NNTP Extensions Working Group
Internet Draft
Updates: 2980 (if approved)
Expires: December 2005

J. Vinocur
Cornell University
K. Murchison
Oceana Matrix Ltd.

June 2005

NNTP Extension for Streaming Feeds
draft-ietf-nntpext-streaming-06

Status of this memo

Copyright Notice

Abstract

This memo defines an extension to the Network News Transport Protocol (NNTP) to provide asynchronous (otherwise known as "streaming") transfer of articles. This allows servers to transfer articles to other servers with much greater efficiency.

This document updates and formalizes the CHECK and TAKETHIS commands specified in RFC 2980 and deprecates the MODE STREAM

command.

Note to the RFC Editor

The normative reference to RFC 2234 may be replaced by draft-crocker-abnf-rfc2234bis should it reach RFC status before this one.

The normative reference to [NNTP] is a document which is expected to be published simultaneously with this one and so can be replaced by a reference to the resulting RFC.

Table of Contents

1. Introduction

According to the NNTP specification [NNTP], a peer uses the IHAVE command to query whether a server wants a particular article. Because the IHAVE command cannot be pipelined, the need to stop and wait for the remote end's response greatly restricts the throughput that can be achieved.

The ad-hoc CHECK and TAKETHIS commands, originally documented in [NNTP-COMMON], provide an alternative method of peer-to-peer article transfer which permits a more effective use of network bandwidth. Due to their proven usefulness and wide deployment, they are formalized in this specification.

The ad-hoc MODE STREAM command, also documented in [NNTP-COMMON], is deprecated by this specification, but due to its ubiquity is documented here for backwards compatibility.

1.1. Conventions Used in this Document

The notational conventions used in this document are the same as those in [NNTP] and any term not defined in this document has the same meaning as in that one.

The key words "REQUIRED", "MUST", "MUST NOT", "SHOULD", "SHOULD NOT", "MAY", and "OPTIONAL" in this document are to be interpreted as described in "Key words for use in RFCs to Indicate Requirement Levels" [KEYWORDS].

This document assumes you are familiar with NNTP [NNTP]. In general, the connections described below are from one peer to another, but we will continue to use "client" to mean the initiator of the NNTP connection, and "server" to mean the other endpoint.

In the examples, commands from the client are indicated with [C], and responses from the server are indicated with [S].

2. The STREAMING Extension

This extension provides three new commands: MODE STREAM, CHECK, and TAKETHIS. The capability label for this extension is STREAMING.

2.1. Streaming Article Transfer

The STREAMING extension provides the same functionality as the IHAVE command ([NNTP] section 6.3.2) but splits the query and transfer functionality into the CHECK and TAKETHIS commands respectively. This allows the CHECK and TAKETHIS commands to be

pipelined ([NNTP] section 3.5) and provides for "streaming" article transfer.

A streaming client will often pipeline many CHECK commands and use the responses to construct a list of articles to be sent by a pipelined sequence of TAKETHIS commands, thus increasing the fraction of time spent transferring articles. The CHECK and TAKETHIS commands utilize distinct response codes so that these commands can be intermingled in a pipeline and the response to any single command can be definitively identified by the client.

The client MAY send articles via TAKETHIS without first querying the server with CHECK. The client SHOULD NOT send every article in this fashion unless explicitly configured to do so by the site administrator based on out-of-band information. However, the client MAY use an adaptive strategy where it initially sends CHECK commands for all articles, but switches to using TAKETHIS without CHECK if most articles are being accepted (over 95% acceptance might be a reasonable metric in some configurations). If the client uses such a strategy, it SHOULD also switch back to using CHECK on all articles if the acceptance rate ever falls much below the threshold.

2.2. Advertising the STREAMING Extension

A server supporting the streaming commands described in this document will advertise the "STREAMING" capability label in response to the CAPABILITIES command. The server MUST continue to advertise this capability after a client has issued the MODE STREAM command. This capability MAY be advertised both before and after any use of MODE READER, with the same semantics.

Example of a client using CAPABILITIES and MODE STREAM on a mode-switching server:

```
[C] CAPABILITIES
[S] 101 Capability list:
[S] VERSION 2
[S] MODE-READER
[S] IHAVE
[S] LIST ACTIVE
[S] STREAMING
[S] .
[C] MODE STREAM
[S] 203 Streaming permitted
[C] CAPABILITIES
[S] 101 Capability list:
[S] VERSION 2
```

```
[S] MODE-READER
[S] IHAVE
[S] LIST ACTIVE
[S] STREAMING
[S] .
[C] MODE READER
[S] 200 Posting allowed
[C] CAPABILITIES
[S] 101 Capability list:
[S] VERSION 2
[S] READER
[S] POST
[S] LIST ACTIVE NEWSGROUPS HEADERS
[S] HDR
[S] .
```

2.3. MODE STREAM Command

Historically this command was used by a client to discover if a server supported the CHECK and TAKETHIS commands. This command is deprecated in favor of the CAPABILITIES discovery command and is only provided here for compatibility with legacy implementations [NNTP-COMMON] of these transport commands.

New clients SHOULD use the CAPABILITIES command to check a server for support of the STREAMING extension but MAY use the MODE STREAM command for backwards compatibility with legacy servers that don't support the CAPABILITIES discovery command. Servers MUST accept the MODE STREAM command for backwards compatibility with legacy clients which don't use the CAPABILITIES discovery command.

NOTE: This command may be removed from a future version of this specification, therefore clients are urged to transition to the CAPABILITIES command wherever possible.

2.3.1. Usage

```
Syntax
   MODE STREAM

Responses
   203   Streaming permitted
```

2.3.2. Description

If a server supports this extension, it MUST return a 203 response to the MODE STREAM command (or 501 if an argument is given). The MODE STREAM command MUST NOT affect the server state in any way

(that is, it is not a mode change despite the name), therefore this command MAY be pipelined. A server MUST NOT require that the MODE STREAM command be issued by the client before accepting the CHECK or TAKETHIS commands.

2.3.3. Examples

Example of a client checking the ability to stream articles on a server which does not support this extension:

```
[C] MODE STREAM
[S] 501 Unknown MODE variant
```

Example of a client checking the ability to stream articles on a server which supports this extension:

```
[C] MODE STREAM
[S] 203 Streaming permitted
```

2.4. CHECK Command

2.4.1. Usage

Syntax
```
CHECK message-id
```

Responses
```
238 message-id   Send article to be transferred
431 message-id   Transfer not possible; try again later
438 message-id   Article not wanted
```

Parameters
```
message-id = Article message-id
```

The first parameter of the 238, 431, and 438 responses MUST be the message-id provided by the client as the parameter to CHECK.

2.4.2. Description

The CHECK command informs the server that the client has an article with the specified message-id. If the server desires a copy of that article, a 238 response MUST be returned, indicating that the client may send the article using the TAKETHIS command. If the server does not want the article (if, for example, the server already has a copy of it), a 438 response MUST be returned, indicating that the article is not wanted. Finally, if the article isn't wanted immediately but the client should retry later if possible (if, for example, another client has offered to send the

same article to the server), a 431 response MUST be returned.

NOTE: The responses to CHECK are advisory; the server MUST NOT rely on the client to behave as requested by these responses.

2.4.3. Examples

Example of a client checking whether the server would like a set of articles and getting a mixture of responses:

```
[C] CHECK <i.am.an.article.you.will.want@example.com>
[S] 238 <i.am.an.article.you.will.want@example.com>
[C] CHECK <i.am.an.article.you.have@example.com>
[S] 438 <i.am.an.article.you.have@example.com>
[C] CHECK <i.am.an.article.you.defer@example.com>
[S] 431 <i.am.an.article.you.defer@example.com>
```

Example of pipelining the CHECK commands in the previous example:

```
[C] CHECK <i.am.an.article.you.will.want@example.com>
[C] CHECK <i.am.an.article.you.have@example.com>
[C] CHECK <i.am.an.article.you.defer@example.com>
[S] 238 <i.am.an.article.you.will.want@example.com>
[S] 438 <i.am.an.article.you.have@example.com>
[S] 431 <i.am.an.article.you.defer@example.com>
```

2.5. TAKETHIS Command

2.5.1. Usage

A client MUST NOT use this command unless the server advertises the STREAMING capability or returns a 203 response to the MODE STREAM command.

```
Syntax
   TAKETHIS message-id

Responses
   239 message-id   Article transferred OK
   439 message-id   Transfer rejected; do not retry

Parameters
   message-id = Article message-id
```

The first parameter of the 239 and 439 responses MUST be the message-id provided by the client as the parameter to TAKETHIS.

2.5.2. Description

The TAKETHIS command is used to send an article with the specified message-id to the server. The article is sent immediately following the CRLF at the end of the TAKETHIS command line. The client MUST send the entire article, including headers and body, to the server as a multi-line data block ([NNTP] section 3.1.1). Thus a single dot (".") on a line indicates the end of the text, and lines starting with a dot in the original text have that dot doubled during transmission. The server MUST return either a 239 response, indicating that the article was successfully transferred, or a 439 response, indicating that the article was rejected. If the server encounters a temporary error that prevents it from processing the article but does not want to reject the article, it MUST reply with a 400 response to the client and close the connection.

This function differs from the POST command in that it is intended for use in transferring already-posted articles between hosts. It SHOULD NOT be used when the client is a personal news reading program, since use of this command indicates that the article has already been posted at another site and is simply being forwarded from another host. However, despite this, the server MAY elect not to post or forward the article if, after further examination of the article, it deems it inappropriate to do so. 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.

The client SHOULD NOT assume that the article has been successfully transferred unless it receives an affirmative response from the server. A lack of response (such as a dropped network connection or a network timeout) or a 400 response SHOULD be treated as a temporary failure and cause the transfer to be tried again later if possible.

Because some news server software may not be able immediately to determine whether or not an article is suitable for posting or forwarding, an NNTP server MAY acknowledge the successful transfer of the article (with a 239 response) but later silently discard it.

2.5.3. Examples

Example of streaming two articles to another site (the first article is accepted and the second is rejected):

```
      [C] TAKETHIS <i.am.an.article.you.will.want@example.com>
      [C] Path: pathost!demo!somewhere!not-for-mail
      [C] From: "Demo User" <nobody@example.com>
      [C] Newsgroups: misc.test
      [C] Subject: I am just a test article
      [C] Date: 6 Oct 1998 04:38:40 -0500
      [C] Organization: An Example Com, San Jose, CA
      [C] Message-ID: <i.am.an.article.you.will.want@example.com>
      [C]
      [C] This is just a test article.
      [C] .
      [C] TAKETHIS <i.am.an.article.you.have@example.com>
      [C] Path: pathost!demo!somewhere!not-for-mail
      [C] From: "Demo User" <nobody@example.com>
      [C] Newsgroups: misc.test
      [C] Subject: I am just a test article
      [C] Date: 6 Oct 1998 04:38:40 -0500
      [C] Organization: An Example Com, San Jose, CA
      [C] Message-ID: <i.am.an.article.you.have@example.com>
      [C]
      [C] This is just a test article.
      [C] .
      [S] 239 <i.am.an.article.you.will.want@example.com>
      [S] 439 <i.am.an.article.you.have@example.com>
```

Example of sending an article to a site where the transfer fails:

```
      [C] TAKETHIS <i.am.an.article.you.will.want@example.com>
      [C] Path: pathost!demo!somewhere!not-for-mail
      [C] From: "Demo User" <nobody@example.com>
      [C] Newsgroups: misc.test
      [C] Subject: I am just a test article
      [C] Date: 6 Oct 1998 04:38:40 -0500
      [C] Organization: An Example Com, San Jose, CA
      [C] Message-ID: <i.am.an.article.you.will.want@example.com>
      [C]
      [C] This is just a test article.
      [C] .
      [S] 400 Service temporarily unavailable
      [Server closes connection.]
```

3. Augmented BNF Syntax for the STREAMING Extension

This section describes the formal syntax of the STREAMING extension using ABNF [ABNF]. It extends the syntax in section 9 of [NNTP], and non-terminals not defined in this document are defined there. The [NNTP] ABNF should be imported first before attempting to validate these rules.

3.1. Commands

This syntax extends the non-terminal "command", which represents an NNTP command.

```
command =/ check-command /
     mode-stream-command /
     takethis-command

check-command       = "CHECK" WS message-id
mode-stream-command = "MODE" WS "STREAM"
takethis-command    = "TAKETHIS" WS message-id
```

3.2. Command Datastream

This syntax extends the non-terminal "command-datastream", which represents the further material sent by the client in the case of streaming commands.

```
command-datastream =/ takethis-datastream

takethis-datastream = encoded-article
```

3.3. Responses

This syntax extends the non-terminal "initial-response-content", which represents an initial response line sent by the server.

```
initial-response-content =/ response-238-content /
     response-239-content /
     response-431-content /
     response-438-content /
     response-439-content

response-238-content = "238" SP message-id
response-239-content = "239" SP message-id
response-431-content = "431" SP message-id
response-438-content = "438" SP message-id
response-439-content = "439" SP message-id
```

3.4. Capability Entries

This syntax extends the non-terminal "capability-entry", which represents a capability that may be advertised by the server.

```
capability-entry =/ streaming-capability

streaming-capability = "STREAMING"
```

4. Summary of Response Codes

This section contains a list of every new response code defined in this document, whether it is multi-line, which commands can generate it, what arguments it has, and what its meaning is.

Response code 203
   Generated by: MODE STREAM
   Meaning: streaming permitted.

Response code 238
   Generated by: CHECK
   1 argument: message-id
   Meaning: send article to be transferred.

Response code 239
   Generated by: TAKETHIS
   1 argument: message-id
   Meaning: article transferred OK.

Response code 431
   Generated by: CHECK
   1 argument: message-id
   Meaning: transfer not possible; try again later.

Response code 438
   Generated by: CHECK
   1 argument: message-id
   Meaning: article not wanted.

Response code 439
   Generated by: TAKETHIS
   1 argument: message-id
   Meaning: transfer rejected; do not retry.

5. Security Considerations

No new security considerations are introduced by this extension, beyond those already described in the core specification [NNTP].

6. IANA Considerations

This section gives a formal definition of the STREAMING extension as required by Section 3.3.3 of [NNTP] for the IANA registry.

- o The STREAMING extension provides for streaming transfer of articles.

- o The capability label for this extension is "STREAMING".
- o The capability label has no arguments.
- o The extension defines three new commands, MODE STREAM, CHECK, and TAKETHIS, whose behavior, arguments, and responses are defined in Sections 2.2, 2.3, and 2.4 respectively.
- o The extension does not associate any new responses with pre-existing NNTP commands.
- o The extension does not affect the behavior of a server or client other than via the new commands.
- o The extension does not affect the maximum length of commands or initial response lines.
- o The extension does not alter pipelining, and the MODE STREAM, CHECK and TAKETHIS command can be pipelined.
- o Use of this extension does not alter the capabilities list.
- o The extension does not cause any pre-existing command to produce a 401, 480, or 483 response.
- o Use of the MODE READER command on a mode-switching server may disable this extension.
- o Published Specification: This document.
- o Author, Change Controller, and Contact for Further Information: Author of this document.

7. References

7.1. Normative References

[ABNF] Crocker, D., Overell, P., "Augmented BNF for Syntax Specifications: ABNF", RFC 2234, November 1997.

[KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[NNTP] Feather, C., "Network News Transport Protocol", draft-ietf-nntpext-base-*.txt, Work in Progress.

7.2. Informative References

[NNTP-COMMON] Barber, S., "Common NNTP Extensions", RFC 2980, October 2000.

8. Authors' Addresses

Jeffrey M. Vinocur
Department of Computer Science
Upson Hall
Cornell University
Ithaca, NY 14853

EMail: vinocur@cs.cornell.edu

Kenneth Murchison
Oceana Matrix Ltd.
21 Princeton Place
Orchard Park, NY 14127 USA

Email: ken@oceana.com

9. Acknowledgments

This document is based heavily on the relevant sections of RFC 2980 [NNTP-COMMON], by Stan Barber.

Special acknowledgment also goes to Russ Allbery, Clive Feather, Andrew Gierth, and others who commented privately on intermediate revisions of this document, as well as the members of the IETF NNTP Working Group for continual (yet sporadic) insight in discussion.

10. Intellectual Property Rights

11. Copyright