Network Working Group
Request for Comments: 3173
Obsoletes: 2393
Category: Standards Track

A. Shacham
Juniper
B. Monsour
Consultant
R. Pereira
Cisco
M. Thomas
Consultant

September 2001

IP Payload Compression Protocol (IPComp)

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2001). All Rights Reserved.

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 [RFC1962]) 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.

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 [RFC2119].

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 [RFC2460], 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 [RFC0791], the compression is applied to the payload of the IP datagram, starting at the first octet following the IP header, and continuing through the last octet of the datagram. No portion of the IP header or the IP header options is compressed. Note: In the case of an encapsulated IP header (e.g., tunnel mode encapsulation in IPsec), the datagram payload is defined to start immediately after the outer IP header; accordingly, the inner IP header is considered part of the payload and 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 an 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 payload and the IPComp header, as defined in section 3, is not smaller than the size of the original 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 the 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 several IP datagrams, say k, are sent without attempting compression. If then the next j datagrams also fail to compress, a larger number of datagrams, say k+n, 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, [RFC1700].

Header Checksum

The Internet Header checksum [RFC0791] 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, [RFC1700].

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

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. Note: Other IPv6 headers may be present between the IPv6 Fragment Header and the IPComp header.

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 designate 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 64-255 are reserved for future use. The values 256-61439 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 relationship 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 policy for establishing IPComp may be either a node-to-node policy where IPComp is applied to every IP packet between the nodes, or a session-based policy where only selected sessions between the nodes employ IPComp.

Two nodes may choose to negotiate IPComp in either or both directions, and they may choose to employ a different compression algorithm in each direction. The nodes MUST, however, negotiate a compression algorithm in each direction for which they establish an IPCA: there is no default compression algorithm.

No compression algorithm is mandatory for an IPComp implementation.

The IPCA is established by dynamic negotiations or by manual configuration. The dynamic negotiations SHOULD use the Internet Key Exchange protocol [IKE], where IPsec is present. The dynamic negotiations MAY be implemented through a different protocol.

4.1. Use of IKE

For IPComp in the context of IP Security, IKE provides the necessary mechanisms and guidelines for establishing IPCA. Using IKE, IPComp can be negotiated as stand-alone or in conjunction with other IPsec protocols.

An IPComp Association is negotiated by the initiator using a Proposal Payload, which includes one or more Transform Payloads. The Proposal Payload specifies the IP Payload Compression Protocol in the protocol ID field and each Transform Payload contains the specific compression algorithm(s) being offered to the responder.

The CPI is sent in the SPI field of the proposal, with the SPI size field set to match. The CPI SHOULD be sent as a 16-bit number, with the SPI size field set to 2. Alternatively, the CPI MAY be sent as a 32-bit value, with the SPI size field set to 4. In this case, the 16-bit CPI number MUST be placed in the two least significant octets of the SPI field, while the two most significant octets MUST be set to zero, and MUST be ignored by the receiving node. The receiving node MUST be able to process both forms of the CPI proposal.

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.

The following attributes are applicable to IPComp proposals:

Encapsulation Mode

To propose a non-default Encapsulation Mode (such as Tunnel Mode), an IPComp proposal MUST include an Encapsulation Mode attribute. If the Encapsulation Mode is unspecified, the default value of Transport Mode is assumed.

Lifetime

An IPComp proposal uses the Life Duration and Life Type attributes to suggest life duration to the IPCA.

When IPComp is negotiated as part of a Protection Suite, all the logically related offers must be consistent. However, an IPComp proposal SHOULD NOT include attributes that are not applicable to IPComp. An IPComp proposal MUST NOT be rejected because it does not include attributes of other protocols in the Protection Suite that are not relevant to IPComp. When an IPComp proposal includes such attributes, those attributes MUST be ignored when setting the IPCA, and therefore ignored in the operation of IPComp.

Implementation note:

A node can avoid the computation necessary for determining the compression algorithm from the CPI if it is using one of the well-known algorithms; this can save time in the decompression process. A node can do this by negotiating a CPI equal in value to the pre-defined Transform identifier of that compression algorithm. Specifically: A node MAY offer a CPI in the pre-defined range by sending a Proposal Payload that MUST contain a single Transform Payload, which is identical to the CPI. When proposing two or more Transform Payloads, a node MAY offer CPIs in the pre-defined range by using multiple IPComp proposals -- each MUST include a single Transform Payload. To clarify: If a Proposal Payload contains two or more Transform Payloads, the CPI MUST be in the negotiated range. A receiving node MUST be able to process each of these proposal forms.

Implementation note:

IPCAs become non-unique when two or more IPComp sessions are established between two nodes, and the same well-known CPI is used in at least two of the sessions. Non-unique IPCAs pose problems in maintaining attributes specific to each IPCA, either negotiated (e.g., lifetime) or internal (e.g., the counters of the adaptive algorithm for handling previously compressed payload). To ensure the uniqueness of IPCAs between two nodes, when two or more of the IPCAs use the same compression algorithm, the CPIs SHOULD be in the negotiated range. However, when the IPCAs are not required to be unique, for example when no attribute is being utilized for these IPCAs, a well-known CPI MAY be used. To clarify: When only a single session using a particular well-known CPI is established between two nodes, this IPCA is unique.

4.2. Use of Non-IKE Protocol

The dynamic negotiations MAY be implemented through a protocol other than IKE. Such a 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 [RFC2003]. 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. IANA Considerations

This document does not require any IANA actions. The well-known numbers used in this document are defined elsewhere; see [SECDOI].

7. Changes made since RFC 2393

This section summarizes the changes in this document from RFC 2393 of which an implementer of RFC 2393 should be aware. All the changes are meant to clarify the negotiation of an IPComp Association (IPCA) using IKE [IKE] in the context of IPsec.

1) Added a clarification that IPComp can be negotiated stand-alone or bundled with other protocols in a Protection Suite.

2) Defined the CPI in the SPI field of an IKE proposal: two-octet field is a SHOULD, four-octet a MAY. Defined the placement of the 16-bit CPI in a four-octet field. Specified that a receiver MUST process both field sizes.

3) Added wording to define the default Encapsulation Mode to be Transport Mode. Required that an IPComp proposal MUST include an Encapsulation Mode attribute when it suggests a non-default encapsulation, such as Tunnel Mode.

4) Added the Lifetime attribute to the list of supported attributes (along with Transport Mode).

5) Specified the handling of attributes of transforms in a Protection Suite that are not applicable to IPComp: These attributes SHOULD NOT be included in an IPComp proposal and MUST be ignored when setting IPCA and in the operation of IPComp. IPComp implementations MUST never reject an IPCOMP proposal that does not include attributes of other transforms.

6) Added implementation notes on the negotiation and usage of CPIs in the predefined (well-known) range.

8. References

[RFC0791] Postel, J., Editor, "Internet Protocol", STD 5, RFC 791, September 1981.

[RFC1700] Reynolds, J. and J. Postel, "Assigned Numbers", STD 2, RFC 1700, October 1994. Or see: http://www.iana.org/numbers.html

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.

[RFC1962] Rand, D., "The PPP Compression Control Protocol (CCP)", RFC 1962, June 1996.

[RFC2003] Perkins, C., "IP Encapsulation within IP", RFC 2003, October 1996.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[IKE] Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

[SECDOI] Piper, D., "The Internet IP Security Domain of Interpretation for ISAKMP", RFC 2407, November 1998.

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

Authors’ Addresses

Abraham Shacham
Juniper Networks, Inc.
1194 North Mathilda Avenue
Sunnyvale, California 94089
United States of America

EMail: shacham@shacham.net

Bob Monsour
18 Stout Road
Princeton, New Jersey 08540
United States of America

EMail: bob@bobmonsour.com

Roy Pereira
Cisco Systems, Inc.
55 Metcalfe Street
Ottawa, Ontario K1P 6L5
Canada

EMail: royp@cisco.com

Matt Thomas
3am Software Foundry
8053 Park Villa Circle
Cupertino, California 95014
United States of America

EMail: matt@3am-software.com

Comments

Comments should be addressed to the ippcp@external.cisco.com mailing list and/or the author(s).

Full Copyright Statement

Copyright (C) The Internet Society (2001). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.