Danny Cohen

February 1979

IEN: 80

INTERNET DATAGRAM PROTOCOL

Version 4

February 1979

prepared for

Defense Advanced Research Projects Agency
Information Processing Techniques Office
1400 Wilson Boulevard
Arlington, Virginia 22209

by

Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, California 90291

TABLE OF CONTENTS

PREFACE

This document describes the Internet Datagram Protocol. There have been two previous editions of this specification, and the present text draws heavily from them. There have been many contributors to this document both in terms of concepts and in terms of text.

Jon Postel

Editor

< INC-PROJECT, IN-JAN-79.NLS.21, >, 17-Feb-79 00:27 JBP ;;;;

February 1979
IEN: 80
Replaces: IENs 54, 44, 41, 28, 26

Internet Datagram Protocol Specification
Version 4

1. INTRODUCTION

1.1. Motivation

The Internet Protocol is designed for use in interconnected systems of packet-switched computer communication networks. Such a system has been called a "catenet" [1]. The internet protocol provides for transmitting blocks of data called datagrams from sources to destinations, where sources and destinations are hosts identified by fixed length addresses. The internet protocol also provides for fragmentation and reassembly of long datagrams, if necessary, for transmission through "small packet" networks.

1.2. Scope

The internet protocol is specifically limited in scope to provide the functions necessary to deliver a package of bits (an internet datagram) from a source to a destination over an interconnected system of networks. There are no mechanisms to promote data reliability, flow control, sequencing, or other services commonly found in host-to-host protocols.

1.3. Interfaces

This protocol is called on by host-to-host protocols in an internet environment. This protocol calls on local network protocols to carry the internet packet to the next gateway or destination host.

For example, a TCP module would call on the internet module to take a TCP segment (including the TCP header and user data) as the data portion of an internet datagram. The TCP module would provide the addresses and other parameters in the internet header to the internet module as arguments of the call. The internet module would then create an internet datagram and call on the local network interface to transmit the internet datagram.

In the ARPANET case, for example, the internet module would call on a local net module which would add the 1822 leader [2] to the internet datagram creating an ARPANET message to transmit to the IMP.

1.4. Operation

The internet protocol implements two basic functions: addressing and fragmentation.

The internet modules use the addresses carried in the internet header to transmit the internet packets toward their destinations. The selection of a path for transmission is called routing.

The internet modules use fields in the internet header to fragment and reassemble internet packets when necessary for transmission through "small packet" networks.

The model of operation is that an internet module resides in each host engaged in internet communication and in each gateway that interconnects networks. These modules share common rules for interpreting address fields and for fragmenting and assembling internet packets. In addition, these modules (especially in gateways) may have procedures for making routing decisions and other functions.

The internet protocol uses four key mechanisms in providing its service: Type of Service, Time to Live, Options, and Header Checksum.

The Type of Service is used to indicate the quality of the service desired; this may be thought of as selecting among Interactive, Bulk, or Real Time, for example. This type of service indication is to be used by gateways to select the actual transmission parameters for a particular network, the network to be used for the next hop, or the next gateway when routing an internet packet.

The Time to Live is an indication of the lifetime of an internet packet. It is set by the sender of the packet and reduced at the points along the route where it is processed. If the time to live reaches zero before the internet packet reaches its destination, the internet packet is destroyed. The time to live can be thought of as a self destruct time limit.

The Options provide for control functions needed or useful in some situations but unnecessary for the most common communications. The options include provisions for timestamps, error reports, and special routing.

The Header Checksum provides a verification that the information used in processing internet packets has been transmitted correctly. The data may contain errors. If the header checksum fails, the internet packet is discarded at once, by the entity which detects the error.

The internet protocol does not provide a reliable communication facility. There are no acknowledgments either end-to-end or hop-by-hop. There is no error control for data, only a header checksum. There are no retransmissions. There is no flow control.

The internet protocol treats each internet datagram as an independent entity unrelated to any other internet datagram. There are no connections or logical circuits (virtual or otherwise).

2. OVERVIEW

2.1. Relation to Other Protocols

The following diagram illustrates the place of the internet protocol in the protocol hierarchy:

```
    +------+ +-----+ +-----+     +-----+
    !Telnet! ! FTP ! !Voice! ... !     !
    +------+ +-----+ +-----+     +-----+
       !  !           !             !
      +-----+      +-----+       +-----+
      ! TCP !      ! RTP !  ...  !     !
      +-----+      +-----+       +-----+
         !            !             !
      +---------------------------------+
      !        Internet Protocol        !
      +---------------------------------+

        +----------------------------+
        !   Local Network Protocol   !
        +----------------------------+
                      !
```

Protocol Relationships

Figure 1.

Internet protocol interfaces on one side to the higher level host-to-host protocols and on the other side to the local network protocol.

2.2. Function Description

The function or purpose of Internet Datagram Protocol is to move datagrams through an interconnected set of networks. This is done by passing the datagrams from one internet module to another until the destination is reached. The internet modules reside in hosts and gateways in the internetwork system. The datagrams are routed from one internet module to another through individual networks based on the interpretation of an internet address. Thus, one important mechanism of the internet protocol is the internet address.

In the routing of messages from one internet module to another, datagrams may need to traverse a network whose maximum packet size is smaller than the size of the datagram. To overcome this difficulty, a fragmentation mechanism is provided in the internet protocol.

Addressing

A distinction is made between names, addresses, and routes [3]. A name indicates what we seek. An address indicates where it is. A route indicates how to get there. The internet protocol deals only with addresses. It is the task of higher level (i.e., host-to-host or application) protocols to make the mapping from names to addresses. It is the task of lower level (i.e., local net or gateways) procedures to make the mapping from addresses to routes.

Addresses are fixed length of four octets (32 bits). An address begins with a one octet network number, followed by a three octet local address. This three octet field is called the "rest" field.

Care must be taken in mapping internet addresses to local net addresses; a single physical host must be able to act as if it were several distinct hosts to the extent of using several distinct internet addresses.

Fragmentation

Fragmentation of an internet datagram may be necessary when it originates in a local net that allows a large packet size and must traverse a local net that limits packets to a smaller size to reach its destination.

An internet datagram can be marked "don't fragment." Any internet datagram so marked is not to be internet fragmented under any circumstances. Intranet fragmentation may be used, however, that is fragmentation, transmission and reassembly across a local network which is invisible to the internet protocol module. If internet datagram marked don't fragment cannot be delivered to its destination without fragmenting it, it is to be discarded instead.

The internet protocol fragmentation procedure utilizes information in three fields of the internet header: the identification, the more-fragments-flag, and the fragment offset.

The originating protocol module of an internet datagram sets the identification field to a value that must be unique for that source-destination pair and protocol for the time the datagram will be active in the internetwork system. The originating protocol module of a complete datagram sets the more-fragments-flag to zero and the fragment offset to zero.

To fragment a long internet packet, an internet protocol module (for example, in a gateway), creates two new internet packets and copies the contents of the internet header fields from the long packet into both new internet headers. The data of the long packet is divided into two portions on a 8 octet (64 bit) boundary (the second portion might not be an integral multiple of 8 octets, but the first must be). Call the number of 8 octet blocks in the first portion NFB (for Number of Fragment Blocks). The first portion of the data is placed in the first new internet packet, and the total length field is set to the length of the first packet. The more-fragments-flag is set to one. The second portion of the data is placed in the second new internet packet, and the total length field is set to the length of the second packet. The more-fragments-flag carries the same value as the long packet. The fragment offset field of the second new internet packet is set to the value of that field in the long packet plus NFB.

This procedure can be generalized for an n-way split, rather than the two-way split described.

To assemble the fragments of an internet datagram, an internet protocol module (for example at a destination host) combines internet packets that all have the same value for the four fields: identification, source, destination, and protocol. The combination is done by placing the data portion of each fragment in the relative position indicated by the fragment offset in that fragment's internet header. The first fragment will have the fragment offset zero, and the last fragment will have the more-fragments-flag reset to zero.

3. SPECIFICATION

3.1. Internetwork Header Format

A summary of the contents of the internetwork header follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!Version!  IHL  !Type of Service!          Total Length         !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!         Identification        !Flags!      Fragment Offset    !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!  Time to Live !    Protocol   !         Header Checksum       !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                       Source Address                          !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                    Destination Address                        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                    Options                    !    Padding    !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Example Internet Packet Header

Figure 2.

Note that each tick mark represents one bit position.

Version: 4 bits

There is a Version field which indicates the "shape," or format, of the internet portion. This is version 4.

IHL: 4 bits

Internet Header Length is the length of the internet header in 32 bit words, and thus points to the beginning of the data. Note that the minimum value for a correct header is 5.

Type of Service: 8 bits

Type of service.

```
Bits 0-1:  Priority.
Bit    2:  Stream or Datagram.
Bits 3-4:  Reliability.
Bit    5:  Speed over Reliability.
Bits 6-7:  Speed.

   0     1     2     3     4     5     6     7
+-----+-----+-----+-----+-----+-----+-----+-----+
!           !     !           !     !           !
!  PRIORITY !STRM !RELIABILITY! S/R !   SPEED   !
!           !     !           !     !           !
+-----+-----+-----+-----+-----+-----+-----+-----+

PRIORITY     STRM      RELIABILITY  S/R      SPEED
11-highest  1-STREAM  11-highest   1-speed  11-highest
10-higher   0-DTGRM   10-higher    0-rlblt  10-higher
01-lower              01-lower              01-lower
00-lowest             00-lowest             00-lowest
```

The type of service is used to specify the treatment of the datagram during its transmission through the internetwork system. In the discussion (section 3.4) below, a chart shows the relationship of the internet type of service to the actual service provided on the ARPANET, the SATNET, and the PRNET.

Total Length: 16 bits

Total Length is the length of the packet, measured in octets, including internet header and data. This field allows the length of a datagram to be up to 65,535 octets. Such long datagrams are impractical for most hosts and networks. All hosts must be prepared to accept datagrams of up to 576 octets (whether they arrive whole in fragments). It is recommended that hosts only send datagrams larger than 576 octets if they have assurance that the destination is prepared to accept the larger datagrams.

Identification: 16 bits

An identifying value assigned by the sender to aid in assembling the fragments of a datagram.

Flags: 3 bits

Various Control Flags.

Bit 0: reserved, must be zero
Bit 1: Don't Fragment This Segment (DF).
Bit 2: More Fragments Flag (MF).

```
  0   1   2
+---+---+---+
!   ! D ! M !
! 0 ! F ! F !
+---+---+---+
```

Fragment Offset: 13 bits

This field indicates where in the segment this fragment belongs. The fragment offset is measured in units of 8 octets (64 bits).

Time to Live: 8 bits

This field indicates the maximum time the datagram is allowed to remain the internetwork system. If this field contains the value zero then the datagram should be destroyed. This field is modified in internet header processing. The time is measured in units of seconds.

Protocol: 8 bits

This field indicates the next level protocol used in the data portion of the internet datagram. The values for various protocols are specified in reference [4].

Header Checksum: 16 bits

A checksum on the header only. Since some header fields may change this is recomputed and verified at each point that the internet header is processed.

The checksum algorithm is:

The checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words in the header. For purposes of computing the checksum, the value of the checksum field is zero.

This checksum is provisional and may be replaced by a CRC procedure, as experience dictates.

Source Address: 32 bits

The source address. The first octet is the Source Network, and the following three octets are the Source Local Address.

Destination Address: 32 bits

The destination address. The first octet is the Destination Network, and the following three octets are the Destination Local Address.

Options: variable

The option field is variable in length. The format is an option-type octet, an option-length octet, and the actual option-data octets. There are two special case options which have only the option-type octet.

The option-length octet, which follows, includes the option-type octet and the option-length octet in the octet count of the option length.

The option-type octet can be viewed as having 3 fields:

1 bit   reserved, must be zero
2 bits option class,
5 bits option number.

The option classes are:

0 = control
1 = internet error
2 = experimental debugging and measurement
3 = reserved for future use

The following internet options are defined:

CLASS	NUMBER	LENGTH	DESCRIPTION
0	0	-	End of Option list. This option occupies only 1 octet; it has no length octet.
0	1	-	No Operation. This option occupies only 1 octet; it has no length octet.
0	2	4	S/P/T. Used to carry Security, Precedence, and user group (TCC) information compatible with AUTODIN II requirements.
0	3	var.	Source Routing. Used to route the internet packet based on information supplied by the source.
1	1	var.	General Error Report. Used to report errors in internet packet processing.
2	4	var.	Internet Timestamp. Used to accumulate timestamping information during internet transit. The length field is variable and may change as the internet packet traverses the networks and gateways of the internet system.
2	5	var.	Satellite Timestamp. Used as above for special satellite network testing.

Specific Option Definitions

End of Option List

```
+--------+
!00000000!
+--------+
  Type=0
```

This option indicates the end of the option list. This might not coincide with the end of the internet header according to the internet header length. This is used at the end of all options, not the end of each option, and need only be used if the end of the options would not otherwise coincide with the end of the internet header.

No Operation

```
+--------+
!00000001!
+--------+
  Type=1
```

This option may be used between options, for example, to align the beginning of a subsequent option on a 32 bit boundary.

S/P/T

This option provides a way for AUTODIN II hosts to send security, precedence, and TCC (closed user groups) parameters through networks whose transport leader does not contain fields for this information. The format for this option is as follows:

```
+--------+--------+--------+--------+
!00000010!00000100!Prec!Sec !  TCC   !
+--------+--------+--------+--------+
  Type=2  Length=4
```

Precedence: 4 bits

Specifies one of 16 levels of precedence

Security: 4 bits

Specifies one of 16 levels of security

Transmission Control Code: 8 bits

Provides a means to compartmentalize traffic and define controlled communities of interest among subscribers.

This option might be used between hosts on the AUTODIN II network and other networks, such as the EDN at DCEC.

Source Routing

```
+--------+--------+--------+--------+--------//--------+
!00000011! length ! pointer!   source route            !
+--------+--------+--------+--------+--------//--------+
  Type=3
```

The source routing option provides a means for the source of an internet datagram to supply routing information to be used by the gateways in forwarding the datagram to the destination.

A source route is composed of a series of internet addresses. The pointer is initially zero, which indicates the first octet of the source route. The segment is routed to address in the source route indicated by the pointer. At that internet module the pointer is advanced to the next address in the source route. This routing and pointer advancing is repeated until the source address is exhausted. At that point the destination may have been reached, if not, the protocol module must attempt to route the packet to the destination in the destination address field by the ordinary routing procedure.

General Error Report

```
+--------+--------+--------+--------+--------//--------+
!00100001! length !err code!   id   !                  !
+--------+--------+--------+--------+--------//--------+
 Type=33
```

The general error report is used to report an error detected in processing an internet packet to the originator of that packet. The "err code" indicates the type of error detected and the "id" is copied from the identification field of the packet in error, additional octets of error information may be present depending on the err code.

ERR CODE:

0 - Undetermined Error, used when no information is available about the type of error or the error does not fit any defined class. Following the id should be as much of the datagram as fits in the option space.

No err codes have been defined for specific classes as yet.

Internet Timestamp

```
+--------+--------+--------+--------+--------//--------+
!01000100! length !   ?    !   ?    !        ?         !
+--------+--------+--------+--------+--------//--------+
 Type=68
```

No information is available on the specific format of Timestamps.

Satellite Timestamp

```
+--------+--------+--------+--------+--------//--------+
!01000101! length !    ?   !    ?   !        ?         !
+--------+--------+--------+--------+--------//--------+
  Type=69
```

No information is available on the specific format of Timestamps.

Padding: variable

The internet header padding is used to ensure that the internet header ends on a 32 bit boundary. The padding is zero.

3.2. Discussion

The basic internet service is datagram oriented and provides for the fragmentation of packets at gateways, with reassembly taking place at the destination internet protocol module in the destination host. Of course, fragmentation and reassembly of packets within a network or by private agreement between the gateways of a network is also allowed since this is transparent to the internet protocols and the higher-level protocols. This transparent type of fragmentation and reassembly is termed "network-dependent" (or intranet) fragmentation and is not discussed further here.

Internet addresses distinguish sources and destinations to the host level and provide a protocol identification field as well. It is assumed that each protocol will provide for whatever multiplexing is necessary within a host.

Addressing

The 8 bit network number, which is the first octet of the variable length address, has a value as specified in reference [4].

The 24 bit local address, assigned by the local network, should allow for a single physical host to act as several distinct internet hosts. That is, there should be mapping between internet host addresses and network/host interfaces that allows several internet addresses to correspond to one interface.

Fragmentation and Reassembly.

The internet identification field, (ID), is used together with the source and destination address, and the protocol fields, to identify packet fragments for reassembly.

The More Fragments flag bit (MF) is set if the packet is not the last fragment. The Fragment Offset field identifies the fragment number, relative to the beginning of the original unfragmented packet. Fragments are numbered in units of 8 octets. The fragmentation strategy is designed so than an unfragmented packet has all zero fragmentation information (MF = 0, fragment offset = 0). If an internet packet is fragmented, its data portion must be broken on 8 octet boundaries.

This format allows 2**13 = 8192 fragments of 8 octets each for a total of 65,536 octets. Note that this is consistent with the datagram total length field.

When fragmentation occurs, options are generally not copied, but remain with the first fragment. Some options, such as source routing, must be copied, however. For concreteness, an example of a fragmented packet is illustrated in example 2 below.

Every internet module must be able to forward a datagram of 68 octets without further fragmentation. This is because an internet header may be up to 60 octets, and the minimum fragment is 8 octets.

Every internet destination must be able to receive a datagram of 576 octets either in one piece or in fragments to be reassembled.

The fields which may be affected by fragmentation include:

(1) options field
(2) more fragments flag
(3) fragment offset
(4) internet header length field
(5) total length field
(6) header checksum

If the Don't Fragment flag (DF) bit is set then internet fragmentation of this packet is NOT permitted, although it may be discarded. This can be used to prohibit fragmentation in cases where the receiving host does not have sufficient resources to reassemble internet fragments.

The choice of this Identifier for a datagram is based on the need to provide a way to uniquely identify the fragments of a particular datagram. The protocol module assembling fragments judges fragments to belong to the same datagram if they have the same source, destination, protocol, and Identifier. Thus, the sender must choose the Identifier to be unique for this source, destination pair and protocol for the time the datagram (or any fragment of it) could be alive in the internetwork.

It seems then that a sending protocol module needs to keep a table of Identifiers, one entry for each destination it has communicated with in the last maximum packet lifetime for the internetwork.

However, since the Identifier field allows 65,536 different values, some host may be able to simply use unique identifiers independent of destination.

It is appropriate for some higher level protocols to choose the identifier. For example, TCP protocol modules may retransmit an identical TCP segment, and the probability for correct reception would be enhanced if the retransmission carried the same identifier as the original transmission since fragments of either datagram could be used to construct a correct TCP segment.

Type of Service

The type of service (TOS) is for internet service quality selection. The type of service is specified along the parameters priority, reliability, and speed. A further concern is the possibility of efficient handling of streams of datagrams.

Priority. An independent measure of the importance of this datagram.

Stream or Datagram. Indicates if there will be other datagrams from this source to this destination at regular frequent intervals justifying the maintenance of stream processing information.

Reliability. A measure of the level of effort desired to ensure delivery of this datagram.

Speed over Reliability. Indicates the relative importance of speed and reliability when a conflict arises in meeting the pair of requests.

Speed. A measure of the importance of prompt delivery of this datagram.

The following chart presents the recommended mappings from the internet protocol type of service into the service parameters actually available on the ARPANET, the SATNET, and the PRNET:

Application	INTERNET	ARPANET	PRNET	SATNET
TELNET on TCP	P:stream S:fast R:normal P:speed	T: 3 S: S	R: ptp A: no	T: block D: min H: inf R: no
FTP on TCP	P:stream S:normal R:normal P:reliable	T: 0 S: M	R: ptp A: no	T: block D: normal H: inf R: no
interactive narrow band speech	P:stream* S:asap R:least P:speed	T: 3 S: S	R: ptp A: no	T: stream D: min H: short R: no
datagram	P:datagram S:fast R:normal P:speed	T: 3 or 0 S: S or M	R:station A: no	T: block D: min H: short R: no

key: P=package, S=speed, R=reliability, P=preference, *=requires stream set up; T=type, S=size; R=route, A=ack; T=type, D=delay, H=holding time, R=reliability

Time to Live

The time to live is set by the sender to the maximum time the datagram is allowed to be in the internetwork system. If the datagram is in the internetwork system longer than the time to live, then the datagram should be destroyed. This field should be decreased at each point that the internet header is processed to reflect the time spent processing the datagram. Even if no local information is available on the time actually spent, the field should be decremented. The time is measured in units of seconds (i.e. the value 1 means one second). Thus, the maximum time to live is 255 seconds or 4.25 minutes.

Options

The options are just that, optional. That is, the presence or absence of an option is the choice of the sender, but each internet module must understand how to process every option.

Checksum

The internet header checksum is recomputed if the internet header is changed owing to additions or changes to internet options or due to fragmentation. This checksum at the internet level will protect the internet header fields from transmission errors.

Changes to the source, destination, protocol, or identification fields or the data of the datagram are not permitted since these fields may be covered by a higher level end-to-end checksum.

3.3. Examples & Scenarios

Example 1:

This is an example of the minimal data carrying internet datagram:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!Ver= 4 !IHL= 5 !Type of Service!     Total Length = 21         !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!      Identification = 111     !Flg=0!  Fragment Offset = 0    !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!   Time = 123  !  Protocol = 1 !       header checksum         !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                        source address                         !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                      destination address                      !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!     data      !
+-+-+-+-+-+-+-+-+
```

Example Internet Datagram

Figure 3.

Note that each tick mark represents one bit position.

This is a internet datagram in version 4 of internet protocol; the internet header consists of five 32 bit words, and the total length of the datagram is 21 octets. This packet is a complete datagram (not a fragment).

Example 2:

In this example, we show first a moderate size internet datagram (552 data octets), then two internet fragments that might result from the fragmentation of this datagram.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!Ver= 4 !IHL= 5 !Type of Service!     Total Length = 552        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!    Identification = 111      !Flg=0!   Fragment Offset = 0    !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!  Time = 123   ! Protocol = 6  !       header checksum         !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                        source address                         !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                      destination address                      !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
\                                                               \
\                                                               \
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!             data              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Example Internet Datagram

Figure 4.

Now the first fragment that results from splitting the datagram after 256 data octets.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!Ver= 4 !IHL= 5 !Type of Service!       Total Length = 276      !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!     Identification = 111      !Flg=1!    Fragment Offset = 0  !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!  Time = 119   ! Protocol = 6  !        Header Checksum        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                        source address                         !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                      destination address                      !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
\                                                               \
\                                                               \
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Example Internet Fragment

Figure 5.

And the second fragment.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!Ver= 4 !IHL= 5 !Type of Service!       Total Length = 296      !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!     Identification = 111      !Flg=0!  Fragment Offset  =  32 !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!   Time = 119  ! Protocol = 6  !        Header Checksum        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                        source address                         !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                      destination address                      !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
\                                                               \
\                                                               \
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!            data               !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Example Internet Fragment

Figure 6.

Example 3:

Here, we show an example of a datagram containing options.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!Ver= 4 !IHL= 8 !Type of Service!       Total Length = 576      !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!       Identification = 111    !Flg=0!     Fragment Offset = 0 !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!   Time = 123  !  Protocol = 6 !       Header Checksum         !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                        source address                         !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                      destination address                      !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
! Opt. Code = x ! Opt.  Len.= 3 ! option value  ! Opt. Code = x !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
! Opt. Len. = 4 !           option value        ! Opt. Code = 1 !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
! Opt. Code = y ! Opt. Len. = 3 !  option value ! Opt. Code = 0 !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
\                                                               \
\                                                               \
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Example Internet Datagram

Figure 7.

3.4. Interfaces

Internet protocol interfaces on one side to the local network and on the other side to either a higher level protocol or an application program. In the following, the higher level protocol or application program will be called "the user" since it is using the internet module. Since internet protocol is a datagram protocol, there is no memory or state maintained between datagram transmissions, and each call on the internet protocol module by the user supplies all the necessary information.

For example, the following two calls satisfy the requirements for the user to internet protocol module communication:

```
SEND (dest, TOS, TTL, BufPTR, len, Id, DF, options => result)

  where:

    dest = destination address
    TOS = type of service
    TTL = time to live
    BufPTR = buffer pointer
    len = length of buffer
    Id  = Identifier
    DF = Don't Fragment
    options = option data
    result = response
      OK = sent ok
      Error = error in arguments or local network error

RECV (BufPTR => result, source, dest, prot, TOS, len)

  where:

    BufPTR = buffer pointer
    result = response
      OK = received ok
      Error = error in arguments
    source = source address
    dest = destination address
    prot = protocol
    TOS = type of service
    len = length of buffer
```

When the user sends a datagram, it executes the SEND call supplying all the arguments. The internet protocol module, on receiving this call, checks the arguments and prepares and sends the message. If the arguments are good and the datagram is accepted by the local network, the call returns successfully. If either the arguments are bad, or the datagram is not accepted by the local network, the call returns unsuccessfully. On unsuccessful returns, a reasonable report should be made as to the cause of the problem, but the details of such reports are up to individual implementations.

When a datagram arrives at the internet protocol module from the local network, either there is a pending RECV call from user addressed or there is not. In the first case, the pending call is satisfied by passing the information from the datagram to the user. In the second case, the user addressed is notified of a pending datagram. If the user addressed does not exist, an error datagram is returned to the sender, and the data is discarded.

The notification of a user may be via a pseudo interrupt or similar mechanism, as appropriate in the particular operating system environment of the implementation.

A user's RECV call may then either be immediately satisfied by a pending datagram, or the call may be pending until a datagram arrives.

GLOSSARY

1822

BBN Report 1822, "The Specification of the Interconnection of a Host and an IMP". The specification of interface between a host and the ARPANET.

ARPANET message

The unit of transmission between a host and an IMP in the ARPANET. The maximum size is about 1012 octets (8096 bits).

ARPANET packet

A unit of transmission used internally in the ARPANET between IMPs. The maximum size is about 126 octets (1008 bits).

datagram

A logical unit of data, in particular an internet datagram is the unit of data transfered between the internet module and a higher level module.

Destination

The destination address, an internet header field.

DF

The Don't Fragment bit carried in the type of service field.

Flags

An internet header field carrying various control flags.

fragment

A portion of a logical unit of data, in particular an internet fragment is a portion of an internet datagram.

Fragment Offset

This internet header field indicates where in the internet datagram this fragment belongs.

header

Control information at the beginning of a message, segment, datagram, packet or block of data.

Identification

An internet header field identifying value assigned by the sender to aid in assembling the fragments of a datagram.

IHL

The internet header field Internet Header Length is the length of the internet header measured in 32 bit words.

IMP

The Interface Message Processor, the packet switch of the ARPANET.

Internet Address

A four octet (32 bit) source or destination address consisting of a Network field and a Local Address field.

internet fragment

A portion of the data of an internet datagram with an internet header.

internet packet

Either an internet datagram or an internet fragment.

internet datagram

The unit of data exchanged between a pair of internet modules (includes the internet header).

leader

Control information at the beginning of a message or block of data. In particular, in the ARPANET, the control information on an ARPANET message at the host-IMP interface.

Local Address

The address of a host within a network. The actual mapping of an internet local address on to the host addresses in a network is quite general, allowing for many to one mappings.

MF

The More-Fragments Flag carried in the internet header Flags field.

module

An implementation, usually in software, of a protocol or other procedure.

more-fragments-flag

A flag indicating whether or not this internet packet contains the end of an internet datagram, carried in the internet header Flags field.

NFB

The Number of Fragment Blocks in a portion of an internet packet. That is, the length of a portion of data measured in 8 octet units.

octet

An eight bit byte.

Options

The internet header Options field may contain several options, and each option may be several octets in length. The options are used primarily in testing situations, for example to carry timestamps.

packet

A package of data with a header which may or may not be logically complete. More often a physical packaging than a logical packaging of data.

Padding

The internet header Padding field is used to ensure that the data begins on 32 bit word boundary. The padding is zero.

Protocol

The next higher level protocol identifier, an internet header field.

Rest

The 3 octet (24 bit) local address portion of an Internet Address.

RTP

Real Time Protocol: A host-to-host protocol for communication of time critical information.

Source

The source address, an internet header field.

TCP

Transmission Control Protocol: A host-to-host protocol for reliable communication in internetwork environments.

Total Length

The internet header field Total Length is the length of the packet in octets including internet header and data.

Type of Service

An internet header field which indicates the type (or quality) of service for this internet packet.

Version

The Version field indicates the format of the internet header.

XNET

A cross-net debugging protocol.

REFERENCES

[1] Cerf, V., "The Catenet Model for Internetworking," Information Processing Techniques Office, Defense Advanced Research Projects Agency, IEN 48, July 1978.

[2] Bolt Beranek and Newman, "Specification for the Interconnection of a Host and an IMP," BBN Technical Report 1822, May 1978 (Revised).

[3] Shoch, J., "A Note On Inter-Network Naming, Addressing, and Routing," Xerox Palo Alto Research Center, IEN 19, January 1978.

[4] Postel, J., "Assigned Numbers," RFC 750, NIC 45500, 26 September 1978.