INTERNET-DRAFT
February 21, 1993
Obsoletes: draft-deering-sip-00.txt

S. Deering
Xerox PARC

Simple Internet Protocol Plus (SIPP) Specification

draft-ietf-sipp-spec-00.txt

Abstract

This document specifies a proposed new version of the Internet Protocol. Changes from the previous specification (SIP) are listed in Appendix A.

Status of this Memo

Internet Drafts are working documents of the Internet Engineering Task Force (IETF), its Areas, and its Working Groups. Note that other groups may also distribute working documents as Internet Drafts.

Internet Drafts are draft documents valid for a maximum of six months. This Internet Draft expires on September 21, 1994. Internet Drafts may be updated, replaced, or obsoleted by other documents at any time. It is not appropriate to use Internet Drafts as reference material or to cite them other than as a "working draft" or "work in progress."

Please check the I-D abstract listing contained in each Internet Draft directory to learn the current status of this or any other Internet Draft.

Distribution of this memo is unlimited.

Contents

1. Introduction

The Simple Internet Protocol Plus (SIPP) is a new version of the Internet Protocol, designed as a successor to IP version 4 (IPv4) [RFC-791]. The changes from IPv4 to SIPP fall primarily into the following categories:

- Expanded Addressing Capabilities

  SIPP increases the IP address size from 32 bits to 64 bits, to support more levels of addressing hierarchy and a much greater number of addressable nodes. SIPP addressing can be further extended, in units of 64 bits, by a facility equivalent to IPv4's Loose Source and Record Route option, in combination with a new address type called "cluster addresses" which identify topological regions rather than individual nodes. The scalability of multicast routing is improved by adding a "scope" field to multicast addresses.

- Header Format Simplification

  Some IPv4 header fields have been dropped or made optional, to reduce the common-case processing cost of packet handling and to keep the bandwidth cost of the SIPP header almost as low as that of IPv4, despite the increased size of the addresses.

- Improved Support for Options

  Changes in the way IP header options are encoded allows for more efficient forwarding, less stringent limits on the length of options, and greater flexibility for introducing new options in the future.

- Flow Labeling Capability

  A new capability is added to enable the labeling of packets belonging to particular traffic "flows" for which the sender requests special handling, such as non-default quality of service or "real-time" service.

This document specifies the basic SIPP header and the initially-defined SIPP options. It also discusses packet size issues, the semantics of Flow Labels, and the effects of SIPP on transport-layer protocols. Other aspects of SIPP are specified in separate documents, including the following:

- Simple Internet Protocol Plus (SIPP): Routing and Addressing [SIPP-ROAD]

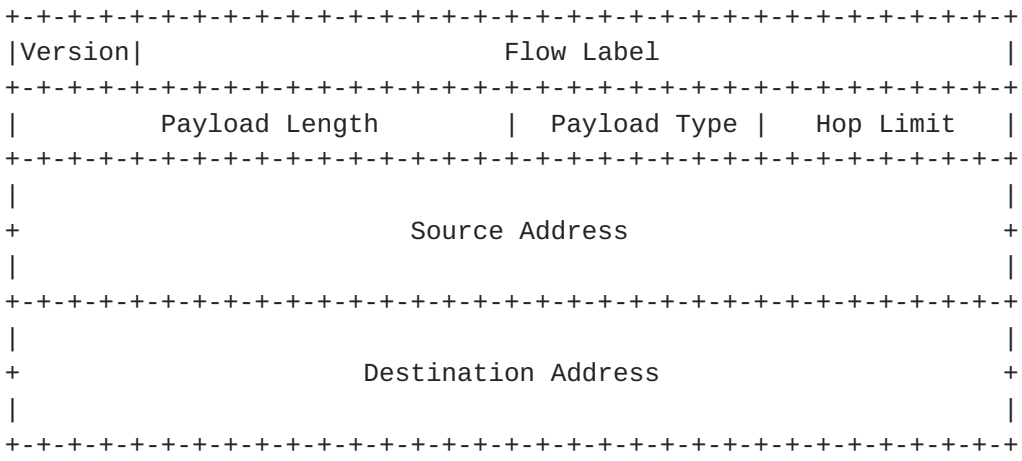
- ICMP and IGMP for SIPP Specification [SIPP-ICMP]

- IPAE: The SIPP Interoperability and Transition Mechanism [IPAE]

2. Terminology

node	- a protocol module that implements SIPP.
router	- a node that forwards SIPP packets not explicitly addressed to itself.
host	- any node that is not a router.
address	- a SIPP-layer identifier for a node or set of nodes.
subnet	- in the SIPP unicast addressing hierarchy, a lowest-level (finest-grain) aggregation of addresses sharing a common prefix, i.e., high-order address bits.
link	- a communication facility or medium over which nodes can communicate at the link layer, i.e., the layer immediately below SIPP.
interface	- a node's attachment to a link.
neighbors	- nodes attached to the same link.
link MTU	- the maximum transmission unit, i.e., maximum packet size in octets, that can be conveyed in one piece over a link (where a packet is a SIPP header plus payload).
path MTU	- the minimum link MTU of all the links in a path between a source node and a destination node.
packetization layer	- any protocol layer above SIPP that is responsible for packetizing data to fit within outgoing SIPP packets. Typically, transport-layer protocols, such as TCP, are packetization protocols, but there may also be higher- layer packetization protocols, such as protocols implemented on top of UDP.

3. SIPP Header Format



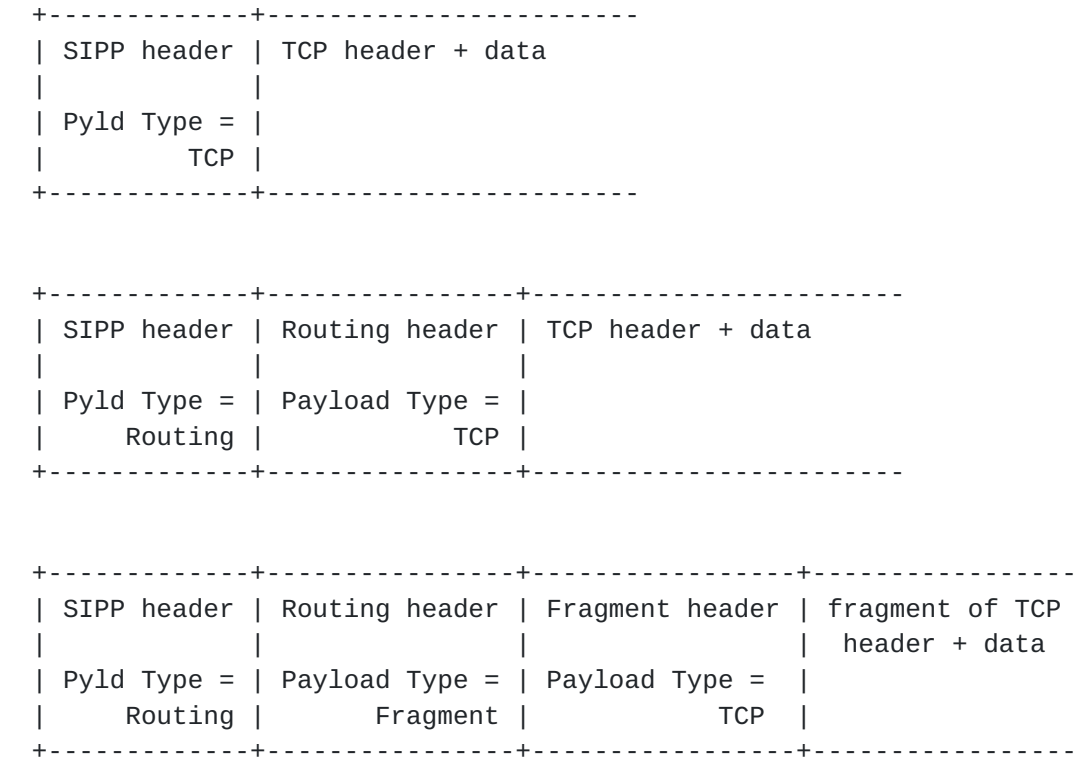
Version	4-bit Internet Protocol version number = 6.
Flow Label	28-bit field. See "Flow Labels" section, below.
Payload Length	16-bit unsigned integer. Length of payload, i.e., the rest of the packet following the SIPP header, in octets.
Payload Type	8-bit selector. Identifies the type of header immediately following the SIPP header. Uses the same values as the IPv4 Protocol field [RFC-1340].
Hop Limit	8-bit unsigned integer. Decremented by 1 by each node that forwards the packet. The packet is discarded if Hop Limit is decremented to zero.
Source Address	64 bits. An address of the initial sender of the packet. See [SIPP-ROAD] for details.
Destination Address	64 bits. An address of the intended recipient of the packet (possibly not the ultimate recipient, if an optional Routing Header is present). See section 4.2 and [SIPP-ROAD] for details.

4. SIPP Options

In SIPP, optional internet-layer information is encoded in separate headers that may be placed between the SIPP header and the transport- layer header

in a packet. There are a small number of such optional headers, each identified by a distinct Payload Type. As illustrated in these examples, a

SIPP packet may carry zero, one, or more optional headers, each identified by the Payload Type field of the preceding header:



With one exception, optional headers are not examined or processed by any node along a packet's delivery path, until the packet reaches the node (or set of nodes, in the case of multicast) identified in the Destination Address field of the SIPP header. There, normal demultiplexing on the Payload Type field of the SIPP header invokes the module to process the first optional header, or the transport header if no optional header is present. The contents and semantics of each header determine whether or not to proceed to the next header.

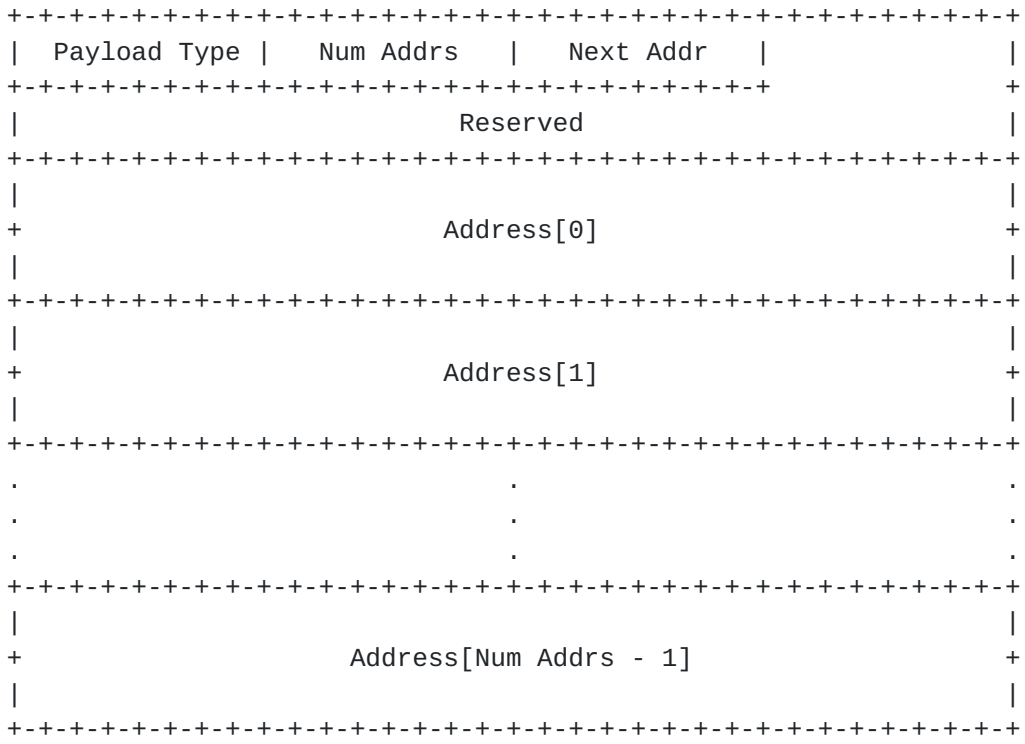
The one exception is the Hop-by-Hop Options header, which carries options that must be examined by every node along a packet's delivery path. The Hop-by-Hop options header, when present, must immediately follow the SIPP header. Its presence is indicated by the special value zero in the Payload Type field of the SIPP header.

Each optional header is an integer multiple of 8 octets long, in order to retain 8-octet alignment for subsequent headers. Multi-octet fields within each optional header are aligned on their natural boundaries, i.e., fields of width n octets are placed at an integer multiple of n octets from the start of the header.

4.1 Routing Header

The Routing header is used by a SIPP source to list one or more

intermediate nodes (or topological clusters) to be "visited" on the way to a packet's destination. This function is very similar to IPv4's Loose Source and Record Route option. The Routing header is identified by a Payload Type of 43 in the immediately preceding header, and has the following format:



Payload Type	8-bit selector. Identifies the type of header immediately following the Routing header. Uses the same values as the IPv4 Protocol field [RFC-1340].
Num Addrs	8-bit unsigned integer. Number of addresses in the Routing header.
Next Addr	8-bit unsigned integer. Index of next address to be processed; initialized to 0 by the originating node.
Reserved	40-bit field. Initialized to zero for transmission; ignored on reception.

A Routing header is not examined or processed until it reaches the node identified in the Destination Address field of the SIPP header. In that node, dispatching on the Payload Type of the immediately preceding header causes the Routing module to be invoked, which performs the following algorithm:

- If Next Addr < Num Addrs, swap the SIPP Destination Address and Address[Next Addr], increment Next Addr by one, and re-submit the

packet to the SIPP module for forwarding to the next destination.

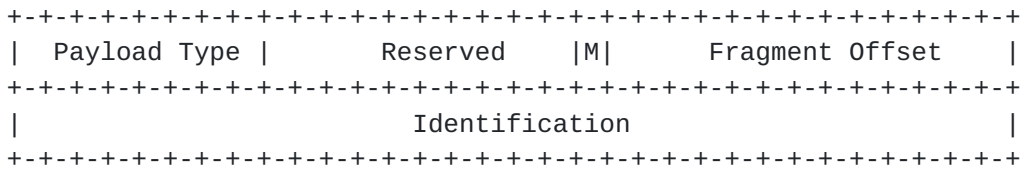
- If Next Addr = Num Addrs, dispatch to the local protocol module identified by the Payload Type field in the Routing header.

- If Next Addr > Num Addrs, send an ICMP Parameter Problem message to the Source Address, pointing to the Num Addrs field.

Multicast addresses may not appear in a Routing header.

4.2 Fragment Header

The Fragment header is used by a SIPP source to send payloads larger than would fit in the path MTU to their destinations. (Note: unlike IPv4, fragmentation in SIPP is performed only by source nodes, not by routers along a packet's delivery path -- see section 5.) The Fragment header is identified by a Payload Type of 44 in the immediately preceding header, and has the following format:



Payload Type	8-bit selector. Identifies the type of header immediately following the Fragment header. Uses the same values as the IPv4 Protocol field [RFC-1340].
Reserved	10-bit field. Initialized to zero for transmission; ignored on reception.
M flag	1 = more fragments; 0 = last fragment.
Fragment Offset	13-bit unsigned integer. The offset, in 8-octet units, of the following payload, relative to the start of the original, unfragmented payload.
Identification	32 bits. See description below.

The fragmentation algorithm is as follows: The payload (including any optional headers that need be processed only by the destination node(s)) is divided into fragments, each, except possibly the last, being an integer multiple of 8 octets long. Each fragment is prepended with a Fragment header and sent in a separate SIPP packet. The M ("more") flag is set to 1 on all fragments of the same payload except the last. The original payload is assigned an Identification value that is different than that of any

other fragmented payload sent recently* with the same SIPP Source Address, SIPP Destination Address, and Fragment Payload Type. (If an optional

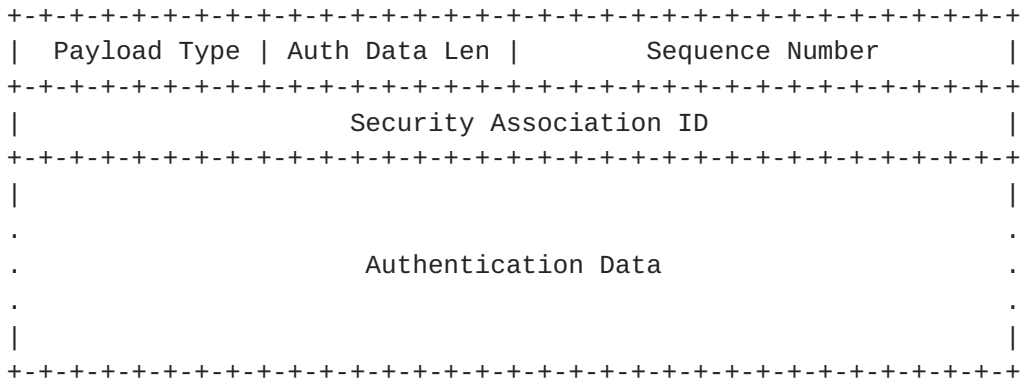
Routing header is present, the SIPP Destination Address is that of the final destination.) The Identification value is carried in the Fragment header of all of the original payload's fragments, and is used by the destination to identify all fragments belonging to the same original payload.

* "recently" means within the maximum likely lifetime of a packet, including transit time from source to destination and time spent awaiting reassembly with other fragments of the same payload. However, it is not required that a source node know the maximum packet lifetime. Rather, it is assumed that the requirement can be met by maintaining the Identification value as a simple, 32-bit, "wrap-around" counter, incremented each time a payload must be fragmented. It is an implementation choice whether to maintain a single counter for the node or multiple counters, e.g., one for each of the node's possible source addresses, or one for each active (source address, destination address, payload type) combination.

In a packet with a Fragment header, the Payload Length field of the SIPP header contains the length of that packet only (excluding the SIPP header itself), not the length of the original, unfragmented payload.

4.3 Authentication Header

The Authentication header is used to provide authentication and integrity assurance for SIPP packets. Non-repudiation may be provided by an authentication algorithm used with the Authentication header, but it is not provided with all authentication algorithms that might be used with this header. Privacy of the payload following the Authentication header may optionally be provided by encryption, using an algorithm and keying information that has been pre-established as part of a security association. The Authentication header is identified by a Payload Type of TBD in the immediately preceding header, and has the following format:



Payload Type 8-bit selector. Identifies the type of header

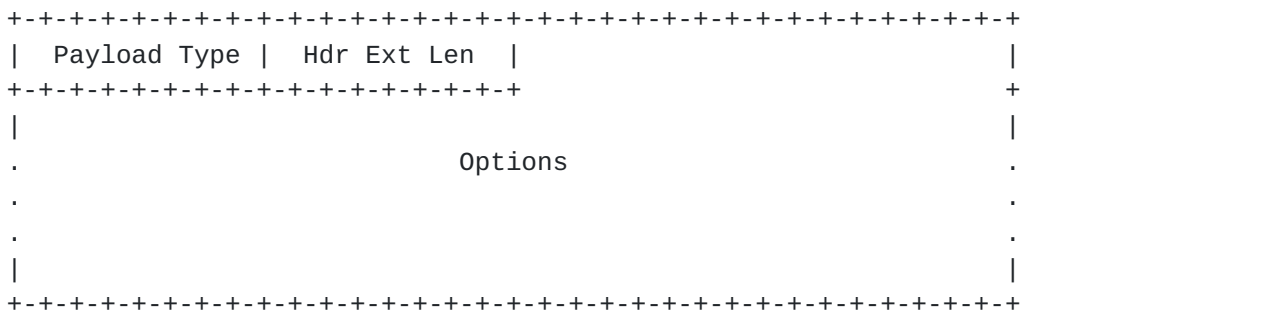
immediately following the Authentication header. Uses the same values as the IPv4 Protocol field [RFC-1340].

Auth Data Len	8-bit unsigned integer. Length of the Authentication Data field in 8-octet units.
Sequence Number	16-bit wrap-around counter. The sequence number of this packet, relative to other packets sent in the same security association.
Security Assoc. ID	32 bits. When combined with the SIPP Source Address, identifies to the receiver(s) the pre-established security association to which this packet belongs.
Authentication Data	Variable-length field, an integer multiple of 8 octets long. Algorithm-specific information required authenticate the source of the packet and assure its integrity, as specified for the pre-established security association.

The usage of the Authentication header is specified in [SIPP_AUTH]. All SIPP nodes are required to support the keyed MD5 algorithm used used with the Authentication header as described in that document.

4.4 Hop-by-Hop Options Header

The Hop-by-Hop (HBH) Options header is used to carry optional information that must be examined by every node along a packet's delivery path. The HBH Options header is identified by a Payload Type of 0 in the SIPP header, and has the following format:

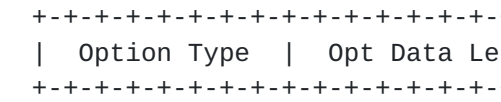


Payload Type	8-bit selector. Identifies the type of header immediately following the HBH Options header. Uses the same values as the IPv4 Protocol field [RFC-1340].
Hdr Ext Len	8-bit unsigned integer. Length of the HBH Options header in 8-octet units, not including the first 8 octets.

Options	Variable-length field, an integer multiple of 8 octets long. One or more individual TLV-

encoded options, as described below.

The individual options within the Options area are each encoded in a Type-Length-Value (TLV) format, as follows:



Option Type — 8-bit identifier of the type of option.

Opt Data Len — 8-bit unsigned integer. Length of the Option Data field of this option, in octets.

Option Data — Variable-length field. Option-Type-specific data.

The Option Type identifiers are internally encoded such that their highest-order two bits specify the action that must be taken if the processing SIPP node does not recognize the Option Type:

00 - skip over this option and continue processing the HBH Options header.

01 - discard the packet.

10 - discard the packet and send an ICMP Parameter Problem message to the packet's Source Address, pointing to the unrecognized Option Type.

11 - discard the packet and, only if the packet's Destination Address is not a multicast address, send an ICMP Parameter Problem message to the packet's Source Address, pointing to the unrecognized Option Type.

The third-highest-order bit of the Option Type specifies whether or not the Option Data of this option shall be included in the integrity assurance computation performed when an Authentication header is present. Option data that changes en route must be excluded from that computation.

0 - include in integrity computation

1 - exclude from integrity computation

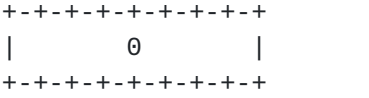
Individual options may have alignment requirements, to ensure that multi-octet values within Option Data fields fall on natural boundaries. The alignment requirement of an option is specified using the notation xn+y, meaning the Option Type must appear at an integer multiple of x octets from the start of the HBH Options header, plus y octets. For example:

2n means any 2-octet offset from the start of the header.

8n+2 means any 8-octet offset from the start of the header, plus 2 octets.

The only hop-by-hop options initially defined are padding options, which are used when necessary to align subsequent options and to pad out the Options area of the header to a multiple of 8 octets in length. These padding options must be recognized by all SIPP implementations:

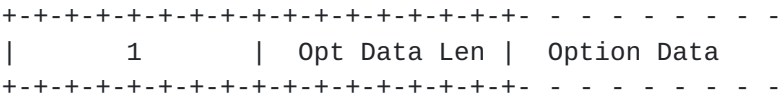
Pad1 option (alignment requirement: none)



NOTE! the format of the Pad1 option is a special case -- it does not have length and value fields.

The Pad1 option is used to insert one octet of padding into the Options area of an HBH Options header. If more than one octet of padding is required, the PadN option, described next, should be used, rather than multiple Pad1 options.

PadN option (alignment requirement: none)



The PadN option is used to insert two or more octets of padding into the Options area of an HBH Options header. For N octets of padding, the Opt Data Len field contains the value N-2, and the Option Data consists of N-2 zero-valued octets.

4.5 Option Header Order

When more than one optional header is used in the same packet, they should appear in the following order:

SIPP header
Hop-by-Hop Options header
Routing header
Fragment header
Authentication header

If and when other optional headers are defined, their ordering constraints relative to the above listed headers must be specified.

SIPP nodes are not required to verify that the order of headers in received

packets satifies the above order; sending packets with optional headers in some other order may or may not achieve useful effects.

5. Packet Size Issues

SIPP requires that every link in the internet have an MTU of 576 octets or greater. On any link that cannot convey a 576-octet packet in one piece, link-specific fragmentation and reassembly must be provided at a layer below SIPP.

> (Note: this minimum link MTU is NOT the same as the one in IPv4. In IPv4, the minimum link MTU is 68 octets [RFC-791, page 25]; 576 octets is the minimum reassembly buffer size required in an IPv4 node, which has nothing to do with link MTUs.)

From each link to which a node is directly attached, the node must be able to accept packets as large as that link's MTU. Links that have a configurable MTU, such as PPP links [RFC-1548], should be configured with an MTU of 600 octets or greater.

SIPP nodes are expected to implement Path MTU Discovery [RFC-1191], in order to discover and take advantage of paths with MTU greater than 576 octets. However, a minimal SIPP implementation (e.g., in a boot ROM) may simply restrict itself to sending packets no larger than 576 octets, and omit implementation of Path MTU Discovery.

In order to send a packet larger than a path's MTU, a node may use the optional SIPP Fragment Header to fragment the packet at the source and have it reassembled at the destination(s). However, the use of such fragmentation is discouraged in any application that is able to adapt its packets to fit the measured path MTU (i.e., down to 576 octets). A node must not send a packet larger than the path MTU (i.e., fragments that reassemble to a size larger than the path MTU) unless it has explicit knowledge that the destination(s) can reassemble a packet of that size.

In response to a SIPP packet that is sent to an IPv4 destination (i.e., a packet that undergoes translation from SIPP to IPv4), the originating SIPP node may receive an ICMP Packet Too Big message reporting a Next-Hop MTU less than 576. In that case, the SIPP node is not required to reduce the size of subsequent packets to less than 576, but must include a Fragment Header in those packets so that the SIPP-to-IPv4 translating router can obtain a suitable Identification value to use in resulting IPv4 fragments. Note that this means the payload may have to be reduced to 544 octets (576 minus 24 for the SIPP Header and 8 for the Fragment Header), and smaller still if additional optional SIPP headers are used.

> Note: Since SIPP allows a subnet to span more than one link, Path MTU Discovery must be performed even between nodes on the same subnet.

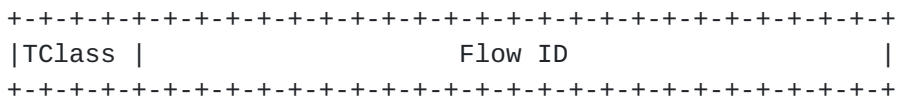
Note: Unlike IPv4, it is unnecessary in SIPP to set a "Don't Fragment"

flag in the packet header in order to perform Path MTU Discovery; that is an implicit attribute of every SIPP packet. Also, those parts of the RFC-1191 procedures that involve use of a table of MTU "plateaus" do not apply to SIPP, because the SIPP version of the "Datagram Too Big" message always identifies the exact MTU to be used.

6. Flow Label

The Flow Label field in the SIPP header may be used by a source to label those packets for which it requests special handling by the SIPP routers, such as non-default quality of service or "real-time" service. This aspect of SIPP is, at the time of writing, still experimental and subject to change as the requirements for flow support in the Internet become clearer. Hosts or routers that do not support the functions of the Flow Label field are required to set the field to zero when originating a packet, and to ignore the field when receiving a packet.

The Flow Label is a 28-bit field, internally structured into two subfields as follows:



TClass 4-bit traffic class, described below.

Flow ID 24-bit flow identifier, described below.

A flow is a sequence of packets sent from a particular source to a particular (unicast or multicast) destination for which the source desires special handling by the intervening routers. The nature of that special handling might be conveyed to the routers by a control protocol, such as a resource reservation protocol, or by information within the flow's packets themselves, e.g., in a hop-by-hop option. The details of such control protocols or options are beyond the scope of this document.

There may be multiple active flows from a source to a destination, as well as traffic that is not associated with any flow. A flow is identified by the combination of a Source Address and a non-zero Flow ID. Packets that do not belong to a flow carry a Flow ID of zero.

A Flow ID is assigned to a flow by the flow's source node. New Flow IDs must be chosen (pseudo-)randomly and uniformly from the range 1 to FFFFFF hex. The purpose of the random allocation is to make any set of bits within the Flow ID suitable for use as a hash key by the routers, for looking up the special-handling state associated with the flow. A Flow ID must not be re-used by a source for a new flow while any state associated with the previous usage still exists in any router. The lifetime of flow

state in routers depends on the method by which that state is created, and is beyond the scope of this document.

All packets sent with the same Source Address and same non-zero Flow ID must also carry the same Destination Address, same Hop-by-Hop Options header contents (if present), and same Routing header contents (if present). The routers or destinations are permitted, but not required, to verify that this condition is satisfied. If a violation is detected, it should be reported to the source by an ICMP Parameter Problem message, pointing to any one of the invalid fields.

The TClass subfield provides a means separate from the Flow ID for a source to identify the desired delivery priority of its packets, relative to other packets from the same source. The TClass values are divided into two ranges: values 0 through 7 are used to label flow-controlled packets, e.g., packets that belong to a TCP connection, and values 8 through 15 are used to label non-flow-controlled packets, e.g., "real-time" packets being sent without any flow-control feedback from the receivers.

For flow-controlled traffic, the following TClass values are recommended for particular application categories:

0 - uncharacterized traffic
1 - "filler" traffic (e.g., netnews)
2 - unattended data transfer (e.g., email)
3 -
4 - attended bulk transfer (e.g., FTP, NFS)
5 -
6 - interactive traffic (e.g., telnet, X)
7 - internet control traffic (e.g., routing protocols, SNMP)

For non-flow-controlled traffic, the lowest TClass value (8) should be used for those packets that the sender is most willing to have discarded under conditions of congestion (e.g., high-fidelity video traffic), and the highest value (15) should be used for those packets that the sender is least willing to have discarded (e.g., low-fidelity audio traffic). There is no relative ordering implied between the flow-controlled classes and the non-flow-controlled classes.

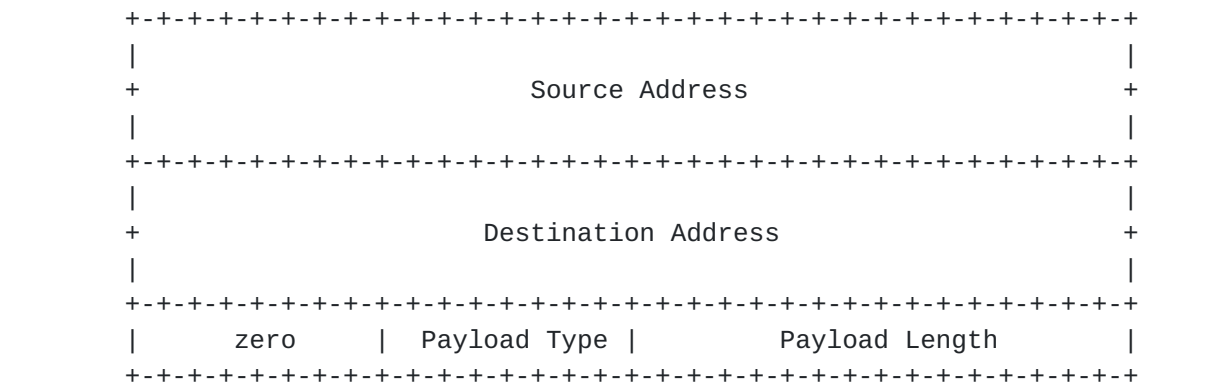
For packets bearing non-zero Flow IDs, the method by which the flow requirements are conveyed (e.g., a control protocol or a hop-by-hop option) may include the ability to redefine the semantics of the TClass subfield, for example, to define it as a priority relative to packets belonging to the same flow or set of related flows.

7. Transport-Layer Protocol Issues

7.1 Transport-Layer Checksums

When TCP, UDP, ICMP, or IGMP (subsequently referred to as "transport" protocols) is used between two or more SIPP systems, the transport checksum

computation includes a "pseudo-header" of the following form:



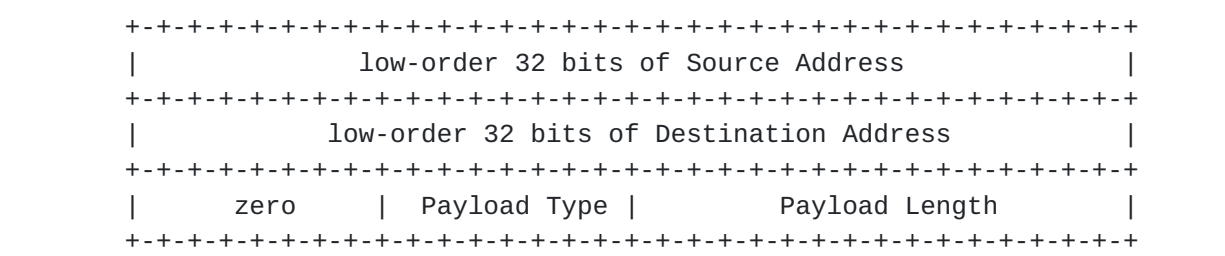
- If the packet contains a Routing header, the Destination Address used in the pseudo-header is that of the final destination. At the originating system, that address will be in the last element of the Routing header; at the recipient(s), that address will be in the Destination Address field of the SIP header.

- The Payload Type used in the pseudo-header is that of the transport protocol (6 for TCP, 17 for UDP, 1 for ICMP, or 2 for IGMP). It will differ from the Payload Type in the SIPP header if there are additional headers between the SIPP header and the transport header.

- The Payload Length used in the pseudo-header is the length of the transport packet, incluing the transport header. It will be less than the Payload Length in the SIPP header if there are additional headers between the SIP header and the transport header.

- The UDP checksum is not optional when UDP is used between SIPP systems. At a recipient, a UDP checksum value of zero is considered valid only if the checksum computation across the the pseudo-header, UDP header, and UDP data yields a result of either zero or hex FFFF (which are equal values in ones-complement arithmetic).

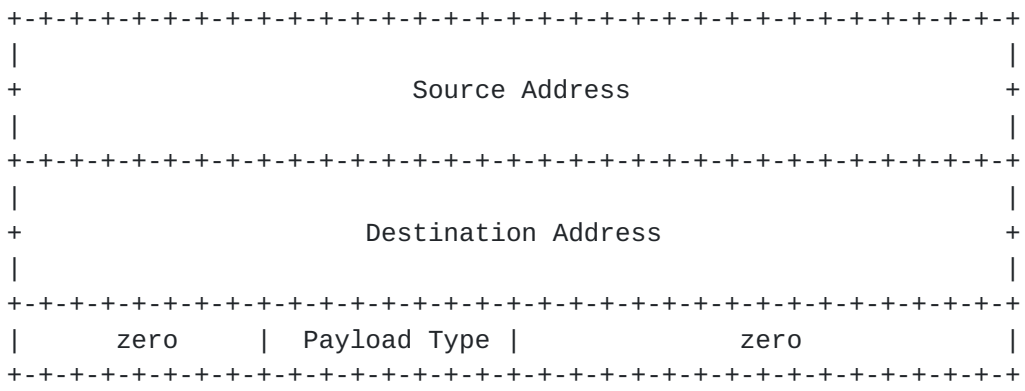
For TCP or UDP, if the remote address (i.e., the Source Address in an incoming packet, or the Destination Address in an outgoing packet) is that of an IPv4 system, i.e., has a C-bit value of 1, then the following pseudo-header is used, instead of the one shown above:



When the remote address is that of an IPv4 system, a UDP Checksum field of zero in an incoming packet indicates the absence of a UDP checksum. Such

packets shall be accepted as is. However, a SIPP system must always compute a checksum for outgoing UDP outgoing packets, and it must be non-zero if the remote address is that of an IPv4 system (i.e., if the result of the checksum computation is zero, the value hex FFFF must be sent in the UDP Checksum field).

For ICMP or IGMP, if the remote address is that of an IPv4 system, then the following pseudo-header is used, instead of the one shown above:



It is different from the TCP and UDP case because IPv4 systems do not include a pseudo-header in the ICMP or IGMP checksums, and therefore those checksums have to be adjusted when translating between SIPP and IPv4. The Payload Length is omitted because, otherwise, translating gateways would not be able to adjust the checksum properly when translating between a SIPP fragment and an IPv4 fragment, since the Payload Length would not be known to the translating gateway. Note that this makes ICMP and IGMP packets between SIPP and IPv4 systems vulnerable to undetected padding or truncation, if the SIPP Payload Length field is corrupted en route. Truncation would be undetected only if the missing original octets were all zero; padding would be undetected only if the added octets were all zero.

7.2 Maximum Packet Lifetime

SIPP does not enforce maximum packet lifetime. Any transport protocol that relies on IPv4 to limit packet lifetime must take this change into account, for example, by providing its own mechanisms for detecting and discarding obsolete packets.

Appendix A. Changes from the SIP Draft of November 10, 1992

- Changed name from "SIP" to "SIPP" as part of merger with Pip.
- Revised Introduction section.
- Changed the term "system" to "node", in the terminology section.
- Changed Reserved field in SIP header to Flow Label.
- In the section on packet size issues, added reference to possible use of the Fragment Header, described handling of path MTUs less than 576 (due to SIPP-to-IPv4 translation), and specified that PMTU Discovery must be performed even between nodes on the same subnet.
- Added discussion of general structure of option headers and the ordering constraints on option headers.
- Changed "Source Routing Header" to "Routing Header", and changed the field layout.
- Changed "Fragmentation Header" to "Fragment Header", changed the field layout, and changed the text describing the option, including dropping the requirement that all nodes be able to reassemble packets up to 1024 octets in length.
- Added the Authentication header.
- Added the Hop-by-Hop Options header.
- Added section on Flow Labels.
- Addresses section moved to a separate SIPP Routing and Addressing Specification, in which the following changes were made:
    - Addresses now identify nodes, not interfaces.
    - Added some introductory text to the Addressing section, discussing the binding of addresses to interfaces, and the independence of subnets from physical links.
    - Changed text representation of SIPP addresses.
    - Eliminated discussion of address masks; changed to refer to prefix lengths instead.
    - Added definition of local-use addresses.
    - Changed the term "anyone address" to "cluster address", and

changed definition to include only boundary routers of a cluster, not internal nodes.

  - Added specification of "C-bit = 1" unicast addresses.

  - Changed the names of some of the multicast scope levels from geographic terms (metro, country) to administrative terms (organization, community).

- Subsections on ICMP and IGMP changes moved to a separate document.

- Deleted subsection on Changes to Link-Layer Protocols, and revised subsection on Changes to Transport-Layer Protocols (renamed Transport-Layer Protocol Issues).

- Deleted appendix on SIP Design Rationale.

- Deleted appendix on Future Directions.

Important differences from the SIP paper that appeared in the May 1993 issue of IEEE Network Magazine:

- First 32 values of Flow Label field no longer correspond to IPv4 TOS values.

- Change in the way Hop-by-Hop Options are encoded.

Security Considerations

<to be done>

Acknowledgments

The author gratefully acknowledges the many helpful suggestions of the members of the SIPP working group (formerly the SIP, IPAE, and Pip working groups), the End-to-End Protocols research group, and the Internet Community At Large.

Author's Address

Steve Deering
Xerox Palo Alto Research Center
3333 **Coyote Hill Road**
Palo Alto, CA 94304
phone: (415) 812-4839
email: deering@parc.xerox.com

References

[RFC-791] J. Postel. Internet Protocol. RFC-791, September, 1981.

[RFC-1191] J. Mogul and S. Deering. Path MTU Discovery. RFC-1191, November 1990.

[RFC-1340] J. Reynolds and J. Postel. Assigned Numbers. RFC-1340, July 1992.

[RFC-1548] W. Simpson. The Point-to-Point Protocol (PPP). RFC-1548, December, 1993.

[SIPP-AUTH] R. Atkinson. SIPP Authentication Header. Internet Draft, December 1993.

[SIPP-ICMP] R. Govindan and S. Deering. ICMP and IGMP for SIPP Specification. Internet Draft, February 1994.

[SIPP-ROAD] S. Deering, P. Francis, and R. Govindan. Simple Internet Protocol Plus (SIPP): Routing and Addressing. Internet Draft, January 1994.

[IPAE] R. Gilligan, E. Nordmark, and B. Hinden. IPAE: The SIPP Interoperability and Transition Mechanism. Internet Draft, October 1993.