M. Sabin, Consultant
R. Monsour, Hi/fn Inc.

LZS Payload Compression Transform for ESP
<draft-sabin-lzs-payload-01.txt>

Status of this Memo

It is intended that a future version of this draft be submitted to the IESG for publication as an Informational RFC. Comments about this draft should be submitted to the authors or to the IPSEC mailing list (ipsec@tis.com).

Abstract

This memo proposes a "payload compression transform" based on the LZS compression algorithm. The transform can be used to compress the payload field of an IP datagram that uses the Encapsulating Security Payload (ESP) format. The compression transform proposed here is stateless, meaning that a datagram can be decompressed independently of any other datagram. Compression is performed prior to the encryption operation of ESP, which has the side benefit of reducing the amount of data that must be encrypted.

This memo anticipates a forthcoming ESP document that will supercede [Atkins96]. The forthcoming document will allow for ESP payloads to be compressed via transforms such as the one described in this memo.

Acknowledgments

The LZS details presented here are similar to those in "PPP Stac LZS Compression Protocol," by R. Friend and W. A. Simpson [RFC-1974].

The authors wish to thank the many participants of the IPSEC mailing list whose discussion made possible the architecture for integrating compression with ESP.

Table of Contents

1. Introduction

Encrypted data is random in nature and not compressible. When an IP datagram is encrypted, compression methods used at lower protocol layers -- e.g., PPP compression [RFC-1962] -- no longer work. If both compression and encryption are desired, compression must be performed first.

A side benefit of compressing the data first is that the amount of data which must be encrypted is reduced. In some implementations, compression is done in hardware and encryption is done in software, and this can represent a significant reduction in software overhead.

The Encapsulating Security Payload (ESP) format is well suited to combining compression with encryption, for a variety of reasons:

- ESP is the place were encryption is applied to an IP datagram. It is straightforward to precede the encryption step by an optional compression step. The compression step transforms an uncompressed ESP payload into a compressed ESP payload. This "payload compression transform" can be independently defined and

used with any ESP transform.

- ESP provides a security parameters index (SPI) that links a datagram to security parameters negotiated elsewhere. A destination uses the SPI to look up the ESP transform needed to decode an incoming datagram. If compression details are included

among the negotiated parameters, a destination can also use the SPI to look up the compression transform needed to decode the ESP payload.

This memo proposes a payload compression transform based on the LZS compression algorithm. The transform can be used to compress any ESP payload. The transform is stateless, meaning that the payload of a datagram can be decompressed independently of any other datagram. This is important in IP, where the delivery of individual datagrams is not guaranteed.

1.1 Background of LZS Compression

The LZS algorithm is a lossless compression method that uses a sliding window of 2,048 bytes. During compression, redundant sequences of data are replaced with tokens that represent those sequences. During decompression, the original sequences are substituted for the tokens, in such a way that the original data is exactly recovered. LZS differs from lossy schemes, such as those often used for video compression, that do not exactly reproduce the original data.

Details of LZS formatting can be found in [ANSI94].

The efficiency of the LZS algorithm depends on the degree of redundancy in the original data. A typical compression ratio is 2:1. LZS achieves a compression ratio of 2.34:1 on the University of Calgary Text Compression Corpus [Calgary].

1.2 Licensing

Hi/fn, Inc., holds patents on the LZS algorithms. A restricted license reference implementation is available for use in IPSEC applications at no cost. Source and object licenses are available on a non-discriminatory basis. Hardware implementations are also available. For more information, contact Hi/fn at the address

listed with the authors' addresses.

1.3 Requirements Terminology

In this document, the words that are used to define the significance of each particular requirement are usually capitalized. These words are:

- MUST: This word, or the adjective "REQUIRED," means that the item is an absolute requirement of the specification.

- SHOULD: This word, or the adjective "RECOMMENDED," means

that there might exist valid reasons in particular circumstances to ignore this item, but the full implications should be understood and the case carefully weighed before taking a different course.

- MAY: This word, or the adjective "OPTIONAL," means that the item is truly optional. One vendor might choose to include the item because of a particular marketplace requirement or because it enhances the product, while another vendor might omit the item.

2. Format of Transformed Payload

The input to the payload compression transform is a payload to be encapsulated by ESP. The output of the transform is a new payload. The output payload contains the input payload's data in either compressed or uncompressed format. If the uncompressed format is used, the output payload is identical to the input payload. If the compressed format is used, the output payload consists of the input payload data, compressed and formatted according to [ANSI94].

The input and output payloads are each an integral number of bytes in length.

The sender MUST reset the compression history prior to processing each datagram's payload. This ensures that each datagram's payload can be decompressed independently of any other, as is needed when datagrams are received out of order.

The sender MUST flush the compressor each time it transmits a

compressed datagram. Flushing means that all data going into the compressor is included in the output, i.e., no data is held back in the hope of achieving better compression. Flushing is necessary to prevent a datagram's data from spilling over into a later datagram.

3. Compression Control Bit

The Compression Control (CC) bit is a single bit that indicates whether or not the payload is compressed. A value of 1 indicates compressed, and a value of 0 indicates uncompressed.

The CC bit is not part of the payload transform. We anticipate it being defined in the upcoming ESP document.

4. Compression Procedure

The compression procedure consists of the following steps:

i) The sender resets the compression history.

ii) The sender decides whether or not to compress the payload.

- If the sender chooses to compress the payload, the LZS algorithm is applied. The resulting compressed data is formatted according to [ANSI94]. The CC bit is set to 1.

- If the sender chooses not to compress the payload, the CC bit is cleared to 0.

An implementation SHOULD monitor the results of the payload compression operation and reject the operation if it results in expansion. In such a case, the uncompressed payload SHOULD be transmitted with the CC bit cleared to 0.

After the payload has been transformed by these steps, the transformed payload is submitted to the encode procedure of the selected ESP transform.

5. Decompression Procedure

Prior to applying the decompression procedure, the decode procedure of the selected ESP transform is applied to extract the payload.

The decompression procedure consists of the following step:

- The receiver checks the CC bit. If CC = 1, the LZS decompression algorithm is applied to the payload data. If CC = 0, decompression is not applied.

6. Additions to ISAKMP DOI

The Internet Security Association and Key Management Protocol (ISAKMP) defines a framework for negotiating security associations. The IPSEC Domain of Interpretation (DOI) for ISAKMP, described in [Piper], defines the attributes of a security association that can be negotiated.

In order to accommodate the negotiation of compression, we propose the following additions to section 4.5, "IPSEC Security Association Attributes," in [Piper]:

Attribute Classes

	class	value	type
>	Compression Algorithm	12	B

Class Values

> Compression Algorithm
> RESERVED 0
> LZS 1
>
> Values 2-61439 are reserved to IANA. Values 61440-65535 are
> for private use among mutually consenting parties.
>
> There is no default value for Compression Algorithm, as it
> must be specified to correctly identify the applicable
> transform.

7. Security Considerations

This memo discusses the use of lossless compression in a security

protocol, specifically, ESP. The proposed use of compression is believed to have no effect on the security of the encryption and authentication algorithms used in ESP, nor is it believed to have any effect on the underlying security architecure of IPSEC.

The use of compression does change the length of ESP payloads, in a manner that depends on the data prior to encryption. Thus, the use of compression may have an effect on the ability of an eavesdropper to glean information by analyzing the length of transmitted packets.

8. References

[ANSI94] American National Standards Institute, Inc., "Data Compression Method for Information Systems," ANSI X3.241-1994, August 1994.

[Atkins96] Atkinson, R., "IP Encapsulating Security Protocol," RFC-xxxx, June 1996.

[Calgary] Text Compression Corpus, University of Calgary, available at ftp://ftp.cpsc.ucalgary.ca/pub/projects/text.compression.corpus.

[Piper] Piper, D., "The Internet IP Security Domain of Interpretation for ISAKMP," work in progress, available at <draft-ietf-ipsec-doi-02.txt>.

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

[RFC-1974] Friend, R., and Simpson, W.A., "PPP Stac LZS Compression

Protocol," RFC-1974, August 1996.

9. Authors' Addresses

Michael Sabin
883 Mango Avenue
Sunnyvale, CA 94087
Email: mike.sabin@worldnet.att.net

Robert Monsour
Hi/fn Inc.

12636 High Bluff Drive
San Diego, CA 92130
Email: rmonsour@earthlink.net

10. Appendix: Compression Efficiency versus Datagram Size

The following table offers some guidance on the compression efficiency that can be achieved as a function of datagram size. Each entry in the table shows the compression ratio that was achieved when the proposed transform was applied to a test file using datagrams of a specified size.

The test file was the University of Calgary Text Compression Corpus [Calgary]. The length of the file prior to compression was 3,278,000 bytes. When the entire file was compressed as a single payload, a compression ratio of 2.34 resulted.

Datagram size, bytes	64	128	256	512	1024	2048	4096	8192	16384
Compression ratio	1.18	1.28	1.43	1.58	1.74	1.91	2.04	2.11	2.14