```
Internet Engineering Task Force (IETF)                        P. Masotta
Request for Comments: 7440                                         Serva
Category: Standards Track                                   January 2015
ISSN: 2070-1721
```

TFTP Windowsize Option

Abstract

The "Trivial File Transfer Protocol" (RFC 1350) is a simple, lockstep, file transfer protocol that allows a client to get or put a file onto a remote host. One of its primary uses is in the early stages of nodes booting from a Local Area Network (LAN). TFTP has been used for this application because it is very simple to implement. The employment of a lockstep scheme limits throughput when used on a LAN.

This document describes a TFTP option that allows the client and server to negotiate a window size of consecutive blocks to send as an alternative for replacing the single-block lockstep schema. The TFTP option mechanism employed is described in "TFTP Option Extension" (RFC 2347).

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

TFTP is virtually unused for Internet transfers today, TFTP is still massively used in network boot/installation scenarios including EFI (Extensible Firmware Interface). TFTP's inherently low transfer rate has been, so far, partially mitigated by the use of the blocksize negotiated extension [RFC2348]. Using this method, the original limitation of 512-byte blocks are, in practice, replaced in Ethernet environments by blocks no larger than 1468 Bytes to avoid IP block fragmentation. This strategy produces insufficient results when transferring big files, for example, the initial ramdisk of Linux distributions or the PE images used in network installations by Microsoft WDS/MDT/SCCM. Considering TFTP looks far from extinction today, this document presents a negotiated extension, under the terms of the "TFTP Option Extension" [RFC2347], that produces TFTP transfer rates comparable to those achieved by modern file transfer protocols.

2. Conventions Used in This Document

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

In this document, these words will appear with that interpretation only when in ALL CAPS. Lowercase uses of these words are not to be interpreted as carrying the significance given in RFC 2119.

3. Windowsize Option Specification

The TFTP Read Request or Write Request packet is modified to include the windowsize option as follows. Note that all fields except "opc" MUST be ASCII strings followed by a single-byte NULL character.

```
   2B     string   1B   string   1B     string     1B   string   1B
+-------+---~~---+----+---~~---+----+-----~~-----+----+---~~---+----+
|  opc  |filename|  0 |  mode  |  0 | windowsize |  0 | #blocks|  0 |
+-------+---~~---+----+---~~---+----+-----~~-----+----+---~~---+----+
```

opc
   The opcode field contains either a 1 for Read Requests or a 2 for Write Requests, as defined in [RFC1350].

filename
   The name of the file to be read or written, as defined in [RFC1350].

mode
   The mode of the file transfer: "netascii", "octet", or "mail", as defined in [RFC1350].

windowsize
   The windowsize option, "windowsize" (case insensitive).

#blocks
   The base-10 ASCII string representation of the number of blocks in a window. The valid values range MUST be between 1 and 65535 blocks, inclusive. The windowsize refers to the number of consecutive blocks transmitted before stopping and waiting for the reception of the acknowledgment of the last block transmitted.

For example:

```
+------+--------+----+-------+----+-----------+----+----+----+
|0x0001| foobar |0x00| octet |0x00| windowsize |0x00| 16 |0x00|
+------+--------+----+-------+----+-----------+----+----+----+
```

is a Read Request for the file named "foobar" in octet transfer mode with a windowsize of 16 blocks (option blocksize is not negotiated in this example, the default of 512 Bytes per block applies).

If the server is willing to accept the windowsize option, it sends an Option Acknowledgment (OACK) to the client. The specified value MUST be less than or equal to the value specified by the client. The client MUST then either use the size specified in the OACK or send an ERROR packet, with error code 8, to terminate the transfer.

The rules for determining the final packet are unchanged from [RFC1350] and [RFC2348].

The reception of a data window with a number of blocks less than the negotiated windowsize is the final window. If the windowsize is greater than the amount of data to be transferred, the first window is the final window.

4. Traffic Flow and Error Handling

The next diagram depicts a section of the traffic flow between the Data Sender (DSND) and the Data Receiver (DRCV) parties on a generic windowsize TFTP file transfer.

The DSND MUST cyclically send to the DRCV the agreed windowsize consecutive data blocks before normally stopping and waiting for the ACK of the transferred window. The DRCV MUST send to the DSND the ACK of the last data block of the window in order to confirm a successful data block window reception.

In the case of an expected ACK not timely reaching the DSND (timeout), the last received ACK SHALL set the beginning of the next windowsize data block window to be sent.

In the case of a data block sequence error, the DRCV SHOULD notify the DSND by sending an ACK corresponding to the last data block correctly received. The notified DSND SHOULD send a new data block window whose beginning MUST be set based on the ACK received out of sequence.

Traffic with windowsize = 1 MUST be equivalent to traffic specified by [RFC1350].

For normative traffic not specifically addressed in this section, please refer to [RFC1350] and its updates.

```
        [ DRCV ]      <---traffic--->      [ DSND ]
          ACK#      ->               <-   Data Block#   window block#
                             ...
                             <-           |DB n+01|          1
                             <-           |DB n+02|          2
                             <-           |DB n+03|          3
                             <-           |DB n+04|          4
        |ACK n+04|           ->
                             <-           |DB n+05|          1
                      Error |<-           |DB n+06|          2
                             <-           |DB n+07|          3
        |ACK n+05|           ->
                             <-           |DB n+06|          1
                             <-           |DB n+07|          2
                             <-           |DB n+08|          3
                             <-           |DB n+09|          4
        |ACK n+09|           ->
                             <-           |DB n+10|          1
                      Error |<-           |DB n+11|          2
                             <-           |DB n+12|          3
        |ACK n+10|           ->| Error
                             <-           |DB n+13|          4
                                         - timeout -
                             <-           |DB n+10|          1
                             <-           |DB n+11|          2
                             <-           |DB n+12|          3
                             <-           |DB n+13|          4
        |ACK n+13|           ->
                             ...
```

Section of a Windowsize = 4 TFTP Transfer
Including Errors and Error Recovery

5. Proof of Concept and Windowsize Evaluation

Performance tests were run on the prototype implementation using a variety of windowsizes and a fixed blocksize of 1456 bytes. The tests were run on a lightly loaded Gigabit Ethernet, between two Toshiba Tecra Core 2 Duo 2.2 Ghz laptops, in "octet" mode, transferring a 180 MByte file.

```
              ^
              |
          300 +
      Seconds |                           windowsize | time (s)
              |                           ----------   ------
              |     x                         1         257
          250 +                               2         131
              |                               4          76
              |                               8          54
              |                              16          42
          200 +                              32          38
              |                              64          35
              |
              |
          150 +
              |
              |           x
              |
          100 +
              |
              |                 x
              |
           50 +                       x
              |                             x
              |                                   x     x
              |
            0 +-//--+-----+-----+-----+-----+-----+-----+-->
                    1     2     4     8    16    32    64

                   Windowsize (in Blocks of 1456 Bytes)
```

Comparatively, the same 180 MB transfer performed over a drive mapped on Server Message Block (SMB) / Common Internet File System (CIFS) on the same scenario took 23 seconds.

The comparison of transfer times (without a gateway) between the standard lockstep schema and the negotiated windowsizes are:

Windowsize	Time Reduction (%)
1	-0%
2	-49%
4	-70%
8	-79%
16	-84%
32	-85%
64	-86%

The transfer time decreases with the use of a windowed schema. The reason for the reduction in time is the reduction in the number of the required synchronous acknowledgements exchanged.

The choice of appropriate windowsize values on a particular scenario depends on the underlying networking technology and topology, and likely other factors as well. Operators SHOULD test various values and SHOULD be conservative when selecting a windowsize value because as the former table and chart shows, there is a point where the benefit of continuing to increase the windowsize is subject to diminishing returns.

6. Congestion and Error Control

From a congestion control (CC) standpoint, the number of blocks in a window does not pose an intrinsic threat to the ability of intermediate devices to signal congestion through drops. The rate at which TFTP UDP datagrams are sent SHOULD follow the CC guidelines in Section 3.1 of [RFC5405].

From an error control standpoint, while [RFC1350] and subsequent updates do not specify a circuit breaker (CB), existing implementations have always chosen to fail under certain circumstances. Implementations SHOULD always set a maximum number of retries for datagram retransmissions, imposing an appropriate threshold on error recovery attempts, after which a transfer SHOULD always be aborted to prevent pathological retransmission conditions.

An implementation example scaled for an Ethernet environment (1 Gbit/s, MTU=1500) would be to set:

```
windowsize = 8
blksize = 1456
maximum retransmission attempts per block/window = 6
timeout between retransmissions = 1 S
minimum inter-packet delay = 80 uS
```

Implementations might well choose other values based on expected and/or tested operating conditions.

7. Security Considerations

TFTP includes no login or access control mechanisms. Care must be taken when using TFTP for file transfers where authentication, access control, confidentiality, or integrity checking are needed. Note that those security services could be supplied above or below the layer at which TFTP runs. Care must also be taken in the rights granted to a TFTP server process so as not to violate the security of the server’s file system. TFTP is often installed with controls such that only files that have public read access are available via TFTP. Also listing, deleting, renaming, and writing files via TFTP are typically disallowed. TFTP file transfers are NOT RECOMMENDED where the inherent protocol limitations could raise insurmountable liability concerns.

TFTP includes no protection against an on-path attacker; care must be taken in controlling windowsize values according to data sender, data receiver, and network environment capabilities. TFTP service is frequently associated with bootstrap and initial provisioning activities; servers in such an environment are in a position to impose device or network specific throughput limitations as appropriate.

This document does not add any security controls to TFTP; however, the specified extension does not pose additional security risks either.

8. References

8.1. Normative References

[RFC1350] Sollins, K., "The TFTP Protocol (Revision 2)", STD 33, RFC 1350, July 1992, <http://www.rfc-editor.org/info/rfc1350>.

[RFC2347] Malkin, G. and A. Harkin, "TFTP Option Extension", RFC 2347, May 1998, <http://www.rfc-editor.org/info/rfc2347>.

[RFC2348] Malkin, G. and A. Harkin, "TFTP Blocksize Option", RFC 2348, May 1998, <http://www.rfc-editor.org/info/rfc2348>.

[RFC5405] Eggert, L. and G. Fairhurst, "Unicast UDP Usage Guidelines for Application Designers", BCP 145, RFC 5405, November 2008, <http://www.rfc-editor.org/info/rfc5405>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

Author’s Address

Patrick Masotta
Serva

EMail: patrick.masotta.ietf@vercot.com
URI: http://www.vercot.com/˜serva/