Network Working Group
Request for Comments: 977

Brian Kantor (U.C. San Diego)
Phil Lapsley (U.C. Berkeley)
February 1986

Network News Transfer Protocol

A Proposed Standard for the Stream-Based Transmission of News

Status of This Memo

NNTP specifies a protocol for the distribution, inquiry, retrieval, and posting of news articles using a reliable stream-based transmission of news among the ARPA-Internet community. NNTP is designed so that news articles are stored in a central database allowing a subscriber to select only those items he wishes to read. Indexing, cross-referencing, and expiration of aged messages are also provided. This RFC suggests a proposed protocol for the ARPA-Internet community, and requests discussion and suggestions for improvements. Distribution of this memo is unlimited.

1. Introduction

For many years, the ARPA-Internet community has supported the distribution of bulletins, information, and data in a timely fashion to thousands of participants. We collectively refer to such items of information as "news". Such news provides for the rapid dissemination of items of interest such as software bug fixes, new product reviews, technical tips, and programming pointers, as well as rapid-fire discussions of matters of concern to the working computer professional. News is very popular among its readers.

There are popularly two methods of distributing such news: the Internet method of direct mailing, and the USENET news system.

1.1. Internet Mailing Lists

The Internet community distributes news by the use of mailing lists. These are lists of subscriber's mailbox addresses and remailing sublists of all intended recipients. These mailing lists operate by remailing a copy of the information to be distributed to each subscriber on the mailing list. Such remailing is inefficient when a mailing list grows beyond a dozen or so people, since sending a separate copy to each of the subscribers occupies large quantities of network bandwidth, CPU resources, and significant amounts of disk storage at the destination host. There is also a significant problem in maintenance of the list itself: as subscribers move from one job to another; as new subscribers join and old ones leave; and as hosts come in and out of service.

1.2. The USENET News System

Clearly, a worthwhile reduction of the amount of these resources used can be achieved if articles are stored in a central database on the receiving host instead of in each subscriber's mailbox. The USENET news system provides a method of doing just this. There is a central repository of the news articles in one place (customarily a spool directory of some sort), and a set of programs that allow a subscriber to select those items he wishes to read. Indexing, cross-referencing, and expiration of aged messages are also provided.

1.3. Central Storage of News

For clusters of hosts connected together by fast local area networks (such as Ethernet), it makes even more sense to consolidate news distribution onto one (or a very few) hosts, and to allow access to these news articles using a server and client model. Subscribers may then request only the articles they wish to see, without having to wastefully duplicate the storage of a copy of each item on each host.

1.4. A Central News Server

A way to achieve these economies is to have a central computer system that can provide news service to the other systems on the local area network. Such a server would manage the collection of news articles and index files, with each person who desires to read news bulletins doing so over the LAN. For a large cluster of computer systems, the savings in total disk space is clearly worthwhile. Also, this allows workstations with limited disk storage space to participate in the news without incoming items consuming oppressive amounts of the workstation's disk storage.

We have heard rumors of somewhat successful attempts to provide centralized news service using IBIS and other shared or distributed file systems. While it is possible that such a distributed file system implementation might work well with a group of similar computers running nearly identical operating systems, such a scheme is not general enough to offer service to a wide range of client systems, especially when many diverse operating systems may be in use among a group of clients. There are few (if any) shared or networked file systems that can offer the generality of service that stream connections using Internet TCP provide, particularly when a wide range of host hardware and operating systems are considered.

NNTP specifies a protocol for the distribution, inquiry, retrieval, and posting of news articles using a reliable stream (such as TCP) server-client model. NNTP is designed so that news articles need only

be stored on one (presumably central) host, and subscribers on other hosts attached to the LAN may read news articles using stream connections to the news host.

NNTP is modelled upon the news article specifications in RFC 850, which describes the USENET news system. However, NNTP makes few demands upon the structure, content, or storage of news articles, and thus we believe it easily can be adapted to other non-USENET news systems.

Typically, the NNTP server runs as a background process on one host, and would accept connections from other hosts on the LAN. This works well when there are a number of small computer systems (such as workstations, with only one or at most a few users each), and a large central server.

1.5. Intermediate News Servers

For clusters of machines with many users (as might be the case in a university or large industrial environment), an intermediate server might be used. This intermediate or "slave" server runs on each computer system, and is responsible for mediating news reading requests and performing local caching of recently-retrieved news articles.

Typically, a client attempting to obtain news service would first attempt to connect to the news service port on the local machine. If this attempt were unsuccessful, indicating a failed server, an installation might choose to either deny news access, or to permit connection to the central "master" news server.

For workstations or other small systems, direct connection to the master server would probably be the normal manner of operation.

This specification does not cover the operation of slave NNTP servers. We merely suggest that slave servers are a logical addition to NNTP server usage which would enhance operation on large local area networks.

1.6. News Distribution

NNTP has commands which provide a straightforward method of exchanging articles between cooperating hosts. Hosts which are well connected on a local area or other fast network and who wish to actually obtain copies of news articles for local storage might well find NNTP to be a more efficient way to distribute news than more traditional transfer methods (such as UUCP).

In the traditional method of distributing news articles, news is propagated from host to host by flooding - that is, each host will send all its new news articles on to each host that it feeds. These hosts will then in turn send these new articles on to other hosts that they feed. Clearly, sending articles that a host already has obtained a copy of from another feed (many hosts that receive news are redundantly fed) again is a waste of time and communications resources, but for transport mechanisms that are single-transaction based rather than interactive (such as UUCP in the UNIX-world <1>), distribution time is diminished by sending all articles and having the receiving host simply discard the duplicates. This is an especially true when communications sessions are limited to once a day.

Using NNTP, hosts exchanging news articles have an interactive mechanism for deciding which articles are to be transmitted. A host desiring new news, or which has new news to send, will typically contact one or more of its neighbors using NNTP. First it will inquire if any new news groups have been created on the serving host by means of the NEWGROUPS command. If so, and those are appropriate or desired (as established by local site-dependent rules), those new newsgroups can be created.

The client host will then inquire as to which new articles have arrived in all or some of the newsgroups that it desires to receive, using the NEWNEWS command. It will receive a list of new articles from the server, and can request transmission of those articles that it desires and does not already have.

Finally, the client can advise the server of those new articles which the client has recently received. The server will indicate those articles that it has already obtained copies of, and which articles should be sent to add to its collection.

In this manner, only those articles which are not duplicates and which are desired are transferred.

2. The NNTP Specification

2.1. Overview

The news server specified by this document uses a stream connection (such as TCP) and SMTP-like commands and responses. It is designed to accept connections from hosts, and to provide a simple interface to the news database.

This server is only an interface between programs and the news databases. It does not perform any user interaction or presentation-level functions. These "user-friendly" functions are better left to the client programs, which have a better understanding of the environment in which they are operating.

When used via Internet TCP, the contact port assigned for this service is 119.

2.2. Character Codes

Commands and replies are composed of characters from the ASCII character set. When the transport service provides an 8-bit byte (octet) transmission channel, each 7-bit character is transmitted right justified in an octet with the high order bit cleared to zero.

2.3. Commands

Commands consist of a command word, which in some cases may be followed by a parameter. Commands with parameters must separate the parameters from each other and from the command by one or more space or tab characters. Command lines must be complete with all required parameters, and may not contain more than one command.

Commands and command parameters are not case sensitive. That is, a command or parameter word may be upper case, lower case, or any mixture of upper and lower case.

Each command line must be terminated by a CR-LF (Carriage Return - Line Feed) pair.

Command lines shall not exceed 512 characters in length, counting all characters including spaces, separators, punctuation, and the trailing CR-LF (thus there are 510 characters maximum allowed for the command and its parameters). There is no provision for continuation command lines.

2.4. Responses

Responses are of two kinds, textual and status.

2.4.1. Text Responses

Text is sent only after a numeric status response line has been sent that indicates that text will follow. Text is sent as a series of successive lines of textual matter, each terminated with CR-LF pair. A single line containing only a period (.) is sent to indicate the end of the text (i.e., the server will send a CR-LF pair at the end of the last line of text, a period, and another CR-LF pair).

If the text contained a period as the first character of the text line in the original, that first period is doubled. Therefore, the client must examine the first character of each line received, and for those beginning with a period, determine either that this is the end of the text or whether to collapse the doubled period to a single one.

The intention is that text messages will usually be displayed on the user's terminal whereas command/status responses will be interpreted by the client program before any possible display is done.

2.4.2. Status Responses

These are status reports from the server and indicate the response to the last command received from the client.

Status response lines begin with a 3 digit numeric code which is sufficient to distinguish all responses. Some of these may herald the subsequent transmission of text.

The first digit of the response broadly indicates the success, failure, or progress of the previous command.

```
1xx - Informative message
2xx - Command ok
3xx - Command ok so far, send the rest of it.
4xx - Command was correct, but couldn't be performed for
      some reason.
5xx - Command unimplemented, or incorrect, or a serious
      program error occurred.
```

The next digit in the code indicates the function response category.

- x0x - Connection, setup, and miscellaneous messages
- x1x - Newsgroup selection
- x2x - Article selection
- x3x - Distribution functions
- x4x - Posting
- x8x - Nonstandard (private implementation) extensions
- x9x - Debugging output

The exact response codes that should be expected from each command are detailed in the description of that command. In addition, below is listed a general set of response codes that may be received at any time.

Certain status responses contain parameters such as numbers and names. The number and type of such parameters is fixed for each response code to simplify interpretation of the response.

Parameters are separated from the numeric response code and from each other by a single space. All numeric parameters are decimal, and may have leading zeros. All string parameters begin after the separating space, and end before the following separating space or the CR-LF pair at the end of the line. (String parameters may not, therefore, contain spaces.) All text, if any, in the response which is not a parameter of the response must follow and be separated from the last parameter by a space. Also, note that the text following a response number may vary in different implementations of the server. The 3-digit numeric code should be used to determine what response was sent.

Response codes not specified in this standard may be used for any installation-specific additional commands also not specified. These should be chosen to fit the pattern of x8x specified above. (Note that debugging is provided for explicitly in the x9x response codes.) The use of unspecified response codes for standard commands is prohibited.

We have provided a response pattern x9x for debugging. Since much debugging output may be classed as "informative messages", we would expect, therefore, that responses 190 through 199 would be used for various debugging outputs. There is no requirement in this specification for debugging output, but if such is provided over the connected stream, it must use these response codes. If appropriate to a specific implementation, other x9x codes may be used for debugging. (An example might be to use e.g., 290 to acknowledge a remote debugging request.)

2.4.3. General Responses

The following is a list of general response codes that may be sent by the NNTP server. These are not specific to any one command, but may be returned as the result of a connection, a failure, or some unusual condition.

In general, 1xx codes may be ignored or displayed as desired; code 200 or 201 is sent upon initial connection to the NNTP server depending upon posting permission; code 400 will be sent when the NNTP server discontinues service (by operator request, for example); and 5xx codes indicate that the command could not be performed for some unusual reason.

```
100 help text
190
  through
199 debug output

200 server ready - posting allowed
201 server ready - no posting allowed

400 service discontinued

500 command not recognized
501 command syntax error
502 access restriction or permission denied
503 program fault - command not performed
```

3. Command and Response Details

On the following pages are descriptions of each command recognized by the NNTP server and the responses which will be returned by those commands.

Each command is shown in upper case for clarity, although case is ignored in the interpretation of commands by the NNTP server. Any parameters are shown in lower case. A parameter shown in [square brackets] is optional. For example, [GMT] indicates that the triglyph GMT may present or omitted.

Every command described in this section must be implemented by all NNTP servers.

There is no prohibition against additional commands being added; however, it is recommended that any such unspecified command begin with the letter "X" to avoid conflict with later revisions of this specification.

Implementors are reminded that such additional commands may not redefine specified status response codes. Using additional unspecified responses for standard commands is also prohibited.

3.1. The ARTICLE, BODY, HEAD, and STAT commands

There are two forms to the ARTICLE command (and the related BODY, HEAD, and STAT commands), each using a different method of specifying which article is to be retrieved. When the ARTICLE command is followed by a message-id in angle brackets ("<" and ">"), the first form of the command is used; when a numeric parameter or no parameter is supplied, the second form is invoked.

The text of the article is returned as a textual response, as described earlier in this document.

The HEAD and BODY commands are identical to the ARTICLE command except that they respectively return only the header lines or text body of the article.

The STAT command is similar to the ARTICLE command except that no text is returned. When selecting by message number within a group, the STAT command serves to set the current article pointer without sending text. The returned acknowledgement response will contain the message-id, which may be of some value. Using the STAT command to select by message-id is valid but of questionable value, since a selection by message-id does NOT alter the "current article pointer".

3.1.1. ARTICLE (selection by message-id)

ARTICLE <message-id>

Display the header, a blank line, then the body (text) of the specified article. Message-id is the message id of an article as shown in that article's header. It is anticipated that the client will obtain the message-id from a list provided by the NEWNEWS command, from references contained within another article, or from the message-id provided in the response to some other commands.

Please note that the internally-maintained "current article pointer" is NOT ALTERED by this command. This is both to facilitate the presentation of articles that may be referenced within an article

being read, and because of the semantic difficulties of determining the proper sequence and membership of an article which may have been posted to more than one newsgroup.

3.1.2. ARTICLE (selection by number)

ARTICLE [nnn]

Displays the header, a blank line, then the body (text) of the current or specified article. The optional parameter nnn is the

numeric id of an article in the current newsgroup and must be chosen from the range of articles provided when the newsgroup was selected. If it is omitted, the current article is assumed.

The internally-maintained "current article pointer" is set by this command if a valid article number is specified.

[the following applies to both forms of the article command.] A response indicating the current article number, a message-id string, and that text is to follow will be returned.

The message-id string returned is an identification string contained within angle brackets ("<" and ">"), which is derived from the header of the article itself. The Message-ID header line (required by RFC850) from the article must be used to supply this information. If the message-id header line is missing from the article, a single digit "0" (zero) should be supplied within the angle brackets.

Since the message-id field is unique with each article, it may be used by a news reading program to skip duplicate displays of articles that have been posted more than once, or to more than one newsgroup.

3.1.3. Responses

220 n <a> article retrieved - head and body follow
(n = article number, <a> = message-id)
221 n <a> article retrieved - head follows
222 n <a> article retrieved - body follows
223 n <a> article retrieved - request text separately
412 no newsgroup has been selected
420 no current article has been selected
423 no such article number in this group
430 no such article found

3.2. The GROUP command

3.2.1. GROUP

GROUP ggg

The required parameter ggg is the name of the newsgroup to be selected (e.g. "net.news"). A list of valid newsgroups may be obtained from the LIST command.

The successful selection response will return the article numbers of the first and last articles in the group, and an estimate of the number of articles on file in the group. It is not necessary that the estimate be correct, although that is helpful; it must only be equal to or larger than the actual number of articles on file. (Some implementations will actually count the number of articles on file. Others will just subtract first article number from last to get an estimate.)

When a valid group is selected by means of this command, the internally maintained "current article pointer" is set to the first article in the group. If an invalid group is specified, the previously selected group and article remain selected. If an empty newsgroup is selected, the "current article pointer" is in an indeterminate state and should not be used.

Note that the name of the newsgroup is not case-dependent. It must otherwise match a newsgroup obtained from the LIST command or an error will result.

3.2.2. Responses

```
211 n f l s group selected
        (n = estimated number of articles in group,
        f = first article number in the group,
        l = last article number in the group,
        s = name of the group.)
411 no such news group
```

3.3. The HELP command

3.3.1. HELP

HELP

Provides a short summary of commands that are understood by this implementation of the server. The help text will be presented as a textual response, terminated by a single period on a line by itself.

3.3.2. Responses

100 help text follows

3.4. The IHAVE command

3.4.1. IHAVE

IHAVE <messageid>

The IHAVE command informs the server that the client has an article whose id is <messageid>. If the server desires a copy of that article, it will return a response instructing the client to send the entire article. If the server does not want the article (if, for example, the server already has a copy of it), a response indicating that the article is not wanted will be returned.

If transmission of the article is requested, the client should send the entire article, including header and body, in the manner specified for text transmission from the server. A response code indicating success or failure of the transferral of the article will be returned.

This function differs from the POST command in that it is intended for use in transferring already-posted articles between hosts. Normally it will not be used when the client is a personal newsreading program. In particular, this function will invoke the server's news posting program with the appropriate settings (flags, options, etc) to indicate that the forthcoming article is being forwarded from another host.

The server may, however, elect not to post or forward the article if after further examination of the article it deems it inappropriate to do so. The 436 or 437 error codes may be returned as appropriate to the situation.

Reasons for such subsequent rejection of an article may include such

problems as inappropriate newsgroups or distributions, disk space limitations, article lengths, garbled headers, and the like. These are typically restrictions enforced by the server host's news software and not necessarily the NNTP server itself.

3.4.2. Responses

235 article transferred ok
335 send article to be transferred. End with <CR-LF>.<CR-LF>
435 article not wanted - do not send it
436 transfer failed - try again later
437 article rejected - do not try again

An implementation note:

Because some host news posting software may not be able to decide immediately that an article is inappropriate for posting or forwarding, it is acceptable to acknowledge the successful transfer of the article and to later silently discard it. Thus it is permitted to return the 235 acknowledgement code and later discard the received article. This is not a fully satisfactory solution to the problem. Perhaps some implementations will wish to send mail to the author of the article in certain of these cases.

3.5. The LAST command

3.5.1. LAST

LAST

The internally maintained "current article pointer" is set to the previous article in the current newsgroup. If already positioned at the first article of the newsgroup, an error message is returned and the current article remains selected.

The internally-maintained "current article pointer" is set by this command.

A response indicating the current article number, and a message-id string will be returned. No text is sent in response to this command.

3.5.2. Responses

223 n a article retrieved - request text separately
(n = article number, a = unique article id)

412 no newsgroup selected
420 no current article has been selected
422 no previous article in this group

3.6. The LIST command

3.6.1. LIST

LIST

Returns a list of valid newsgroups and associated information. Each newsgroup is sent as a line of text in the following format:

group last first p

where <group> is the name of the newsgroup, <last> is the number of the last known article currently in that newsgroup, <first> is the number of the first article currently in the newsgroup, and <p> is either 'y' or 'n' indicating whether posting to this newsgroup is allowed ('y') or prohibited ('n').

The <first> and <last> fields will always be numeric. They may have leading zeros. If the <last> field evaluates to less than the <first> field, there are no articles currently on file in the newsgroup.

Note that posting may still be prohibited to a client even though the LIST command indicates that posting is permitted to a particular newsgroup. See the POST command for an explanation of client prohibitions. The posting flag exists for each newsgroup because some newsgroups are moderated or are digests, and therefore cannot be posted to; that is, articles posted to them must be mailed to a moderator who will post them for the submitter. This is independent of the posting permission granted to a client by the NNTP server.

Please note that an empty list (i.e., the text body returned by this command consists only of the terminating period) is a possible valid response, and indicates that there are currently no valid newsgroups.

3.6.2. Responses

215 list of newsgroups follows

3.7. The NEWGROUPS command

3.7.1. NEWGROUPS

NEWGROUPS date time [GMT] [<distributions>]

A list of newsgroups created since <date and time> will be listed in the same format as the LIST command.

The date is sent as 6 digits in the format YYMMDD, where YY is the last two digits of the year, MM is the two digits of the month (with leading zero, if appropriate), and DD is the day of the month (with leading zero, if appropriate). The closest century is assumed as part of the year (i.e., 86 specifies 1986, 30 specifies 2030, 99 is 1999, 00 is 2000).

Time must also be specified. It must be as 6 digits HHMMSS with HH being hours on the 24-hour clock, MM minutes 00-59, and SS seconds 00-59. The time is assumed to be in the server's timezone unless the token "GMT" appears, in which case both time and date are evaluated at the 0 meridian.

The optional parameter "distributions" is a list of distribution groups, enclosed in angle brackets. If specified, the distribution portion of a new newsgroup (e.g, 'net' in 'net.wombat') will be examined for a match with the distribution categories listed, and only those new newsgroups which match will be listed. If more than one distribution group is to be listed, they must be separated by commas within the angle brackets.

Please note that an empty list (i.e., the text body returned by this command consists only of the terminating period) is a possible valid response, and indicates that there are currently no new newsgroups.

3.7.2. Responses

231 list of new newsgroups follows

3.8. The NEWNEWS command

3.8.1. NEWNEWS

NEWNEWS newsgroups date time [GMT] [<distribution>]

A list of message-ids of articles posted or received to the specified newsgroup since "date" will be listed. The format of the listing will be one message-id per line, as though text were being sent. A single line consisting solely of one period followed by CR-LF will terminate the list.

Date and time are in the same format as the NEWGROUPS command.

A newsgroup name containing a "*" (an asterisk) may be specified to broaden the article search to some or all newsgroups. The asterisk will be extended to match any part of a newsgroup name (e.g., net.micro* will match net.micro.wombat, net.micro.apple, etc). Thus if only an asterisk is given as the newsgroup name, all newsgroups will be searched for new news.

(Please note that the asterisk "*" expansion is a general replacement; in particular, the specification of e.g., net.*.unix should be correctly expanded to embrace names such as net.wombat.unix and net.whocares.unix.)

Conversely, if no asterisk appears in a given newsgroup name, only the specified newsgroup will be searched for new articles. Newsgroup names must be chosen from those returned in the listing of available groups. Multiple newsgroup names (including a "*") may be specified in this command, separated by a comma. No comma shall appear after the last newsgroup in the list. [Implementors are cautioned to keep the 512 character command length limit in mind.]

The exclamation point ("!") may be used to negate a match. This can be used to selectively omit certain newsgroups from an otherwise larger list. For example, a newsgroups specification of "net.*,mod.*,!mod.map.*" would specify that all net.<anything> and all mod.<anything> EXCEPT mod.map.<anything> newsgroup names would be matched. If used, the exclamation point must appear as the first character of the given newsgroup name or pattern.

The optional parameter "distributions" is a list of distribution groups, enclosed in angle brackets. If specified, the distribution portion of an article's newsgroup (e.g, 'net' in 'net.wombat') will be examined for a match with the distribution categories listed, and only those articles which have at least one newsgroup belonging to

the list of distributions will be listed. If more than one distribution group is to be supplied, they must be separated by commas within the angle brackets.

The use of the IHAVE, NEWNEWS, and NEWGROUPS commands to distribute news is discussed in an earlier part of this document.

Please note that an empty list (i.e., the text body returned by this command consists only of the terminating period) is a possible valid response, and indicates that there is currently no new news.

3.8.2. Responses

230 list of new articles by message-id follows

3.9. The NEXT command

3.9.1. NEXT

NEXT

The internally maintained "current article pointer" is advanced to the next article in the current newsgroup. If no more articles remain in the current group, an error message is returned and the current article remains selected.

The internally-maintained "current article pointer" is set by this command.

A response indicating the current article number, and the message-id string will be returned. No text is sent in response to this command.

3.9.2. Responses

223 n a article retrieved - request text separately
(n = article number, a = unique article id)
412 no newsgroup selected
420 no current article has been selected
421 no next article in this group

3.10. The POST command

3.10.1. POST

POST

If posting is allowed, response code 340 is returned to indicate that the article to be posted should be sent. Response code 440 indicates that posting is prohibited for some installation-dependent reason.

If posting is permitted, the article should be presented in the format specified by RFC850, and should include all required header lines. After the article's header and body have been completely sent by the client to the server, a further response code will be returned to indicate success or failure of the posting attempt.

The text forming the header and body of the message to be posted should be sent by the client using the conventions for text received from the news server: A single period (".") on a line indicates the end of the text, with lines starting with a period in the original text having that period doubled during transmission.

No attempt shall be made by the server to filter characters, fold or limit lines, or otherwise process incoming text. It is our intent that the server just pass the incoming message to be posted to the server installation's news posting software, which is separate from this specification. See RFC850 for more details.

Since most installations will want the client news program to allow the user to prepare his message using some sort of text editor, and transmit it to the server for posting only after it is composed, the client program should take note of the herald message that greeted it when the connection was first established. This message indicates whether postings from that client are permitted or not, and can be used to caution the user that his access is read-only if that is the case. This will prevent the user from wasting a good deal of time composing a message only to find posting of the message was denied. The method and determination of which clients and hosts may post is installation dependent and is not covered by this specification.

3.10.2. Responses

240 article posted ok
340 send article to be posted. End with <CR-LF>.<CR-LF>
440 posting not allowed
441 posting failed

(for reference, one of the following codes will be sent upon initial connection; the client program should determine whether posting is generally permitted from these:) 200 server ready - posting allowed 201 server ready - no posting allowed

3.11. The QUIT command

3.11.1. QUIT

QUIT

The server process acknowledges the QUIT command and then closes the connection to the client. This is the preferred method for a client to indicate that it has finished all its transactions with the NNTP server.

If a client simply disconnects (or the connection times out, or some other fault occurs), the server should gracefully cease its attempts to service the client.

3.11.2. Responses

205 closing connection - goodbye!

3.12. The SLAVE command

3.12.1. SLAVE

SLAVE

Indicates to the server that this client connection is to a slave server, rather than a user.

This command is intended for use in separating connections to single users from those to subsidiary ("slave") servers. It may be used to indicate that priority should therefore be given to requests from this client, as it is presumably serving more than one person. It might also be used to determine which connections to close when system load levels are exceeded, perhaps giving preference to slave servers. The actual use this command is put to is entirely implementation dependent, and may vary from one host to another. In NNTP servers which do not give priority to slave servers, this command must nonetheless be recognized and acknowledged.

3.12.2. Responses

202 slave status noted

4. Sample Conversations

These are samples of the conversations that might be expected with the news server in hypothetical sessions. The notation C: indicates commands sent to the news server from the client program; S: indicate responses received from the server by the client.

4.1. Example 1 - relative access with NEXT

```
S:      (listens at TCP port 119)

C:      (requests connection on TCP port 119)
S:      200 wombatvax news server ready - posting ok

(client asks for a current newsgroup list)
C:      LIST
S:      215 list of newsgroups follows
S:      net.wombats 00543 00501 y
S:      net.unix-wizards 10125 10011 y
        (more information here)
S:      net.idiots 00100 00001 n
S:      .

(client selects a newsgroup)
C:      GROUP net.unix-wizards
S:      211 104 10011 10125 net.unix-wizards group selected
        (there are 104 articles on file, from 10011 to 10125)

(client selects an article to read)
C:      STAT 10110
S:      223 10110 <23445@sdcsvax.ARPA> article retrieved - statistics
        only (article 10110 selected, its message-id is
        <23445@sdcsvax.ARPA>)

(client examines the header)
C:      HEAD
S:      221 10110 <23445@sdcsvax.ARPA> article retrieved - head
        follows (text of the header appears here)
S:      .

(client wants to see the text body of the article)
C:      BODY
S:      222 10110 <23445@sdcsvax.ARPA> article retrieved - body
        follows (body text here)
S:      .

(client selects next article in group)
```

```
   C:      NEXT
   S:      223 10113 <21495@nudebch.uucp> article retrieved - statistics
           only (article 10113 was next in group)

   (client finishes session)
   C:      QUIT
   S:      205 goodbye.
```

4.2. Example 2 - absolute article access with ARTICLE

```
   S:      (listens at TCP port 119)

   C:      (requests connection on TCP port 119)
   S:      201 UCB-VAX netnews server ready -- no posting allowed

   C:      GROUP msgs
   S:      211 103 402 504 msgs Your new group is msgs
           (there are 103 articles, from 402 to 504)

   C:      ARTICLE 401
   S:      423 No such article in this newsgroup

   C:      ARTICLE 402
   S:      220 402 <4105@ucbvax.ARPA> Article retrieved, text follows
   S:      (article header and body follow)
   S:      .

   C:      HEAD 403
   S:      221 403 <3108@mcvax.UUCP> Article retrieved, header follows
   S:      (article header follows)
   S:      .

   C:      QUIT
   S:      205 UCB-VAX news server closing connection.  Goodbye.
```

4.3. Example 3 - NEWGROUPS command

```
   S:      (listens at TCP port 119)

   C:      (requests connection on TCP port 119)
   S:      200 Imaginary Institute News Server ready (posting ok)

   (client asks for new newsgroups since April 3, 1985)
   C:      NEWGROUPS 850403 020000

   S:      231 New newsgroups since 03/04/85 02:00:00 follow
```

```
   S:      net.music.gdead
   S:      net.games.sources
   S:      .

   C:      GROUP net.music.gdead
   S:      211 0 1 1 net.music.gdead Newsgroup selected
           (there are no articles in that newsgroup, and
           the first and last article numbers should be ignored)

   C:      QUIT
   S:      205 Imaginary Institute news server ceasing service.  Bye!
```

4.4. Example 4 - posting a news article

```
   S:      (listens at TCP port 119)

   C:      (requests connection on TCP port 119)
   S:      200 BANZAIVAX news server ready, posting allowed.

   C:      POST
   S:      340 Continue posting; Period on a line by itself to end
   C:      (transmits news article in RFC850 format)
   C:      .
   S:      240 Article posted successfully.

   C:      QUIT
   S:      205 BANZAIVAX closing connection.  Goodbye.
```

4.5. Example 5 - interruption due to operator request

```
   S:      (listens at TCP port 119)

   C:      (requests connection on TCP port 119)
   S:      201 genericvax news server ready, no posting allowed.

           (assume normal conversation for some time, and
           that a newsgroup has been selected)

   C:      NEXT
   S:      223 1013 <5734@mcvax.UUCP> Article retrieved; text separate.

   C:      HEAD
   C:      221 1013 <5734@mcvax.UUCP> Article retrieved; head follows.

   S:      (sends head of article, but halfway through is
           interrupted by an operator request.  The following
           then occurs, without client intervention.)
```

```
   S:      (ends current line with a CR-LF pair)
   S:      .
   S:      400 Connection closed by operator.  Goodbye.
   S:      (closes connection)
```

4.6. Example 6 - Using the news server to distribute news between systems.

```
   S:      (listens at TCP port 119)

   C:      (requests connection on TCP port 119)
   S:      201 Foobar NNTP server ready (no posting)

   (client asks for new newsgroups since 2 am, May 15, 1985)
   C:      NEWGROUPS 850515 020000
   S:      235 New newsgroups since 850515 follow
   S:      net.fluff
   S:      net.lint
   S:      .

   (client asks for new news articles since 2 am, May 15, 1985)
   C:      NEWNEWS * 850515 020000
   S:      230 New news since 850515 020000 follows
   S:      <1772@foo.UUCP>
   S:      <87623@baz.UUCP>
   S:      <17872@GOLD.CSNET>
   S:      .

   (client asks for article <1772@foo.UUCP>)
   C:      ARTICLE <1772@foo.UUCP>
   S:      220 <1772@foo.UUCP> All of article follows
   S:      (sends entire message)
   S:      .

   (client asks for article <87623@baz.UUCP>
   C:      ARTICLE <87623@baz.UUCP>
   S:      220 <87623@baz.UUCP> All of article follows
   S:      (sends entire message)
   S:      .

   (client asks for article <17872@GOLD.CSNET>
   C:      ARTICLE <17872@GOLD.CSNET>
   S:      220 <17872@GOLD.CSNET> All of article follows
   S:      (sends entire message)
   S:      .
```

```
   (client offers an article it has received recently)
   C:      IHAVE <4105@ucbvax.ARPA>
   S:      435 Already seen that one, where you been?

   (client offers another article)
   C:      IHAVE <4106@ucbvax.ARPA>
   S:      335 News to me!  <CRLF.CRLF> to end.
   C:      (sends article)
   C:      .
   S:      235 Article transferred successfully.  Thanks.

   (or)

   S:      436 Transfer failed.

   (client is all through with the session)
   C:      QUIT
   S:      205 Foobar NNTP server bids you farewell.
```

4.7. Summary of commands and responses.

The following are the commands recognized and responses returned by the NNTP server.

4.7.1. Commands

```
ARTICLE
BODY
GROUP
HEAD
HELP
IHAVE
LAST
LIST
NEWGROUPS
NEWNEWS
NEXT
POST
QUIT
SLAVE
STAT
```

4.7.2. Responses

```
100 help text follows
199 debug output
```

200 server ready - posting allowed
201 server ready - no posting allowed
202 slave status noted
205 closing connection - goodbye!
211 n f l s group selected
215 list of newsgroups follows
220 n <a> article retrieved - head and body follow 221 n <a> article retrieved - head follows
222 n <a> article retrieved - body follows
223 n <a> article retrieved - request text separately 230 list of new articles by message-id follows
231 list of new newsgroups follows
235 article transferred ok
240 article posted ok

335 send article to be transferred. End with <CR-LF>.<CR-LF>
340 send article to be posted. End with <CR-LF>.<CR-LF>

400 service discontinued
411 no such news group
412 no newsgroup has been selected
420 no current article has been selected
421 no next article in this group
422 no previous article in this group
423 no such article number in this group
430 no such article found
435 article not wanted - do not send it
436 transfer failed - try again later
437 article rejected - do not try again.
440 posting not allowed
441 posting failed

500 command not recognized
501 command syntax error
502 access restriction or permission denied
503 program fault - command not performed

4.8. A Brief Word about the USENET News System

In the UNIX world, which traditionally has been linked by 1200 baud dial-up telephone lines, the USENET News system has evolved to handle central storage, indexing, retrieval, and distribution of news. With the exception of its underlying transport mechanism (UUCP), USENET News is an efficient means of providing news and bulletin service to subscribers on UNIX and other hosts worldwide. The USENET News

system is discussed in detail in RFC 850. It runs on most versions of UNIX and on many other operating systems, and is customarily distributed without charge.

USENET uses a spooling area on the UNIX host to store news articles, one per file. Each article consists of a series of heading text, which contain the sender's identification and organizational affiliation, timestamps, electronic mail reply paths, subject, newsgroup (subject category), and the like. A complete news article is reproduced in its entirety below. Please consult RFC 850 for more details.

```
Relay-Version: version B 2.10.3 4.3bsd-beta 6/6/85; site
sdcsvax.UUCP
Posting-Version: version B 2.10.1 6/24/83 SMI; site unitek.uucp
Path:sdcsvax!sdcrdcf!hplabs!qantel!ihnp4!alberta!ubc-vision!unitek
!honman
From: honman@unitek.uucp (Man Wong)
Newsgroups: net.unix-wizards
Subject: foreground -> background ?
Message-ID: <167@unitek.uucp>
Date: 25 Sep 85 23:51:52 GMT
Date-Received: 29 Sep 85 09:54:48 GMT
Reply-To: honman@unitek.UUCP (Hon-Man Wong)
Distribution: net.all
Organization: Unitek Technologies Corporation
Lines: 12

I have a process (C program) which generates a child and waits for
it to return.  What I would like to do is to be able to run the
child process interactively for a while before kicking itself into
the background so I can return to the parent process (while the
child process is RUNNING in the background).  Can it be done?  And
if it can, how?

Please reply by E-mail.  Thanks in advance.

Hon-Man Wong
```

5. References

[1] Crocker, D., "Standard for the Format of ARPA Internet Text Messages", RFC-822, Department of Electrical Engineering, University of Delaware, August, 1982.

[2] Horton, M., "Standard for Interchange of USENET Messages", RFC-850, USENET Project, June, 1983.

[3] Postel, J., "Transmission Control Protocol- DARPA Internet Program Protocol Specification", RFC-793, USC/Information Sciences Institute, September, 1981.

[4] Postel, J., "Simple Mail Transfer Protocol", RFC-821, USC/Information Sciences Institute, August, 1982.

6. Acknowledgements

The authors wish to express their heartfelt thanks to those many people who contributed to this specification, and especially to Erik Fair and Chuq von Rospach, without whose inspiration this whole thing would not have been necessary.

7. Notes

<1> UNIX is a trademark of Bell Laboratories.