IEN: 54
Section: 2.3.2.1

INTERNETWORK PROTOCOL SPECIFICATION

Version 4

Jonathan B. Postel

September 1978

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

(213) 822-1511

TABLE OF CONTENTS

PREFACE

This is the revised specification of the Internet Protocol (version 4). Many people have contributed the concepts and ideas embodied in this specification, credit should go to at least the following: Vint Cerf, Danny Cohen, Dave Clark, Dick Watson, Ray Tomlinson, John Shoch, and the whole Internet Working Group.

September 1978
IEN: 54
Section: 2.3.2.1
Replaces: IENs 44, 41, 28, 26

Internetwork Protocol Specification

Version 4

1. INTRODUCTION

The Internet Protocol is designed for use in interconnected systems of packet-switched computer communication networks. The internet protocol provides for transmitting segments of data 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 segments, if necessary, for transmission through "small packet" networks.

1.1. History

This protocol has been developed as one result of the ARPA sponsored internetwork experiments program. The history until January 1978 is the history of the host-to-host protocol TCP.

The first publication of the ideas on which TCP is based was a paper in the IEEE Transactions on Communications by Cerf and Kahn in 1974 [1]. Later that year a protocol specification was published by a group led by Cerf at Stanford University [2]. A second specification was prepared in 1976 by a group led by Postel at SRI for the Defense Communication Agency for the AUTODIN II network [3]. In 1977 Cerf, at ARPA, prepared a substantial revision of the TCP specification [4]. Recently Postel revised Cerf's revision to distinguish the internet aspects from the host-to-host aspects [5].

Since January 1978 ideas about the internet protocol have continued to evolve and two documents were circulated by Postel [6] and Cerf [7]. The present specification draws on both of these and the discussions of the Internetwork Working Group. A brief memo on a revision of TCP in light of these developments was circulated by Cerf [8]. In June 1978, a draft edition of this document was circulated [9].

1.2. Scope

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

The protocol is intended to be utilized in gateways that interconnect sets of networks.

1.3. Documentation

No documentation beyond that cited in the History Section (1.1) above is known. Those documents do provide some background, as do a series of working notes circulated in the ARPA research community. These notes are called Internetwork Experiment Notes (or IENs) and are collected into an Internet Notebook.

1.4. 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 segment. 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 segment and call on the local network interface to transmit the internet segment.

In the ARPANET case, for example, the internet module would call on a local net module which would add the 1822 leader [10] to the internet segment creating an ARPANET message to transmit to the IMP.

1.5. 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 transmision is called routing. Routing is not a topic discussed by the internet protocol (at least not this version of it).

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 when routing an internet packet through a particular network.

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.

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 segment as an independent entity unrelated to any other internet segment. There are no connections or logical circuits (virtual or otherwise).

2. PHILOSOPHY

2.1. Related Work

The TCP development cited in the History Section (1.1) is closely related to this work. Other work on the interconnection of networks can be found in the reports of the International Network Working Group (INWG) [11].

2.2. Mechanisms Explained

Addressing

A distinction is made between names, addresses, and routes [12]. 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 an one octet network number, followed by a three octet host number.

Care must be taken in mapping internet addresses to local net addresses; we want to permit one physical host to act as if it were several distinct hosts to the extent of using several distinct internet addresses.

Fragmentation

Fragmentation of an internet segment 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 segment can be marked "don't fragment." Any internet segment so marked is not to be internet fragmented under any circumstances (however, intranet fragmentation may be used, that is a fragmentation and reassembly across a local network which is invisible to the internet protocol module). If such an internet segment can not 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 sender of an internet segment sets the identification field to a value that must be unique for that source-destination pair for the time the segment will be active in the internetwork system. The originator of a complete segment 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 even 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 correct value. 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 correct value. 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 segment, an internet protocol module (for example at a destination host) combines internet packets that all have the same value for the three fields: identification, destination, and source. 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.

2.3. Functional Specification of Interfaces

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.

2.4. Problems Remaining

Major Items

A formal specification system must be selected, and the formal specification created.

The protocol must be verified.

Implementation recommendations must be provided.

Examples and scenarios must be created.

Technical Points

Source Routing

It is thought that in some cases the sender may wish or need to specify the route to be traversed through the internetwork system rather than the address of the destination. Current plans call for an option to be developed to carry such information.

Address Assignment

Care must be taken in mapping internet addresses to local net addresses, we want to permit one physical host to act as if it were several distinct hosts to the extent of using several distinct internet addresses.

Longer Addresses

In some cases, it may be desired to use longer adddresses than are permitted in the regular internet header address fields.

Type of Service

The types of service defined have yet to be proven in use; experimentation is needed. A method for stream setup is not yet defined.

Header Checksum

Experience with the header checksum procedure is needed; it may be that it will be replaced by a stronger checksum procedure.

Options

Additional options are to be defined.

Treatment of Errors

The development of error reporting conventions is needed.

2.5. Lessons Learned

It is still very early in the game to say much about lessons learned, but we will make the following observations:

Addressing:

Addressing is a complicated issue and it is still not clear what the best approach is. One camp argues that "All addressing information should be on the outermost envelope, i.e., in the internet header" -- while another camp stresses the need to minimize header (overhead) bits.

Fragmentation:

Fragmentation must be in the domain of the gateways, yet the gateways must have the least possible knowledge of end-to-end protocols.

Features:

The outermost protocol (i.e. internet protocol) must make no assumptions about the type of service the application desires.

For example, it would have been easy to have the internet checksum cover the whole segment instead of just the header, but it might be desired by some application to have data delivered even if it contains errors.

2.6. Future Directions

One feature currently under discussion is a provision for multplexing several next level protocol packages in one internet segment.

3. SPECIFICATION

3.1. Formalisms Explained

No formal specification technique has been selected as yet.

3.2. Formal Specification

No formal specification is available as yet.

3.3. 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.

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 Relibility.
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-relib 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 segment 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.

Identification: 8 bits

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

Flags: 3 bits

Various Control Flags.

Bit 0: Options Present (OP).
Bit 1: Don't Fragment This Segment (DF).
Bit 2: More Fragments Flag (MF).

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

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, except that unchecksummed option fields are replaced with zeros in the computation.

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

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

Destination Address: 32 bits

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

Source Address: 32 bits

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

Options: variable

The option field is variable in length. The format is an option-type octet, a length octet, and the actual option octets. Although it is not clear that any option can be inserted at a point that could not also recompute the header checksum, the ability to have unchecksummed options is provided.

The high order bit of the option-type octet, if set, indicates that the option should NOT be included in any checksum. The length octet, which follows, includes the option-type octet and the length octet in the octet count of the option length.

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

1 bit   checksum exclusion flag,
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:

CKSUM	CLASS	NUMBER	LENGTH	DESCRIPTION
0	0	0	-	End of Option list. This option occupies only 1 octet; it has no length octet.
0	0	1	-	Padding. This option occupies only 1 octet; it has no length octet.
0	0	2	4	S/P/T. Used to carry Security, Precedence, and user group (TCC) information compatible with AUTODIN II requirements.
0	0	3	var.	Source Routing. Used to route the internet packet based on information supplied by the source.
0	1	1	var.	General Error Report. Used to report errors in internet packet processing.
X	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.
X	2	5	var.	Satellite Timestamp. Used as above for special satellite network testing.

Specific Option Definitions

End of Option List

```
+--------+
!00000000!
+--------+
  Code=0
```

This option code 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.

Padding

```
+--------+
!00000001!
+--------+
  Code=1
```

This option code may be used between options, for example, to align the begining of a subsequent option on a word 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   !
+--------+--------+--------+--------+
  Code=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  !
+--------+--------+--------+-------//--------+
  Code=3
```

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

A source route is composed of a series of 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 exausted, at that point the destination has been reached.

General Error Report

```
+--------+--------+--------+--------+--------//--------+
!00100001! length !err code!   id   !                  !
+--------+--------+--------+--------+--------//--------+
  Code=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.

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

Internet Timestamp

```
+--------+--------+--------+--------+--------//--------+
!x1000100! length !   ?    !   ?    !        ?         !
+--------+--------+--------+--------+--------//--------+
```

No information is available on the specific format of Timestamps.

Satellite Timestamp

```
+--------+--------+--------+--------+--------//--------+
!x1000101! length !   ?    !   ?    !        ?         !
+--------+--------+--------+--------+--------//--------+
```

No information is available on the specific format of Timestamps.

Padding: variable

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

3.4. 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 RFC 739 [13]. In any case, the latest information can be obtained from Jon Postel.

The 24 bit host number, 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 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 the segment total length field. Since a typical internet header is most likely 160 bits long, fragmentation under this scheme has an efficiency of 800/(224+800) = 0.83 for internet packets carried in ARPANET type 3 packets (and 608/(160+192+608) = 0.63 for the data in the first fragment of a TCP segment). Of course, efficiencies higher than this are possible for systems whose minimum packet size is larger than 1008 bits.

When fragmentation occurs, options are generally not copied, but remain with the first fragment. For concreteness, an example of a fragmented packet is illustrated in example 2 below.

The fields which may be affected by fragmentation include:

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

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

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 segments.

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

Stream or Datagram. Indicates if there will be other segments 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 segment.

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 segment.

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=relibility, P=preference, *=requires stream set up; T=type, S=size; R=route, A=ack; T=type, D=delay, H=holding time, R=relibility

Time to Live

The time to live is set by the sender to the maximum time the segment is allowed to be in the internetwork system. If the segment is in the internetwork system longer than the time to live, then the segment should be destroyed. This field should be decreased at each point that the internet header is processed to reflect the time spent processing the segment. 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 Present flag bit (OP) is set if options are present in the internet header.

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 or a change to the address pointer field. This checksum at the internet level will protect the internet header fields from transmission errors.

3.5. Examples & Scenarios

Example 1:

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

```
 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        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                      destination address                      !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                         source address                        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!     data      !
+-+-+-+-+-+-+-+-+
```

Example Internet Packet Header

Figure 3.

Note that each tick mark represents one bit position.

This is a internet segment in version 4 of internet protocol; the internet header consists of five 32 bit words, and the total length of the segment is 28 octets. This packet is a complete segment (not a fragment).

Example 2:

In this example, we show first an moderate size internet segment (7646 data octets), then two internet fragments that might result from the fragmentation of this segment.

```
 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 = 7666       !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!     Identification = 111      !Flg=0!  Fragment Offset = 0    !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!   Time = 123  ! Protocol = 6  !        header checksum        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                     destination address                       !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                       source address                          !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
\                                                               \
\                                                               \
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!             data              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Example Internet Packet Header

Figure 4.

Now the first fragment that results from splitting the segment after 3824 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 = 3844     !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!      Identification = 111     !Flg=1!   Fragment Offset = 0   !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!   Time = 119  ! Protocol = 6  !        Header Checksum        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                     destination address                       !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                       source address                          !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
\                                                               \
\                                                               \
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Example Internet Packet Header

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 = 3842     !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!     Identification = 111      !Flg=0!  Fragment Offsett = 478 !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!   Time = 119  ! Protocol = 4  !        Header Checksum        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                      destination address                      !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                         source address                        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             data                              !
\                                                               \
\                                                               \
!                             data                              !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!            data               !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Example Internet Packet Header

Figure 6.

Example 3:

Here, we show an example of a header 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 = 1232     !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!      Identification = 111     !Flg=4!   Fragment Offset = 0   !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!   Time = 123  !  Protocol = 6 !       Header Checksum         !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                     destination address                       !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                       source 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 Packet Header

Figure 7.

3.6. 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 segment 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 by communication:

SEND (dest, TOS, TTL, BufPTR, len)

where:

dest = destination address
TOS = type of service
TTL = time to live
BufPTR = buffer pointer
len = length of buffer

Response:

OK = sent ok
Error = error in arguments or local network error

RECV (BufPTR)

Reponse:

OK = received ok with the addtional information:
source address and length
Error = error in arguments or local network error

When the user sends a segment, 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 segment is accepted by the local network, the call returns successfully. If either the arguments are bad, or the segment 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 segment 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 segment to the user. In the second case, the user addressed is notified of a pending segment. If the user addressed does not exist, an error segment 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 segment, or the call may be pending until a segment arrives.

4. VERIFICATION

Requires further research.

5. IMPLEMENTATION

5.1. What Not to Leave Out

???

5.2. User Interfaces

???

5.3. Mechanisms

???

5.4. Data Structures

???

5.5. Program Sizes, Performance Data

???

5.6. Test Sequences, Procedures, Exerciser

???

5.7. Parameter Values: Timeouts, Segment sizes, Buffer strategies

???

5.8. Debugging

???

REFERENCES

[1] Vinton G. Cerf and Robert E. Kahn, "A Protocol for Packet Network Intercommunication," IEEE Transactions on Communications, volume COM-22, No. 5, May 1974, p. 637-648. (An early version of this paper appeared as INWG General Note 39, IFIP Working Group 6.1, September 1973).

[2] Vinton G. Cerf, Yogen K. Dalal, Carl Sunshine, "Specification of Internet Transmission Control Program," INWG General Note 72, IFIP Working Group 6.1, RFC 675, NIC 31505, December 1974.

[3] Jonathan B. Postel, Larry L. Garlick, Raphael Rom, "Transmission Control Protocol Specification," Augmentation Research Center, Stanford Research Institute, Menlo Park, CA, 15 July 1976.

[4] Vinton G. Cerf, "Specification of Internet Transmission Control Program - TCP (Version 2)," IEN 5, March 1977.

[5] Vinton G. Cerf and Jonathan B. Postel, "Specification of Internetwork Transmission Control Program - TCP Version 3," Information Sciences Institute, IEN 21, January 1978.

[6] Jonathan B. Postel, "Draft Internetwork Protocol Specification - Version 2," Information Sciences Institute, IEN 28, February 1978.

[7] Vinton G. Cerf, "A Proposed New Internet Header Format," Advanced Research Projects Agency, IEN 26, February 1978.

[8] Vinton G. Cerf, "A Proposal for TCP Version 3.1 Header Format," Advanced Research Projects Agency, IEN 27, February 1978.

[9] Jonathan B. Postel, "Draft Internetwork Protocol Specification - Version 4," Information Sciences Institute, IEN 41, June 1978.

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

[11] INWG, the International Network Working Group, Chairman: Mr. Derek L. A. Barber, Project EIN, National Physical Laboratory, Teddington, Middlesex, England.

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

[13] J. Postel, "Assigned Numbers," RFC 739, NIC 42341, 11 November 1977.

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.

Address

An address is a fixed length quantity of four octets (32 bits).

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).

Destination

The destination address, an internet header field.

DF

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

DGP

DataGram Protocol: A host-to-host protocol for communication of raw data.

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 segment.

Fragment Offset

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

header

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

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

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 fragment
A portion of the data of an internet segment with an internet header.

internet packet
Either an internet segment or an internet fragment.

internet segment
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.

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 segment, 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.

RTP

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

segment

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

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.