## IP Payload Compression Protocol (IPComp)
<draft-ietf-ippcp-protocol-05.txt>

Status of this Memo

Abstract

   This document describes a protocol intended to provide lossless
   compression for Internet Protocol datagrams in an Internet
   environment.

1. Introduction

   IP payload compression is a protocol to reduce the size of IP
   datagrams.  This protocol will increase the overall communication
   performance between a pair of communicating hosts/gateways ("nodes")
   by compressing the datagrams, provided the nodes have sufficient
   computation power, through either CPU capacity or a compression
   coprocessor, and the communication is over slow or congested links.

   IP payload compression is especially useful when encryption is
   applied to IP datagrams.  Encrypting the IP datagram causes the data
   to be random in nature, rendering compression at lower protocol
   layers (e.g., PPP Compression Control Protocol [RFC-1962])
   ineffective.  If both compression and encryption are required,

compression MUST be applied before encryption.

This document defines the IP payload compression protocol (IPComp),
the IPComp packet structure, the IPComp Association (IPCA), and
several methods to negotiate the IPCA.

Shacham, Monsour, Pereira, Thomas                              [Page 1]

Other documents shall specify how a specific compression algorithm
can be used with the IP payload compression protocol.  Such
algorithms are beyond the scope of this document.

1.1. Specification of Requirements

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL
NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL"
in this document are to be interpreted as described in RFC 2119
[RFC-2119].

2. Compression Process

The compression processing of IP datagrams has two phases:
compressing of outbound IP datagrams ("compression") and
decompressing of inbound datagrams ("decompression").  The
compression processing MUST be lossless, ensuring that the IP
datagram, after being compressed and decompressed, is identical to
the original IP datagram.

Each IP datagram is compressed and decompressed by itself without any
relation to other datagrams ("stateless compression"), as IP
datagrams may arrive out of order or not arrive at all.  Each
compressed IP datagram encapsulates a single IP payload.

Processing of inbound IP datagrams MUST support both compressed and
non-compressed IP datagrams, in order to meet the non-expansion
policy requirements, as defined in section 2.2.

The compression of outbound IP datagrams MUST be done before any IP
security processing, such as encryption and authentication, and
before any fragmentation of the IP datagram.  In addition, in IP
version 6 [RFC-1883], the compression of outbound IP datagrams MUST
be done before the addition of either a Hop-by-Hop Options header or
a Routing Header, since both carry information that must be examined
and processed by possibly every node along a packet's delivery path,
and therefore MUST be sent in the original form.

Similarly, the decompression of inbound IP datagrams MUST be done

after the reassembly of the IP datagrams, and after the completion
of all IP security processing, such as authentication and
decryption.

## 2.1. Compressed Payload

The compression is applied to a single array of octets, which are
contiguous in the IP datagram.  This array of octets always ends at
the last octet of the IP packet payload.  Note: a contiguous array of
octets in the IP datagram may be not contiguous in physical memory.

In IP version 4 [RFC-0791], the compression is applied to the upper
layer protocol (ULP) payload of the IP datagram.  No portion of the
IP header or the IP header options is compressed.

In the IPv6 context, IPComp is viewed as an end-to-end payload, and
MUST not apply to hop-by-hop, routing, and fragmentation extension
headers.  The compression is applied starting at the first IP Header
Option field that does not carry information that must be examined
and processed by nodes along a packet's delivery path, if such IP
Header Option field exists, and continues to the ULP payload of the
IP datagram.

The size of a compressed payload, generated by the compression
algorithm, MUST be in whole octet units.

As defined in section 3, an IPComp header is inserted immediately
preceding the compressed payload.  The original IP header is modified
to indicate the usage of the IPComp protocol and the reduced size of
the IP datagram.  The original content of the Next Header (IPv6) or
protocol (IPv4) field is stored in the IPComp header.

The decompression is applied to a single contiguous array of octets
in the IP datagram.  The start of the array of octets immediately
follows the IPComp header and ends at the last octet of the IP
payload.  If the decompression process is successfully completed, the
IP header is modified to indicate the size of the decompressed IP
datagram, and the original next header as stored in the IPComp
header.  The IPComp header is removed from the IP datagram and the
decompressed payload immediately follows the IP header.

## 2.2. Non-Expansion Policy

If the total size of a compressed ULP payload and the IPComp header,

as defined in [section 3](), is not smaller than the size of the original
ULP payload, the IP datagram MUST be sent in the original
non-compressed form.  To clarify:  If an IP datagram is sent
non-compressed, no IPComp header is added to the datagram.  This
policy ensures saving the decompression processing cycles and
avoiding incurring IP datagram fragmentation when the expanded
datagram is larger than MTU.

Small IP datagrams are likely to expand as a result of compression.
Therefore, a numeric threshold should be applied before compression,
where IP datagrams of size smaller than the threshold are sent in
the original form without attempting compression.  The numeric
threshold is implementation dependent.

An IP datagram with payload that has been previously compressed
tends not to compress any further.  The previously compressed payload
may be the result of external processes, such as compression applied
by an upper layer in the communication stack, or by an off-line
compression utility.  An adaptive algorithm should be implemented to
avoid the performance hit.  For example, if the compression of i
consecutive IP datagrams of an IPCA fails, the next k IP datagrams
are sent without attempting compression.  If the next j datagrams
are also failing to compress, the next k+n datagrams are sent without
attempting compression.  Once a datagram is compressed successfully,
the normal process of IPComp restarts.  Such an adaptive algorithm,

including all the related thresholds, is implementation dependent.

During the processing of the payload, the compression algorithm
MAY periodically apply a test to determine the compressibility of
the processed data, similar to the requirements of [[V42BIS]().  The
nature of the test is algorithm dependent.  Once the compression
algorithm detects that the data is non-compressible, the algorithm
SHOULD stop processing the data, and the payload is sent in the
original non-compressed form.

[3](). Compressed IP Datagram Header Structure

A compressed IP datagram is encapsulated by modifying the IP header
and inserting an IPComp header immediately preceding the compressed
payload.  This section defines the IP header modifications both in
IPv4 and IPv6, and the structure of the IPComp header.

[3.1](). IPv4 Header Modifications

The following IPv4 header fields are set before transmitting the compressed IP datagram:

   Total Length

      The length of the entire encapsulated IP datagram, including
      the IP header, the IPComp header and the compressed payload.

   Protocol

      The Protocol field is set to 108, IPComp Datagram, [RFC-1700].

   Header Checksum

      The Internet Header checksum [RFC-0791] of the IP header.

All other IPv4 header fields are kept unchanged, including any header
options.

## 3.2. IPv6 Header Modifications

The following IPv6 header fields are set before transmitting the
compressed IP datagram:

   Payload Length

      The length of the compressed IP payload.

   Next Header

      The Next Header field is set to 108, IPComp Datagram,
      [RFC-1700].

All other IPv6 header fields are kept unchanged, including any
non-compressed header options.

Shacham, Monsour, Pereira, Thomas                            [Page 4]

The IPComp header is placed in an IPv6 packet using the same rules as
the IPv6 Fragment Header.  However if an IPv6 packet contains both an
IPv6 Fragment Header and an IPComp header, the IPv6 Fragment Header
MUST precede the IPComp header in the packet.

## 3.3.  IPComp Header Structure

The four-octet header has the following structure:

```
    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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |   Next Header   |     Flags     |  Compression Parameter Index |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   Next Header

        8-bit selector.  Stores the IPv4 Protocol field or the IPv6
        Next Header field of the original IP header.

   Flags

        8-bit field.  Reserved for future use.  MUST be set to zero.
        MUST be ignored by the receiving node.

   Compression Parameter Index (CPI)

        16-bit index.  The CPI is stored in network order.  The values
        0-63 define well-known compression algorithms, which require no
        additional information, and are used for manual setup.  The
        values themselves are identical to IPCOMP Transform identifiers
        as defined in [SECDOI].  Consult [SECDOI] for an initial set of
        defined values and for instructions on how to assign new values.
        The values 64-61439 are negotiated between the two nodes in
        definition of an IPComp Association, as defined in section 4.
        Note:  When negotiating one of the well-known algorithms, the
        nodes MAY select a CPI in the pre-defined range 0-63.  The
        values 61440-65535 are for private use among mutually consenting
        parties.
        Both nodes participating can select a CPI value independently of
        each other and there is no relationships between the two
        separately chosen CPIs.  The outbound IPComp header MUST use the
        CPI value chosen by the decompressing node.  The CPI in
        combination with the destination IP address uniquely identifies
        the compression algorithm characteristics for the datagram.

4. IPComp Association (IPCA) Negotiation

   To utilize the IPComp protocol, two nodes MUST first establish an
   IPComp Association (IPCA) between them.  The IPCA includes all
   required information for the operation of IPComp, including the
   Compression Parameter Index (CPI), the mode of operation, the
   compression algorithm to be used, and any required parameter for the
   selected compression algorithm.  The IPComp mode of operation is

either a node-to-node policy where IPComp is applied to every IP
packet between the nodes, or an ULP session based policy where only
selected ULP sessions between the nodes are using IPComp.  For each
IPCA, a different compression algorithm may be negotiated in each
direction, or only one direction may be compressed.  The default is
"no IPComp compression".

The IPCA is established by dynamic negotiations or by manual
configuration.  The dynamic negotiations SHOULD use the Internet
Security Association and Key Management Protocol [ISAKMP], where
IPSec is present.  The dynamic negotiations MAY be implemented
through a different protocol.

## 4.1. Use of ISAKMP

For IPComp in the context of IP Security, ISAKMP provides the
necessary mechanisms to establish IPCA.  IPComp Association is
negotiated by the initiator using a Proposal Payload, which would
include one or more Transform Payloads.  The Proposal Payload would
specify a compression protocol in the protocol id field and each
Transform Payload would contain the specific compression method(s)
being offered to the responder.

In the Internet IP Security Domain of Interpretation (DOI), IPComp is
negotiated as the Protocol ID PROTO_IPCOMP.  The compression
algorithm is negotiated as one of the defined IPCOMP Transform
Identifiers.

## 4.2. Use of Non-ISAKMP Protocol

The dynamic negotiations MAY be implemented through a protocol other
than ISAKMP.  Such protocol is beyond the scope of this document.

## 4.3. Manual Configuration

Nodes may establish IPComp Associations using manual configuration.
For this method, a limited number of Compression Parameters Indexes
(CPIs) is designated to represent a list of specific compression
methods.

## 5. Security Considerations

When IPComp is used in the context of IPSec, it is believed not to
have an effect on the underlying security functionality provided by
the IPSec protocol; i.e., the use of compression is not known to
degrade or alter the nature of the underlying security architecture
or the encryption technologies used to implement it.

When IPComp is used without IPSec, IP payload compression potentially
reduces the security of the Internet, similar to the effects of IP

encapsulation [RFC-2003].  For example, IPComp may make it difficult
for border routers to filter datagrams based on header fields.  In
particular, the original value of the Protocol field in the IP header
is not located in its normal positions within the datagram, and any

transport layer header fields within the datagram, such as port
numbers, are neither located in their normal positions within the
datagram nor presented in their original values after compression.  A
filtering border router can filter the datagram only if it shares the
IPComp Association used for the compression.  To allow this sort of
compression in environments in which all packets need to be filtered
(or at least accounted for), a mechanism must be in place for the
receiving node to securely communicate the IPComp Association to the
border router.  This might, more rarely, also apply to the IPComp
Association used for outgoing datagrams.

6. References

   [RFC-0791] Postel, J., Editor, "Internet Protocol", STD 5, RFC 791,
              September 1981.

   [RFC-1700] Reynolds, J., Postel, J., "Assigned Numbers", RFC 1700,
              October 1994.

   [RFC-1883] Deering, S., Hinden, R., "Internet Protocol, Version 6
              (IPv6) Specification", RFC 1883, April 1996.

   [RFC-1962] Rand, D., "The PPP Compression Control Protocol (CCP)",
              RFC 1962, June 1996.

   [RFC-2003] Perkins, C., "IP Encapsulation within IP", RFC 2003,
              October 1996.

   [RFC-2119] Bradner, S., "Key words for use in RFCs to Indicate
              Requirement Levels", RFC 2119, March 1997.

   [ISAKMP]   Maughan, D., Schertler, M., Schneider, M., and Turner, J.,
              "Internet Security Association and Key Management
              Protocol (ISAKMP)", Internet-Draft:
              draft-ietf-ipsec-isakmp-08.txt, Work in Progress,
              July 1997.

   [SECDOI]   Piper, D., "The Internet IP Security Domain of
              Interpretation for ISAKMP", Internet-Draft:
              draft-ietf-ipsec-ipsec-doi-06.txt, Work in Progress,

November 1997.

[V42BIS]   CCITT, "Data Compression Procedures for Data Circuit
           Terminating Equipment (DCE) Using Error Correction
           Procedures", Recommendation V.42 bis, January 1990.

Authors' Information

   Abraham Shacham
   Cisco Systems
   101 Cooper Street
   Santa Cruz, California 95060
   United States of America
   Email: shacham@cisco.com

   Robert Monsour
   Hi/fn Inc.
   2105 Hamilton Avenue, Suite 230
   San Jose, California 95125
   United States of America
   Email: rmonsour@hifn.com

   Roy Pereira
   TimeStep Corporation
   362 Terry Fox Drive
   Kanata, Ontario K2K 2P5
   Canada
   Email: rpereira@timestep.com

   Matt Thomas
   AltaVista Internet Software
   30 Porter Road
   Littleton, Massachusetts 01460
   United States of America
   Email: matt.thomas@altavista-software.com

Working Group

   The IP Payload Compression Protocol (IPPCP) working group can be
   contacted through its chair:

   Naganand Dorswamy
   Bay Networks
   Email: naganand@baynetworks.com