                        IP MTU Discovery Options

STATUS OF THIS MEMO

   A pair of IP options that can be used to learn the minimum MTU of a
   path through an internet is described, along with its possible uses.
   This is a proposal for an Experimental protocol.  Distribution of
   this memo is unlimited.

INTRODUCTION

   Although the Internet Protocol allows gateways to fragment packets
   that are too large to forward, fragmentation is not always desirable.
   It can lead to poor performance or even total communication failure
   in circumstances that are surprisingly common.  (For a thorough
   discussion of this issue, see [1]).

   A datagram will be fragmented if it is larger than the Maximum
   Transmission Unit (MTU) of some network along the path it follows.
   In order to avoid fragmentation, a host sending an IP datagram must
   ensure that the datagram is no larger than the Minimum MTU (MINMTU)
   over the entire path.

   It has long been recognized that the methods for discovering the
   MINMTU of an IP internetwork path are inadequate.  The methods
   currently available fall into two categories: (1) choosing small MTUs
   to avoid fragmentation or (2) using additional probe packets to
   discover when fragmentation will occur.  Both methods have problems.

   Choosing MTUs requires a balance between network utilization (which
   requires the use of the largest possible datagram) and fragmentation
   avoidance (which in the absence of knowledge about the network path
   encourages the use of small, and thus too many, datagrams).  Any
   choice for the MTU size, without information from the network, is
   likely to either fail to properly utilize the network or fail to
   avoid fragmentation.

   Probe packets have the problem of burdening the network with

unnecessary packets.  And because network paths often change during
the lifetime of a TCP connection, probe packets will have to be sent
on a regular basis to detect any changes in the effective MINMTU.

Implementors sometimes mistake the TCP MSS option as a mechanism for
learning the network MINMTU.  In fact, the MSS option is only a
mechanism for learning about buffering capabilities at the two TCP
peers.  Separate provisions must be made to learn the IP MINMTU.

In this memo, we propose two new IP options that, when used in
conjunction will permit two peers to determine the MINMTU of the
paths between them.  In this scheme, one option is used to determine
the lowest MTU in a path; the second option is used to convey this
MTU back to the sender (possibly in the IP datagram containing the
transport acknowledgement to the datagram which contained the MTU
discovery option).

OPTION FORMATS

    Probe MTU Option (Number 11)

        Format

                +--------+--------+--------+--------+
                |00001011|00000100|   2 octet value |
                +--------+--------+--------+--------+

        Definition

        This option always contains the lowest MTU of all the networks
        that have been traversed so far by the datagram.

        A host that sends this option must initialize the value field to
        be the MTU of the directly-connected network.  If the host is
        multi-homed, this should be for the first-hop network.

        Each gateway that receives a datagram containing this option must
        compare the MTU field with the MTUs of the inbound and outbound
        links for the datagram.  If either MTU is lower than the value in
        the MTU field of the option, the option value should be set to the
        lower MTU.  (Note that gateways conforming to RFC-1009 may not
        know either the inbound interface or the outbound interface at the
        time that IP options are processed.  Accordingly, support for this
        option may require major gateway software changes).

        Any host receiving a datagram containing this option should
        confirm that value of the MTU field of the option is less than or
        equal to that of the inbound link, and if necessary, reduce the

MTU field value, before processing the option.

If the receiving host is not able to accept datagrams as large as
specified by the value of the MTU field of the option, then it
should reduce the MTU field to the size of the largest datagram it
can accept.

Reply MTU Option (Number 12)

Format

```
+--------+--------+--------+--------+
|00001100|00000100|   2 octet value |
+--------+--------+--------+--------+
```

Definition

This option is used to return the value learned from a Probe MTU
option to the sender of the Probe MTU option.

RELATION TO TCP MSS

Note that there are two superficially similar problems in choosing
the size of a datagram.  First, there is the restriction [2] that a
host not send a datagram larger than 576 octets unless it has
assurance that the destination is prepared to accept a larger
datagram.  Second, the sending host should not send a datagram larger
than MINMTU, in order to avoid fragmentation.  The datagram size
should normally be the minimum of these two lower bounds.

In the past, the TCP MSS option [3] has been used to avoid sending
packets larger than the destination can accept.  Unfortunately, this
is not the most general mechanism; it is not available to other
transport layers, and it cannot determine the MINMTU (because
gateways do not parse TCP options).

Because the MINMTU returned by a probe cannot be larger than the
maximum datagram size that the destination can accept, this IP option
could, in theory, supplant the use of the TCP MSS option, providing
an economy of mechanism.  (Note however, that some researchers
believe that the value of the TCP MSS is distinct from the path's
MINMTU.  The MSS is the upper limit of the data size that the peer
will accept, while the MINMTU represents a statement about the data
size supported by the path).

Note that a failure to observe the MINMTU restriction is not normally
fatal; fragmentation will occur, but this is supposed to work.  A
failure to observe the TCP MSS option, however, could be fatal

because it might lead to datagrams that can never be accepted by the
destination.  Therefore, unless and until the Probe MTU option is
universally implemented, at least by hosts, the TCP MSS option must
be used as well.

## IMPLEMENTATION APPROACHES

### Who Sends the Option

There are at least two ways to implement the MTU discovery scheme.
One method makes the transport layer responsible for MTU
discovery; the other method makes the IP layer responsible for MTU
discovery.  A host system should support one of the two schemes.

### Transport Discovery

In the transport case, the transport layer can include the Probe
MTU option in an outbound datagram.  When a datagram containing
the Probe MTU option is received, the option must be passed up to
the receiving transport layer, which should then acknowledge the
Probe with a Reply MTU option in the next return datagram.  Note
that because the options are placed on unreliable datagrams, the
original sender will have to resend Probes (possibly once per
window of data) until it receives a Reply option.  Also note that
the Reply MTU option may be returned on an IP datagram for a
different transport protocol from which it was sent (e.g., TCP
generated the probe but the Reply was received on a UDP datagram).

### IP Discovery

A better scheme is to put MTU discovery into the IP layer, using
control mechanisms in the routing cache.  Whenever an IP datagram
is sent, the IP layer checks in the routing cache to see if a
Probe or Reply MTU option needs to be inserted in the datagram.
Whenever a datagram containing either option is received, the
information in those options is placed in the routing cache.

The basic working of the protocol is somewhat complex.  We trace
it here through one round-trip.  Implementors should realize that
there may be cases where both options are contained in one
datagram.  For the purposes of this exposition, the sender of the
probe is called the Probe-Sender and the receiver, Probe-Receiver.

When the IP layer is asked to send a Probe MTU option (see the
section below on when to probe), it makes some record in the
routing cache that indicates the next IP datagram to Probe-
Receiver should contain the Probe MTU option.

When the next IP datagram to Probe-Receiver is sent, the Probe MTU
option is inserted.  The IP layer in Probe-Sender should continue
to send an occasional Probe MTU in subsequent datagrams until a
Reply MTU option is received.  It is strongly recommended that the
Probe MTU not be sent in all datagrams but only at such a rate
that, on average, one Probe MTU will be sent per round-trip
interval.  (Another way of saying this is that we would hope that
only one datagram in a transport protocol window worth of data has
the Probe MTU option set).  This mechanism might be implemented by
sending every Nth packet, or, in those implementations where the
round-trip time estimate to the destination is cached with the
route, once every estimated RTT.

When a Probe MTU option is received by Probe-Receiver, the
receiving IP should place the value of this option in the next
datagram it sends back to Probe-Sender.  The value is then
discarded.  In other words, each Probe MTU option causes the Reply
MTU option to be placed in one return datagram.

When Probe-Sender receives the Reply MTU option, it should check
the value of the option against the current MINMTU estimate in the
routing cache.  If the option value is lower, it becomes the new
MINMTU estimate.  If the option value is higher, Probe-Sender
should be more conservative about changing the MINMTU estimate.
If a route is flapping, the MINMTU may change frequently.  In such
situations, keeping the smallest MINMTU of various routes in use
is preferred.  As a result, a higher MINMTU estimate should only
be accepted after a lower estimate has been permitted to "age" a
bit.  In other words, if the probe value is higher than the
estimated MINMTU, only update the estimate if the estimate is
several seconds old or more.  Finally, whenever the Probe-Sender
receives a Reply MTU option, it should stop retransmitting probes
to Probe-Receiver.

A few additional issues complicate this discussion.

One problem is setting the default MINMTU when no Reply MTU
options have been received.  We recommend the use of the minimum
of the supported IP datagram size (576 octets) and the connected
network MTU for destinations not on the local connected network,
and the connected network MTU for hosts on the connected network.

The MINMTU information, while kept by the Internet layer, is in
fact, only of interest to the transport and higher layers.
Accordingly, the Internet layer must keep the transport layer
informed of the current value of the estimated MINMTU.
Furthermore, minimal transport protocols, such as UDP, must be
prepared to pass this information up to the transport protocol

user.

It is expected that there will be a transition period during which
some hosts support this option and some do not.  As a result,
hosts should stop sending Probe MTU options and refuse to send any
further options if it does not receive either a Probe MTU option
or Reply MTU option from the remote system after a certain number
of Probe MTU options have been sent.  In short, if Probe-Sender
has sent several probes but has gotten no indication that Probe-
Receiver supports MTU probing, then Probe-Sender should assume
that Probe-Receiver does not support probes.  (Obviously, if
Probe-Sender later receives a probe option from Probe-Receiver, it
should revise its opinion.)

Implementations should not assume that routes to the same
destination that have a different TOS have the same estimated
MINMTU.  We recommend that the MTU be probed separately for each
TOS.

## Respecting the TCP MSS

One issue concerning TCP MSS is that it is usually negotiated
assuming an IP header that contains no options.  If the transport
layer is sending maximum size segments, it may not leave space for
IP to fit the options into the datagram.  Thus, insertion of the
Probe MTU or Reply MTU option may violate the MSS restriction.
Because, unlike other IP options, the MTU options can be inserted
without the knowledge of the transport layer, the implementor must
carefully consider the implications of adding options to an IP
datagram.

One approach is to reserve 4 bytes from the MINMTU reported to the
transport layer; this will allow the IP layer to insert at least
one MTU option in every datagram (it can compare the size of the
outgoing datagram with the MINMTU stored in the route cache to see
how much room there actually is).  This is simple to implement,
but does waste a little bandwidth in the normal case.

Another approach is to provide a means for the IP layer to notify
the transport layer that space must be reserved for sending an
option; the transport layer would then make a forthcoming segment
somewhat smaller than usual.

## When a Probe Can Be Sent

A system that receives a Probe MTU option should always respond
with a Reply MTU option, unless the probe was sent to an IP or LAN
broadcast address.

A Probe MTU option should be sent in any of the following
situations:

    (1) The MINMTU for the path is not yet known;

    (2) A received datagram suffers a fragmentation re-assembly
        timeout. (This is a strong hint the path has changed;
        send a probe to the datagram's source);

    (3) An ICMP Time Exceeded/Fragmentation Reassembly Timeout is
        received (this is the only message we will get that
        indicates fragmentation occurred along the network path);

    (4) The transport layer requests it.

Implementations may also wish to periodically probe a path, even
if there is no indication that fragmentation is occurring.  This
practice is perfectly reasonable; if fragmentation and reassembly
is working perfectly, the sender may never get any indication that
the path MINMTU has changed unless a probe is sent.  We recommend,
however, that implementations send such periodic probes sparingly.
Once every few minutes, or once every few hundred datagrams is
probably sufficient.

There are also some scenarios in which the Probe MTU should not be
sent, even though there may be some indication of an MINMTU
change:

    (1) Probes should not be sent in response to the receipt of
        a probe option.  Although the fact that the remote peer
        is probing indicates that the MINMTU may have changed,
        sending a probe in response to a probe causes a continuous
        exchange of probe options.

    (2) Probes must not be sent in response to fragmented
        datagrams except when the fragmentation reassembly
        of the datagram fails.  The problem in this case is
        that the receiver has no mechanism for informing the remote
        peer that fragmentation has occurred, unless fragmentation
        reassembly fails (in which case an ICMP message is sent).
        Thus, a peer may use the wrong MTU for some time before
        discovering a problem.  If we probe on fragmented
        datagrams, we may probe, unnecessarily, for some time
        until the remote peer corrects its MTU.

    (3) For compatibility with hosts that do not implement the
        option, no Probe MTU Option should be sent more than
        ten times without receiving a Reply MTU Option or a

Probe MTU Option from the remote peer.  Peers which
ignore probes and do not send probes must be treated
as not supporting probes.

(4) Probes should not be sent to an IP or LAN broadcast
address.

(5) We recommend that Probe MTUs not be sent to other hosts
on the directly-connected network, but that this feature
be configurable.  There are situations (for example, when
Proxy ARP is in use) where it may be difficult to determine
which systems are on the directly-connected network.  In
this case, probing may make sense.

SAMPLE IMPLEMENTATION SKETCH

We present here a somewhat more concrete description of how an IP-
layer implementation of MTU probing might be designed.

First, the routing cache entries are enhanced to store seven
additional values:

MINMTU: The current MINMTU of the path.

ProbeRetry: A timestamp indicating when the next probe
should be sent.

LastDecreased: A timestamp showing when the MTU was
last decreased.

ProbeReply: A bit indicating a Reply MTU option should be
sent.

ReplyMTU: The value to go in the Reply MTU option.

SupportsProbes: A bit indicating that the remote peer
can deal with probes (always defaults to
1=true).

ConsecutiveProbes: The number of probes sent without
the receipt of a Probe MTU or Reply
MTU option.

There are also several configuration parameters; these should be
configurable by appropriate network management software; the values
we suggest are "reasonable":

Default_MINMTU: The default value for the MINMTU field of the

                          routing cache entry, to be used when the real
                          MINMTU is unknown.  Recommended value: 576.

      Max_ConsecutiveProbs: The maximum number of probes to send
                          before assuming that the destination does
                          not support the probe option.
                          Recommended value: 10.

      ProbeRetryTime: The time (in seconds) to wait before retrying
                          an unanswered probe.  Recommended value:
                          60 seconds, or 2*RTT if the the RTT is available
                          to the IP layer.

      ReprobeInterval: The time to wait before sending a probe after
                          receiving a successful Reply MTU, in order to
                          detect increases in the route's MINMTU.
                          Recommended value: 5 times the ProbeRetryTime.

      IncreaseInterval: The time to wait before increasing the MINMTU
                          after the value has been decreased, to prevent
                          flapping.  Recommended value: same as
                          ProbeRetryTime.

   When a new route is entered into the routing cache, the initial
   values should be set as follows:

      MINMTU = Default_MINMTU

      ProbeRetry = Current Time

      LastDecreased = Current Time - IncreaseInterval

      ProbeReply = false

      SupportsProbes = true

      ConsecutiveProbes = 0

   This initialization is done before attempting to send the first
   packet along this route, so that the first packet will contain a
   Probe MTU option.

   Whenever the IP layer sends a datagram on this route it checks the
   SupportsProbes bit to see if the remote system supports probing.  If
   the SupportsProbes bit is set, and the timestamp in ProbeRetry is
   less than or equal to the current time, a Probe option should be sent
   in the datagram, and the ProbeRetry field incremented by
   ProbeRetryTime.

Whether or not the Probe MTU option is sent in a datagram, if the
ProbeReply bit is set, then a Reply MTU option with the value of the
ReplyMTU field is placed in the outbound datagram.  The ProbeReply
bit is then cleared.

Every time a Probe option is sent, the ConsecutiveProbes value should
be incremented.  If this value reaches Max_ConsecutiveProbes, the
SupportsProbe bit should be cleared.

When an IP datagram containing the Probe MTU option is received, the
receiving IP sets the ReplyMTU to the Probe MTU option value and sets
the ProbeReply bit in its outbound route to the source of the
datagram.  The SupportsProbe bit is set, and the ConsecutiveProbes
value is reset to 0.

If an IP datagram containing the Reply MTU option is received, the IP
layer must locate the routing cache entry corresponding to the source
of the Reply MTU option; if no such entry exists, a new one (with
default values) should be created.  The SupportsProbe bit is set, and
the ConsecutiveProbes value is reset to 0.  The ProbeRetry field is
set to the current time plus ReprobeInterval.

Four cases are possible when a Reply MTU option is received:

    (1) The Reply MTU option value is less than the current
        MINMTU: the MINMTU field is set to the new value, and
        the LastDecreased field is set to the current time.

    (2) The Reply MTU option value is greater than the
        current MINMTU and the LastDecreased field plus
        IncreaseInterval is less than the current time: set the
        ProbeRetry field to LastDecreased plus IncreaseInterval,
        but do not change MINMTU.

    (3) The Reply MTU option value is greater than the
        current MINMTU and the LastDecreased field plus
        IncreaseInterval is greater than the current time: set
        the MINMTU field to the new value.

    (4) The Reply MTU option value is equal to the current
        MINMTU: do nothing more.

Whenever the MTU field is changed, the transport layer should be
notified, either by an upcall or by a change in a shared variable
(which may be accessed from the transport layer by a downcall).

If a fragmentation reassembly timeout occurs, if an ICMP Time
Exceeded/Fragmentation Reassembly Timeout is received, or if the IP

layer is asked to send a probe by a higher layer, the ProbeRetry
field for the appropriate routing cache entry is set to the current
time.  This will cause a Probe option to be sent with the next
datagram (unless the SupportsProbe bit is turned off).

MANAGEMENT PARAMETERS

   We suggest that the following parameters be made available to local
   applications and remote network management systems:

      (1) The number of probe retries to be made before determining
          a system is down.  The value of 10 is certain to be wrong
          in some situations.

      (2) The frequency with which probes are sent.  Systems may
          find that more or less frequent probing is more cost
          effective.

      (3) The default MINMTU used to initialize routes.

      (4) Applications should have the ability to force a probe
          on a particular route.  There are cases where a probe
          needs to be sent but the sender doesn't know it.  An
          operator must be able to cause a probe in such situations.
          Furthermore, it may be useful for applications to "ping"
          for the MTU.

REFERENCES

   [1]   Kent, C. and J. Mogul, "Fragmentation Considered
         Harmful", Proc. ACM SIGCOMM '87, Stowe, VT, August 1987.

   [2]   Postel, J., Ed., "Internet Protocol", RFC-791,
         USC/Information Sciences Institute, Marina del Rey, CA,
         September 1981.

   [3]   Postel, J., Ed., "Transmission Control Protocol", RFC-793,
         USC/Information Sciences Institute, Marina del Rey, CA,
         September 1981.

   [4]   Postel, J., "The TCP Maximum Segment Size and Related Topics",
         RFC-879, USC/Information Sciences Institute, Marina del Rey,
         CA, November 1983.