------------+-------------------------------+
    |             Offset            |             Count             |
    +-------------------------------+-------------------------------+
    |                                                               |
    |                             Data                              |
    |                     (468 octets or less)                      |
    |                                                               |
    |               +-----------------------------------------------+
    |               | Padding as needed to next multiple of 32 bits |
    +---------------+-----------------------------------------------+
    |         Authenticator (optional, 96 octets or less)           |
    +---------------------------------------------------------------+
```

   LI: This is a two-bit integer that must be zero for control message
   requests and responses. The Leap Indicator value used at this
   position in most NTP modes is in the System Status Word provided in
   some control message responses.

   Version Number (VN): This is a three-bit integer indicating a minimum
   NTP version number. NTP servers should not respond to control
   messages with an unrecognized version number. Requests may
   intentionally use a lower version number to enable interoperability
   with earlier versions. The reference implementation utility ntpq
   uses version 2 by default. Responses must carry the same version as
   the corresponding request.

   Mode: This is a three-bit integer indicating the mode. It must have
   the value 6, indicating an NTP control message.

   Response Bit (R): Set to zero for commands, one for responses.

   Error Bit (E): Set to zero for normal response, one for error
   response.

   More Bit (M): Set to zero for last fragment, one for all others.

   Operation Code (Op): This is a five-bit integer specifying the
   command function. The values are:

```
+-------+--------------------------------------------------+
| Code  |                    Meaning                       |
+-------+--------------------------------------------------+
|   0   | reserved                                         |
|   1   | read status command/response                     |
|   2   | read variables command/response                  |
|   3   | write variables command/response                 |
|   4   | read clock variables command/response            |
|   5   | write clock variables command/response           |
|   6   | set trap address/port command/response           |
|   7   | trap response                                    |
|   8   | runtime configuration command/response           |
|   9   | export configuration to file command/response    |
|  10   | retrieve remote address stats command/response   |
|  11   | retrieve ordered list                            |
|  12   | request client-specific nonce command/response   |
| 13-30 | reserved for future use                          |
|  31   | unset trap address/port command/response         |
+-------+--------------------------------------------------+
```

Sequence: This is a 16-bit integer indicating the sequence number.
Each request should use a different sequence number. Each response
carries the same sequence number as its corresponding request. For
asynchronous trap responses, the responder increments the sequence
number by one each response, allowing trap receivers to detect
missing trap responses. Note the sequence number of each fragment in
a multiple-datagram response carries the same sequence number, copied
from the request.

Status: This is a 16-bit code indicating the current status of the
system, peer or clock, with values coded as described in following
sections.

Association ID: This is a 16-bit unsigned integer identifying a valid

association, or zero for the system clock.

Offset: This is a 16-bit unsigned integer indicating the offset, in
octets, of the first octet in the data area. The offset must be zero
in requests. Responses spanning multiple datagrams use a positive
offset in all but the first datagram.

Count: This is a 16-bit unsigned integer indicating the length of the
data, in octets

Data: This contains the message data for the command or response.
The maximum number of data octets is 468.

Padding: Contains zero to three octets with value zero, as needed to
ensure the overall control message size is a multiple of 4 octets.

Authenticator (optional): When an NTP authentication mechanism is
used, this contains the message authenticator information defined in
section 7.3 of [RFC5905].

3. Status Words

Status words indicate the present status of the system, associations
and clock. They are designed to be interpreted by network-monitoring
programs and are in one of four 16-bit formats shown in Figure 6 and
described in this section. System and peer status words are
associated with responses for all commands except the read clock
variables, write clock variables and set trap address/port commands.
The association identifier zero specifies the system status word,
while a nonzero identifier specifies a particular peer association.
The status word returned in response to read clock variables and
write clock variables commands indicates the state of the clock
hardware and decoding software. A special error status word is used
to report malformed command fields or invalid values.

3.1. System Status Word

The system status word appears in the status field of the response to
a read status or read variables command with a zero association
identifier. The format of the system status word is as follows:

```
 0                   1
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+---+-----------+-------+-------+
|LI | ClockSrc  | Count | Code  |
+---+-----------+-------+-------+
```

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, with bit 0 and bit 1, respectively, coded as follows: (EDITOR:
this could refer to [RFC 5905 section 7.3](#) figure 9 instead.)

```
+------+------------------------------------------------------------+
| LI   |                          Meaning                           |
+------+------------------------------------------------------------+
| 00   | no warning                                                 |
| 01   | insert second after 23:59:59 of the current day            |
| 10   | delete second 23:59:59 of the current day                  |
| 11   | unsynchronized                                             |
+------+------------------------------------------------------------+
```

ClockSrc: This is a six-bit integer indicating the current
synchronization source, with values coded as follows:

```
+-------+------------------------------------------------------------+
| Code  |                          Meaning                           |
+-------+------------------------------------------------------------+
|   0   | unspecified or unknown                                     |
|   1   | Calibrated atomic clock (e.g., PPS, HP 5061)               |
|   2   | VLF (band 4) or LF (band 5) radio (e.g., OMEGA, WWVB)      |
|   3   | HF (band 7) radio (e.g., CHU, MSF, WWV/H)                  |
|   4   | UHF (band 9) satellite (e.g., GOES, GPS)                   |
|   5   | local net (e.g., DCN, TSP, DTS)                            |
|   6   | UDP/NTP                                                    |
|   7   | UDP/TIME                                                   |
|   8   | eyeball-and-wristwatch                                     |
|   9   | telephone modem (e.g., NIST)                               |
| 10-63 | reserved                                                   |
+-------+------------------------------------------------------------+
```

System Event Counter: This is a four-bit integer indicating the
number of system events occurring since the last time the System
Event Code changed. Upon reaching 15, subsequent events with the
same code are not counted.

System Event Code: This is a four-bit integer identifying the latest
system exception event, with new values overwriting previous values,
and coded as follows:

```
+-------+------------------------------------------------------------+
| Code  |                          Meaning                           |
+-------+------------------------------------------------------------+
|   0   | unspecified                                                |
|   1   | frequency correction (drift) file not available            |
|   2   | frequency correction started (frequency stepped)           |
|   3   | spike detected and ignored, starting stepout timer         |
|   4   | frequency training started                                 |
|   5   | clock synchronized                                         |
|   6   | system restart                                             |
|   7   | panic stop (required step greater than panic threshold)    |
|   8   | no system peer                                             |
|   9   | leap second insertion/deletion armed for end of current month |
|  10   | leap second disarmed                                       |
|  11   | leap second inserted or deleted                            |
|  12   | clock stepped (stepout timer expired)                      |
|  13   | kernel loop discipline status changed                      |
|  14   | leapseconds table loaded from file                         |
|  15   | leapseconds table outdated, updated file needed            |
+-------+------------------------------------------------------------+
```

3.2. Peer Status Word

A peer status word is returned in the status field of a response to a
read status, read variables or write variables command and appears
also in the list of association identifiers and status words returned
by a read status command with a zero association identifier. The
format of a peer status word is as follows:

```
 0                   1
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+---------+-----+-------+-------+
|  Flags  | Sel | Count | Code  |
+---------+-----+-------+-------+
```

Peer Status Flags: This is a set of five bits indicating the status
of the peer determined by the packet procedure, with bits assigned as
follows:

```
+--------+--------+------------------------------------------------------+
| Peer   |        |                                                      |
| Status |        |                                                      |
| Flag   |        |                                                      |
| Bit    | Value  |                     Meaning                          |
+--------+--------+------------------------------------------------------+
```

```
|    0   | 0x8000 | configured (peer.config)                           |
|    1   | 0x4000 | authentication enabled (peer.authenable)           |
|    2   | 0x2000 | authentication okay (peer.authentic)               |
|    3   | 0x1000 | reachable (peer.reach != 0)                         |
|    4   | 0x0800 | broadcast association                              |
+--------+--------+----------------------------------------------------+
```

Peer Selection (Sel): This is a three-bit integer indicating the
status of the peer determined by the clock-selection procedure, with
values coded as follows:

```
+-------+-----------------------------------------------------------+
| Peer  |                                                           |
| Sel   |                      Meaning                              |
+-------+-----------------------------------------------------------+
|   0   | rejected                                                  |
|   1   | discarded by intersection algorithm                       |
|   2   | discarded by table overflow (not currently used)          |
|   3   | discarded by the cluster algorithm                        |
|   4   | included by the combine algorithm                         |
|   5   | backup source (with more than sys.maxclock survivors)     |
|   6   | system peer (synchronization source)                      |
|   7   | PPS (pulse per second) peer                               |
+-------+-----------------------------------------------------------+
```

Peer Event Counter: This is a four-bit integer indicating the number
of peer events that occurred since the last time the peer event code
changed. Upon reaching 15, subsequent events with the same code are
not counted.

Peer Event Code: This is a four-bit integer identifying the latest
peer exception event, with new values overwriting previous values,
and coded as follows:

```
   +-------
+------------------------------------------------------------------------+
   | Peer   |
|
   | Event |                                    Meaning
|
   | Code   |
|
   +-------
+------------------------------------------------------------------------+
   |    0  | unspecified
|
   |    1  | association mobilized
|
   |    2  | association demobilized
|
   |    3  | peer unreachable
|
   |    4  | peer reachable
|
   |    5  | association restarted or timed out
|
   |    6  | no reply (used only with one-shot ntpd -q, known as ntpdate mode)
|
   |    7  | peer rate limit exceeded (kiss code RATE received)
|
   |    8  | access denied (kiss code DENY received), not currently implemented
|
   |    9  | leap second insertion/deletion at month's end armed by peer vote
|
   |   10  | became system peer (sys.peer)
|
   |   11  | reference clock event (see clock status word)
|
   |   12  | authentication failed
|
   |   13  | popcorn spike suppressed by peer clock filter register
|
   |   14  | entering interleaved mode
|
   |   15  | recovered from interleave error
|
   +-------
+------------------------------------------------------------------------+
```

3.3. Clock Status Word

There are two ways a reference clock can be attached to a NTP service

host, as an dedicated device managed by the operating system and as a
synthetic peer managed by NTP. As in the read status command, the
association identifier is used to identify which one, zero for the
system clock and nonzero for a peer clock. Only one system clock is
supported by the protocol, although many peer clocks can be
supported. A system or peer clock status word appears in the status
field of the response to a read clock variables or write clock
variables command. This word can be considered an extension of the
system status word or the peer status word as appropriate. The
format of the clock status word is as follows:

```
 0                   1
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
+---------------+-------+-------+
|   Reserved    | Count | Code  |
+---------------+-------+-------+
```

Reserved: An eight-bit integer that should be ignored by requesters
and zeroed by responders.

Clock Event Counter: This is a four-bit integer indicating the number
of clock events that occurred since the last time the clock event
code changed. Upon reaching 15, subsequent events with the same code
are not counted.

Clock Event Code: This is a four-bit integer indicating the current
clock status, with values coded as follows:

```
+--------------+------------------------------------------------------+
| Clock Status |                     Meaning                          |
+--------------+------------------------------------------------------+
|       0      | clock operating within nominals                      |
|       1      | reply timeout                                        |
|       2      | bad reply format                                     |
|       3      | hardware or software fault                           |
|       4      | propagation failure (loss of signal)                 |
|       5      | bad date format or value                             |
|       6      | bad time format or value                             |
|     7-15     | reserved                                             |
+--------------+------------------------------------------------------+
```

[3.4]. Error Status Word

An error status word is returned in the status field of an error
response as the result of invalid message format or contents. Its
presence is indicated when the E (error) bit is set along with the
response (R) bit in the response. The format of the Error Status
Word is:

```
 0                       1
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5
 +---------------+---------------+
 |  Error Code   |   Reserved    |
 +---------------+---------------+
```

Error code: an eight-bit integer coded as follows:

```
+--------------+------------------------------------------------------+
| Error Status |                     Meaning                          |
+--------------+------------------------------------------------------+
|       0      | unspecified                                          |
|       1      | authentication failure                               |
|       2      | invalid message length or format                     |
|       3      | invalid opcode                                       |
|       4      | unknown association identifier                       |
|       5      | unknown variable name                                |
|       6      | invalid variable value                               |
|       7      | administratively prohibited                          |
|     8-255    | reserved                                             |
+--------------+------------------------------------------------------+
```

Reserved: Responders should use zero. Requesters should ignore the
Reserved value to preserve the possibility of future use.

[4]. Commands

Commands consist of the header and optional data field shown in
Section 3. When present, the data field contains a list of
identifiers or assignments in the form
<<identifier>>[=<<value>>],<<identifier>>[=<<value>>],... where
<<identifier>> is the ASCII name of a system or peer variable
specified in Sections 9.1 and 11.1 of RFC 5905 and <<value>> is
expressed as a decimal, hexadecimal or string constant in the syntax
of the C programming language. Where no ambiguity exists, the "s."
or "p." prefixes shown in Figure 5 of Section 7.1 of RFC 5905
[RFC5905] can be suppressed. Whitespace (ASCII nonprinting format
effectors) can be added to improve readability for simple monitoring
programs that do not reformat the data field. Internet Protocol
version 4 addresses are represented as four decimal octets without
leading zeros, separated by dots. Internet Protocol version 6
addresses are represented as mandated by [RFC5952], without
surrounding square brackets unless a port specification is combined
with the address. Timestamps, including reference, originate,
receive and transmit values, as well as the logical clock, are
represented in units of seconds and fractions, preferably in
hexadecimal notation, while delay, offset, dispersion and distance
values are represented in units of milliseconds and fractions,
preferably in decimal notation. All other values are represented as-
is, preferably in decimal notation.

Implementations may define variables other than those listed in
Figures 6, 7, 16, 17, 18, 19, 27 and 29 of RFC 5905. Called
extramural variables, these are distinguished by the inclusion of
some character type other than alphanumeric or "." in the name. For
those commands that return a list of assignments in the response data
field, if the command data field is empty, it is expected that all
available variables defined in Figures 6, 7 and 17 of RFC 5905 will
be included in the response. For the read commands, if the command
data field is nonempty, an implementation may choose to process this
field to individually select which variables are to be returned.

Commands are interpreted as follows:

Read Status (1): The command data field is empty or contains a list
of identifiers separated by commas. The command operates in two ways
depending on the value of the association identifier. If this
identifier is nonzero, the response includes the peer identifier and
status word. Optionally, the response data field may contain other
information, such as described in the Read Variables command. If the
association identifier is zero, the response includes the system
identifier (0) and status word, while the data field contains a list
of binary-coded pairs <<association identifier>> <<status word>>, one
for each currently defined association.

Read Variables (2): The command data field is empty or contains a
list of identifiers separated by commas. If the association
identifier is nonzero, the response includes the requested peer
identifier and status word, while the data field contains a list of
peer variables and values as described above. If the association
identifier is zero, the data field contains a list of system
variables and values. If a peer has been selected as the
synchronization source, the response includes the peer identifier and
status word; otherwise, the response includes the system identifier
(0) and status word.

Write Variables (3): The command data field contains a list of
assignments as described above. The variables are updated as
indicated. The response is as described for the Read Variables
command.

Read Clock Variables (4): The command data field is empty or contains
a list of identifiers separated by commas. The association
identifier selects the system clock variables or peer clock variables
in the same way as in the Read Variables command. The response
includes the requested clock identifier and status word and the data
field contains a list of clock variables and values, including the
last timecode message received from the clock.

Write Clock Variables (5): The command data field contains a list of
assignments as described above. The clock variables are updated as
indicated. The response is as described for the Read Clock Variables
command. The reference implementation daemon requires authentication
for this command.

Set Trap Address/Port (6): The command association identifier, status
and data fields are ignored. The address and port number for
subsequent trap messages are taken from the source address and port
of the control message itself. The initial trap counter for trap
response messages is taken from the sequence field of the command.
The response association identifier, status and data fields are not
significant. Implementations should include sanity timeouts which
prevent trap transmissions if the monitoring program does not renew
this information after a lengthy interval.

Trap Response (7): This command differs from the others described
here, which are initiated by a management agent (such as ntpq) and
responded to by a NTP daemon. Trap Response is sent by a NTP daemon
to any registered trap receivers when a system, peer or clock
exception event occurs. The opcode field is 7 and the R bit is set.
The trap counter is incremented by one for each trap sent and the
sequence field set to that value. The trap message is sent using the
IP address and port fields established by the set trap address/port

command. If a system trap the association identifier field is set to
zero and the status field contains the system status word. If a peer
trap the association identifier field is set to that peer and the
status field contains the peer status word. Optional ASCII-coded
information can be included in the data field.

Configure (8): The command data is parsed and applied as if supplied
in the daemon configuration file. The reference implementation
daemon requires authentication for this command.

Save Configuration (9): Write a snapshot of the current configuration
to the file name supplied as the command data. The reference
implementation daemon requires authentication for this command.
Further, the command is refused unless a directory in which to store
the resulting files has been explicitly configured by the operator.

Read MRU (10): Retrieves records of recently seen remote addresses
and associated statistics. Command data consists of name=value pairs
controlling the selection of records, as well as a requestor-specific
nonce previously retrieved using this command or opcode 12, Request
Nonce. The response consists of name=value pairs where some names
can appear multiple times using a dot followed by a zero-based index
to distinguish them, and to associate elements of the same record
with the same index. A new nonce is provided with each successful
response.

Read ordered list (11): Retrieves an ordered list. If the command
data is empty or the seven characters "ifstats" the associated
statistics, status and counters for each local address are returned.
If the command data is the characters "addr_restrictions" then the
set of IPv4 remote address restrictions followed by the set of IPv6
remote address restrictions (access control lists) are returned.
Other command data returns error code 5 (unknown variable name).
Similar to Read MRU, response information uses zero-based indexes as
part of the variable name preceding the equals sign and value, where
each index relates information for a single address or network. This
opcode requires authentication.

Request Nonce (12): Retrieves a 96-bit nonce specific to the
requesting remote address, which is valid for a limited period.
Command data is not used in the request. The nonce consists of a
64-bit NTP timestamp and 32 bits of hash derived from that timestamp,
the remote address, and salt known only to the server which varies
between daemon runs. The reference implementation honors nonces
which were issued less than 16 seconds prior. Regurgitation of the
nonce by a managment agent demonstrates to the server that the agent
can receive datagrams sent to the source address of the request,
making source address "spoofing" more difficult in a similar way as
TCP's three-way handshake.

Unset Trap (31): Removes the requesting remote address and port from
the list of trap receivers. Command data is not used in the request.
If the address and port are not in the list of trap receivers, the
error code is 4, bad association.

[5](#). **IANA Considerations**

   Editor's Note: To be reviewed by the working group prior to
   completion.

6. Security Considerations

   Editor's Note: To be supplied by the working group prior to
   completion.

7. Acknowledgements

8. References

8.1. Normative References

   [RFC5905]   Mills, D., Martin, J., Burbank, J. and W. Kasch, "Network
               Time Protocol Version 4: Protocol and Algorithms
               Specification", RFC 5905, June 2010.

   [RFC5952]   Kawamura, S. and M. Kawashima, "A Recommendation for IPv6
               Address Text Representation", RFC 5952, August 2010.

8.2. Informative References

   [RFC5906]   Haberman, B. and D. Mills, "Network Time Protocol Version
               4: Autokey Specification", RFC 5906, June 2010.

   [RFC5907]   Gerstung, H., Elliott, C. and B. Haberman, "Definitions of
               Managed Objects for Network Time Protocol Version 4
               (NTPv4)", RFC 5907, June 2010.

   [RFC5908]   Gayraud, R. and B. Lourdelet, "Network Time Protocol (NTP)
               Server Option for DHCPv6", RFC 5908, June 2010.

Authors' Addresses

   Dr. David L. Mills
   University of Delaware
   Newark, Delaware
   US

   Email: mills@udel.edu

   Karen O'Donoghue, editor
   Internet Society
   King George, Virginia
   US

   Email: odonoghue@isoc.org

   David L. Hart
   Redmond, Washington

US

Email: hart@ntp.org

Harlan M. Stenn
Network Time Foundation, Inc.
Talent, Oregon
US

Email: stenn@ntp.org

Philip F. Chimento, editor
Johns Hopkins University/APL
11100 Johns Hopkins Road
Laurel, Maryland 20723
US

Email: Philip.Chimento@jhuapl.edu