## File Transfer Protocol RANG Command for Octet Ranges
### draft-bryan-ftp-range-08

Abstract

   The File Transfer Protocol offers the REST command to designate a
   starting point for a transfer, but does not currently offer any
   method to specify an end point.  This document specifies a new FTP
   RANG command to be used by clients to designate an octet range to a
   file, including a start and end point.  This range can be used to
   permit restarts and repairs of interrupted data transfers in STREAM
   mode, among other things.

Editorial Note (To be removed by RFC Editor)

   Discussion of this draft should take place on the FTPEXT2 working
   group mailing list (ftpext@ietf.org), although this draft is not a WG
   item.  Related documents (including fancy diffs) can be found at
   <http://tools.ietf.org/wg/ftpext2/>.

   The changes in this draft are summarized in Appendix B.

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 August 3, 2013.

Copyright Notice

Table of Contents

## 1.  Introduction

   The File Transfer Protocol offers the REST command [RFC3659] to
   designate a starting point for a transfer, but does not currently
   offer any method to specify an end point.  This document specifies a
   new FTP RANG command to be used by clients to designate an octet
   range to a file, including a start and end point.  This range can be
   used to permit restarts and repairs of interrupted data transfers in
   STREAM mode, among other things.

   The current alternatives for data transfers, without being able to
   specify an end point, are to issue an ABOR command or close the data
   connection.

   HTTP offers similar functionality with the Range: header field in
   Section 14.35 of [RFC2616], where a specific octet (8 bit byte) range
   can optionally be requested.


## 2.  Document Conventions

   This specification describes conformance of File Transfer Protocol
   Extension for RANG, a start and end point in an octet range.

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
   document are to be interpreted as described in BCP 14, [RFC2119], as
   scoped to those conformance targets.

   This document also uses notation defined in STD 9, [RFC0959].  In
   particular, the terms or commands "reply", "user", "file", "FTP
   commands", "user-PI" (user protocol interpreter), "server-FTP
   process", "server-PI", "mode", "Image type", "Stream transfer mode",
   "type", "STOR", "RETR", and "ASCII", are all used here as defined
   there.  The command "REST" is used as defined in Section 5 of
   [RFC3659].

   In the examples of FTP dialogs presented in this document, lines that
   begin "C> " were sent over the control connection from the user-PI to
   the server-PI, and lines that begin "S> " were sent over the control
   connection from the server-PI to the user-PI.  In all cases, the
   prefixes shown above, including the one space, have been added for
   the purposes of this document, and are not a part of the data
   exchanged between client and server.

   Syntax required is defined using the Augmented BNF defined in
   [RFC5234].

## 2.1.  Basic Tokens

   This document imports the core definitions given in Appendix B of
   [RFC5234].  There definitions will be found for basic ABNF elements
   like ALPHA, DIGIT, SP, etc.  To that, the following term is added for
   use in this document.


      TCHAR = VCHAR / SP / HTAB    ; visible plus white space

   The VCHAR (from [RFC5234]) and TCHAR rules give basic character types
   from varying sub-sets of the ASCII character set for use in various
   commands and responses.

   Note that in ABNF, string literals are case insensitive.  That
   convention is preserved in this document, and implies that FTP
   commands and parameters that are added by this specification have
   values that can be represented in any case.  That is, "RANG" is the
   same as "rang", "Rang", "RaNg", etc., and "ftp.example.com" is the
   same as "Ftp.Example.Com", "fTp.eXample.cOm", etc.

## 2.2.  Server Replies

   Section 4.2 of [RFC0959] defines the format and meaning of replies by
   the server-PI to FTP commands from the user-PI.  Those reply
   conventions are used here without change.


      error-response = error-code SP *TCHAR CRLF
      error-code     = ("4" / "5") 2DIGIT

   Implementers should note that the ABNF syntax (which was not used in
   [RFC0959]) used in this document, and other FTP related documents,
   sometimes shows replies using the one line format.  Unless otherwise
   explicitly stated, that is not intended to imply that multi-line
   responses are not permitted.  Implementers should assume that, unless
   stated to the contrary, any reply to any FTP command (including QUIT)
   can be of the multi-line format described in [RFC0959].

   Throughout this document, replies will be identified by the three
   digit code that is their first element.  Thus the term "500 reply"
   means a reply from the server-PI using the three digit code "500".


## 3.  The RANGe Command (RANG)

   A new command "RANG" is added to the FTP command set to allow the
   client to specify both a start point octet range and an end point

octet range of a file from a server-FTP process.

The syntax for the RANG command when the current transfer mode is
STREAM is:


    range-command = "RANG" SP start-point SP end-point CRLF
    start-point   = 1*DIGIT
    end-point     = 1*DIGIT

<start-point> gives the number of octets, counted from the beginning
of the file, that the range starts with.  A value of zero effectively
causes the octet range to be started at first octet.  For example,
"RANG 0 0" would select the first octet of a file.

<end-point> gives the number of octets, counted from the beginning of
the file, that the range ends with.  (That is, the end point is
relative to the start of the file and not relative to the start
point).  NOTE: end-point is inclusive.  For example, "RANG 0 1" would
select the first two octets of a file.

The server-PI will respond to a successful RANG command with a 350
reply, indicating that RANG parameters have been saved, and that
another command, which could be one of the standard file transfer
commands or otherwise, should then follow to complete the ranged
request.

To reset the range command, "RANG 1 0" should be issued.  The
server-PI MUST reply with a 350 reply if "RANG 1 0" is issued by
client-PI because it is a valid way of resetting the range.  (The
range would also be reset if the session is reinitialized with REIN
but this terminates the user and resets all parameters).

RANG requests where <start-point> is larger than <end-point>
(excluding "RANG 1 0") are invalid.  The server-PI responds with 501
to these invalid requests and automatically resets the octet
selection to the default, which is the whole file.


    range-response = range-ok / error-response
    range-ok       = "350" SP *TCHAR CRLF

## 3.1.  FEAT Command Response for RANG Command

When replying to the FEAT command [RFC2389], a server-FTP process
that supports the RANG command, as specified here, MUST include, a
line containing exactly the string "RANG STREAM".  This string is
case insensitive, and MAY be sent in any mixture of upper or lower

case, however it SHOULD be sent in upper case.  That is, the response
SHOULD be:


```
C> FEAT
S> 211-Extensions supported:
S>  ...
S>  RANG STREAM
S>  ...
S> 211 END
```

The ellipses indicate place holders where other features may be
included, and are not required.  The one-space indentation of the
feature lines is mandatory [RFC2389].


```
range-feat = SP "RANG" SP "STREAM" CRLF
```


## 4.  RANG Command Use with RETR and STOR

### 4.1.  Octet Ranges in STREAM Mode

To get a specific part of a file without sending the entire file,
both sides need some way to agree on where in the data stream to
start and end the data transfer.

In STREAM mode, the data connection contains just a stream of
unformatted octets of data.  Explicit restart markers thus cannot be
inserted into the data stream, they would be indistinguishable from
data.  For this reason, the FTP specification [RFC0959] did not
provide the ability to do restarts in stream mode.  However, there is
not really a need to have explicit restart markers in this case, as
restart markers can be implied by the octet offset into the data
stream.

Because the data stream defines the file in STREAM mode, a different
data stream would represent a different file.  Thus, an offset will
always represent the same position within a file.  On the other hand,
in other modes than STREAM, the same file can be transferred using
quite different octet sequences and yet be reconstructed into the one
identical file.  Thus an offset into the data stream in transfer
modes other than STREAM would not give an unambiguous restart or end
point.

If the data representation TYPE is IMAGE and the STRUcture is File,
for many systems the file will be stored exactly in the same format
as it is sent across the data connection.  It is then usually very

easy for the receiver to determine how much data was previously received, and notify the sender of the offset where the transfer should be restarted.  In other representation types and structures more effort will be required, but it remains always possible to determine the offset with finite, but perhaps non-negligible, effort. In the worst case, an FTP process may need to open a data connection to itself, set the appropriate transfer type and structure, and actually transmit the file, counting the transmitted octets.

If the user-FTP process is intending to restart a retrieve, it will directly calculate the restart marker and send that information in the RANGe command.  However, if the user-FTP process is intending to restart sending the file, it needs to be able to determine how much data was previously sent, and correctly received and saved.  The purpose of the SIZE command, as documented in Section 4 of [RFC3659], is to get this information.

## 4.2.  Error Recovery and Range Requests

STREAM mode transfers with FILE STRUcture may be range requested even though no restart marker has been transferred in addition to the data itself.  This is done by using the SIZE command, if needed, in combination with the RANG command, and one of the standard file transfer commands.

When using TYPE ASCII or IMAGE, the SIZE command will return the number of octets that would actually be transferred if the file were to be sent between the two systems, i.e., with type IMAGE, the SIZE normally would be the number of octets in the file.  With type ASCII, the SIZE would be the number of octets in the file including any modifications required to satisfy the TYPE ASCII CR-LF end-of-line convention.

## 4.3.  RETR and STOR

The RANG command can be used to either resume a failed transfer or to selectively modify a range of octets in a file.  Use with RETR is comparatively well defined in all cases, as the client bears the responsibility of merging the retrieved data with the partially retrieved file.  It may choose to use the data obtained other than to complete an earlier transfer, or to re-retrieve data that had been retrieved before.

Server-FTP processes may permit transfer commands other than RETR and STOR, such as APPE and STOU, to complete a restart or repair; however, this is not recommended.  STOU (store unique) is undefined in this usage, as storing the remainder of a file into a unique file name is rarely going to be useful.  If APPE (append) is permitted, it

MUST act identically to STOR when a range has been requested.  That
is, in both cases, octets from the data connection are placed into
the file at the location bounded by the start-point and end-point
values.

The server-PI SHOULD transfer 0 octets with RETR if the specified
start point or start point and end point are larger than the actual
file size.

The server-PI SHOULD transfer the whole range from the start point to
the end of the file with RETR if the end point is larger than the
actual file.

With STOR however, the server-PI must insert the data into the file
named.  The combination of RANG and STOR MUST NOT modify any data
outside the specified range.  For a successful transfer, the client
has to send the full range specified.  If the client sends a
different amount of octets or if the connection is aborted, the
results are unspecified for the data in the specified range only.

The server-PI should reject the STOR if a client issues a RANG with a
start-point greater than the actual file size.  If the start-point is
less than or equal to the file size, the file is opened for writing,
continuing at start-point.

If the file has been smaller than end-point prior to the STOR, it
will be exactly end-point octets in size after a successful transfer.
The RANG command MUST NOT cause a file to be truncated, if the end-
point is smaller than or equal to the file size, the file size MUST
remain unmodified.

The RANG command MUST be the last command issued before the data
transfer command that is to cause a partial data transfer.  The
effect of issuing a RANG command at any other time is undefined.  The
server-PI may react to a badly positioned RANG command by issuing an
error response to the following command, not being a restartable data
transfer command, or it may save the start-point and/or end-point
octet range value and apply it to the next data transfer command, or
it may silently ignore the inappropriate restart attempt.  Because of
this, a user-PI that has issued a RANG command, but that has not
successfully transmitted the following data transfer command for any
reason, should send another RANG command before the next data
transfer command.  If that transfer is not to be restarted, then
"RANG 1 0" should be issued to reset the range selection.

An error response will follow a RANG command only when the server
does not implement the command, or when command syntax is invalid.
Any other errors, including such problems as start-point and/or end-

point octet range out of range, should be reported when the following
transfer command is issued.  Such errors will cause that transfer
request to be rejected with an error indicating the invalid restart
attempt.

### 4.4.  User-PI usage of RANG with RETR and STOR

The user-PI issues the FEAT command to query the server-PI if it
supports the RANG command.  In this example, the server-PI also
supports REST.

```
C> FEAT
S> 211-Extensions supported:
S>  ...
S>  REST STREAM
S>  RANG STREAM
S>  ...
S> 211 END
```

Assume that the transfer of a largish file has previously been
interrupted after 802816 octets had been received by the client, that
the transfer should stop at octet 1000000 of the file, that the
previous transfer was with TYPE=I, and that it has been verified that
the file on the server has not since changed.  In the example below,
data is sent from offset 802816 to, and including, offset 1000000.

```
C> TYPE I
S> 200 Type set to I.
C> PORT 127,0,0,1,15,107
S> 200 PORT command successful.
C> RANG 802816 1000000
S> 350 Restarting at 802816. End Byte range at 1000000.
C> RETR cap60.pl198.tar
S> 150 Opening BINARY mode data connection
[...]
S> 226 Transfer complete.
```

In this example below with STOR, assume that the transfer of a
largish file has previously been interrupted after 802816 octets had
been received by the server, that the transfer should stop at octet
1000000 of the file, that the previous transfer was with TYPE=I, and
that it has been verified that the file on the server has not since
changed.  In the example below, data is sent from offset 802816 to,
and including, offset 1000000

```
C> TYPE I
S> 200 Type set to I.
C> PORT 127,0,0,1,15,107
S> 200 PORT command successful.
C> SIZE metalinkc.py
S> 213 802816
C> RANG 802816 1000000
S> 350 Restarting at 802816. End Byte range at 1000000.
C> STOR metalinkc.py
S> 125 Data connection already open; Transfer starting.
[...]
S> 226 Transfer complete.
```

## 5.  RANG Command Errors

Where the RANG command is unrecognized or there is a syntax error in
parameters or arguments, a 500 or 501 reply can be sent by the
server-PI, as specified in [RFC0959].

The server-PI SHOULD reply with a 551 reply if the server-PI is not
configured to use TYPE I and MODE S.

The server-PI SHOULD reply with a 552 reply if the user is not
allowed to use the RANG command.

## 6.  RANG Command Use with Other Commands

This specification defines the use of RANG in combination with other
specific commands.  Other commands are undefined but could decide to
use RANG in a similar way, to select an octet range, and their
specification would define how they operate with RANG.  The HASH
command [draft-bryan-ftpext-hash] uses RANG to select an octet range
for partial file hashing.

## 7.  IANA Considerations

This new command is added to the "FTP Commands and Extensions"
registry created by [RFC5797].

Command Name: RANG

Description: End point octet range (for STREAM mode).

FEAT String: RANG STREAM

   Command Type: Service execution/parameter setting

   Conformance Requirements: Optional

   Reference: This specification


## 8.  Security Considerations

   This memo does not directly concern security.  It is not believed
   that any of the mechanisms documented here impact in any particular
   way upon the security of FTP.


## 9.  References

### 9.1.  Normative References

   [RFC0959]  Postel, J. and J. Reynolds, "File Transfer Protocol",
              STD 9, RFC 0959, October 1985.

   [RFC2119]  Bradner, S., "Key words for use in RFCs to Indicate
              Requirement Levels", BCP 14, RFC 2119, March 1997.

   [RFC2389]  Hethmon, P. and R. Elz, "Feature negotiation mechanism for
              the File Transfer Protocol", RFC 2389, August 1998.

   [RFC3659]  Hethmon, P., "Extensions to FTP", RFC 3659, March 2007.

   [RFC5234]  Crocker, D. and P. Overell, "Augmented BNF for Syntax
              Specifications: ABNF", STD 68, RFC 5234, January 2008.

### 9.2.  Informative References

   [RFC2616]  Fielding, R., Gettys, J., Mogul, J., Frystyk, H.,
              Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext
              Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

   [RFC5797]  Klensin, J. and A. Hoenes, "FTP Command and Extension
              Registry", RFC 5797, March 2010.

   [draft-bryan-ftpext-hash]
              Bryan, A., Kosse, T., and D. Stenberg, "FTP Extensions for
              Cryptographic Hashes", draft-bryan-ftpext-hash-01 (work in
              progress).

Appendix A.  Acknowledgements and Contributors

   Thanks to the FTPEXT2 Working Group, Kamil Dudka, and Alun Jones.

   Portions of [RFC3659] were wholly reused in this document.


Appendix B.  Document History

   [[ to be removed by the RFC editor before publication as an RFC. ]]

   Known issues concerning this draft:
   o  <https://github.com/antbryan/internetdraft/issues>

   draft-bryan-ftp-range-08 : January 30, 2013.
   o  Re-organize, RANG is not transfer specific, it can be used by
      other commands like HASH.  FTPEXT2 mailing list feedback.

   draft-bryan-ftp-range-07 : January 19, 2013.
   o  FTPEXT2 WG concluded, HASH draft renamed.

   draft-bryan-ftp-range-06 : May 24, 2012.
   o  Editorial, use "octet".

   draft-bryan-ftp-range-05 : April 6, 2012.
   o  FTPEXT2 WG concluded, HASH draft renamed.

   draft-bryan-ftp-range-04 : March 27, 2012.
   o  Editorial nits.

   draft-bryan-ftp-range-03 : March 14, 2011.
   o  Refinements.

   draft-bryan-ftp-range-02 : February 1, 2011.
   o  Refinements.

   draft-bryan-ftp-range-01 : January 25, 2011.
   o  Refinements.  "RANG 1 0" resets octet selection.

   draft-bryan-ftp-range-00 : December 13, 2010.
   o  Initial draft.

Authors' Addresses

   Anthony Bryan
   Pompano Beach, FL
   USA

   Email: anthonybryan@gmail.com
   URI:   http://www.metalinker.org


   Tatsuhiro Tsujikawa
   Shiga
   Japan

   Email: tatsuhiro.t@gmail.com
   URI:   http://aria2.sourceforge.net


   Daniel Stenberg

   Email: daniel@haxx.se
   URI:   http://www.haxx.se/


   Tim Kosse

   Email: tim.kosse@filezilla-project.org
   URI:   http://filezilla-project.org/