Network Working Group
Internet-Draft
Intended status: Informational
Expires: October 12, 2013

V. Smyslov
ELVIS-PLUS

April 10, 2013

IKEv2 Fragmentation
draft-smyslov-ipsecme-ikev2-fragmentation-01

Abstract

This document describes the way to avoid IP fragmentation of large IKEv2 messages. This allows IKEv2 messages to traverse network devices that don't allow IP fragments to pass through.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on October 12, 2013.

Table of Contents

1. Introduction 3
1.1. Conventions Used in This Document 3
2. Protocol details 4
2.1. Overview 4
2.2. Limitations 4
2.3. Negotiation 4
2.4. Using IKE Fragmentation 5
2.5. Fragmenting Message 6
2.5.1. Selecting Fragment Size 7
2.5.2. Fragmenting Messages containing unencrypted Payloads 8
2.6. Receiving IKE Fragment Message 9
2.6.1. Changes in Replay Protection Logic 10
3. Interaction with other IKE extensions 11
4. Security Considerations 12
5. IANA Considerations 13
6. References 14
6.1. Normative References 14
6.2. Informative References 14
Author's Address 15

1. Introduction

The Internet Key Exchange Protocol version 2 (IKEv2), specified in [RFC5996], uses UDP as a transport for its messages. When IKE message size exceed path MTU, it gets fragmented by IP level. The problem is that some network devices, specifically some NAT boxes, don't allow IP fragments to pass through. This apparently blocks IKE communication and, therefore, prevents peers from establishing IPsec SA.

The solution to the problem described in this document is to perform fragmentation of large messages by IKE itself, replacing them by series of smaller messages. In this case the resulting IP Datagrams will be small enough so that no fragmentation on IP level will take place.

1.1. Conventions Used in This Document

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 [RFC2119].

2. Protocol details

2.1. Overview

The idea of the protocol is to split large IKE message into the set of smaller ones, calling Fragment Messages. On the receiving side Fragment Messages are collected and merged together to get original message. In general this approach increases receiver's vulnerability to Denial of Service attack. To reduce this vulnerability Fragment Messages are individually encrypted and authenticated. This implies that message cannot be fragmented until shared secret is calculated.

2.2. Limitations

In general, original message can be fragmented if and only if it contains Encrypted Payload. It means that messages in IKE_SA_INIT Exchange cannot be fragmented. In most cases this is not a problem, since IKE_SA_INIT messages are usually small enough to avoid IP fragmentation. But in some cases (advertising a badly structured long list of algorithms, using large MODP Groups, etc.) those messages may become fairly large and get fragmented by IP level. In these cases the described solution won't help.

Another limitation is that the minimal size of IP Datagram bearing IKE Fragment Message is about 100 bytes depending on the algorithms employed. According to [RFC0791] the minimum IP Datagram size that is guaranteed not to be further fragmented is 68 bytes. So, even the smallest IKE Fragment Messages could be fragmented by IP level in some circumstances. But such extremely small PMTU sizes are very rare in real life.

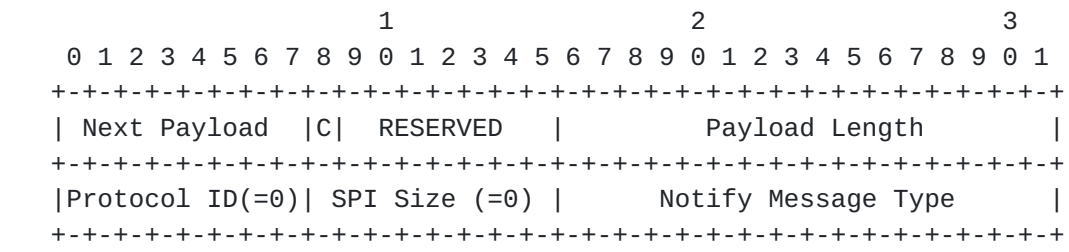
2.3. Negotiation

Initiator MAY indicate its support for IKE Fragmentation and willingness to use it by including Notification Payload of type IKE_FRAGMENTATION_SUPPORTED in IKE_SA_INIT request message. If Responder also supports this extension and is willing to use it, it includes this notification in response message.

```
Initiator                   Responder
-----------                 -----------
HDR, SAi1, KEi, Ni,
   [N(IKE_FRAGMENTATION_SUPPORTED)]  -->

                    <--   HDR, SAr1, KEr, Nr, [CERTREQ],
                               [N(IKE_FRAGMENTATION_SUPPORTED)]
```

The Notify payload is formatted as follows:



- Protocol ID (1 octet) MUST be 0.

- SPI Size (1 octet) MUST be 0, meaning no SPI is present.

- Notify Message Type (2 octets) - MUST be xxxxx, the value assigned for IKE_FRAGMENTATION_SUPPORTED by IANA.

This Notification contains no data.

2.4. Using IKE Fragmentation

After IKE Fragmentation is negotiated, it is up to Initiator of each Exchange, whether to use it or not. In most cases IKE Fragmentation will be used in IKE_AUTH Exchange, especially if certificates are employed. Initiator may first try to send unfragmented message and resend it fragmented only if it didn't receive response after several retransmissions, or it may always send messages fragmented (but see Section 3), or it may fragment only large messages and messages causing large responses.

In general the following guidelines are applicable:

- Initiator MAY fragment outgoing message if it suspects that either request or response message may be fragmented by IP level.

- Initiator SHOULD fragment outgoing message if it suspects that either request or response message may be fragmented by IP level and IKE Fragmentation was already used in one of previous Exchanges in the context of the current IKE SA.

- Initiator SHOULD NOT fragment outgoing message if both request and response messages of the Exchange are small enough not to cause fragmentation on IP level (for example, there is no point in fragmenting Liveness Check messages).

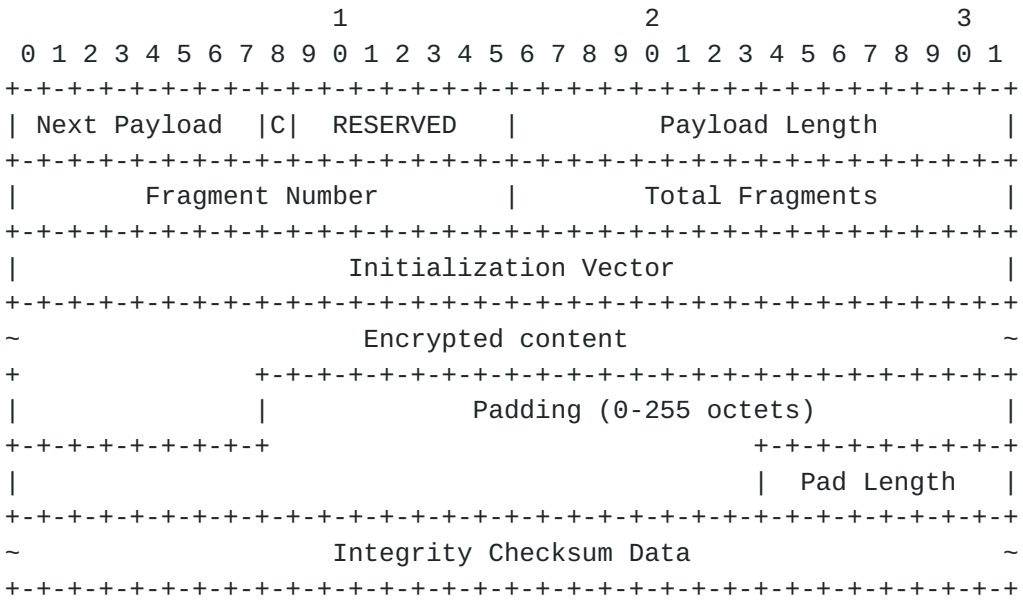
Responder MUST send response message in the same form (fragmented or not) as corresponded request message. If it received unfragmented request message, responded with unfragmented response message and then received fragmented retransmission of the same request, it MUST resend its response back to Initiator fragmented.

2.5. Fragmenting Message

Message to be fragmented MUST contain Encrypted Payload. For the purpose of IKE Fragment Messages construction original (unencrypted) content of Encrypted Payload is broken down into parts. Its content is treated as a binary blob and is broken down regardless of inner Payloads boundaries. Each of resulting parts is treated as a content for Encrypted Fragment Payload.

The Encrypted Fragment Payload, denoted SKF{...}, contains other payloads in encrypted form. The Encrypted Fragment Payload, as well as Encrypted Payload from [RFC5996], if present in a message, MUST be the last payload in the message.

The payload type for an Encrypted Fragment payload is XXX (TBA by IANA).



Encrypted Fragment Payload

- Next Fragment (1 octet) - in the very first fragment MUST be set to Payload Type of the first inner Payload (as in Encrypted Payload). In the rest fragments MUST be set to zero.

- Fragment Number (2 octets) - current fragment number starting from 1. This field MUST be less than or equal to the next field, Total Fragments.

- Total Fragments (2 octets) - number of fragments original message was divided into. This field MUST NOT be zero.

Other fields are identical to those specified in Section 3.14 of [RFC5996].

When prepending IKE Header, Length field MUST be adjusted to reflect the length of constructed message and Next Payload field MUST reflect payload type of the first Payload in the constructed message (that in most cases will be Encrypted Fragment Payload). All newly constructed messages MUST retain the same Message ID as original message. After prepending IKE Header and possibly any of Payloads that precedes Encrypted Payload in original message (see Section 2.5.2), the resulting messages are sent to the peer.

Below is an example of fragmenting some message.

```
HDR(MID=n), SK(NextPld=PLD1) {PLD1 ... PLDN}
```

Original Message

```
HDR(MID=n), SKF(NextPld=PLD1, Frag#=1, TotalFrags=m) {...},
HDR(MID=n), SKF(NextPld=0, Frag#=2, TotalFrags=m) {...},
...
HDR(MID=n), SKF(NextPld=0, Frag#=m, TotalFrags=m) {...}
```

IKE Fragment Messages

2.5.1. Selecting Fragment Size

When breaking content of Encrypted Payload down into parts sender SHOULD chose size of those parts so, that resulting IP Datagram size not exceed some fragmentation threshold - be small enough to avoid IP fragmentation.

If sender has some knowledge about PMTU size it MAY use it. If sender is a Responder in the Exchange and it has received fragmented request, it MAY use maximum size of received IKE Fragment Message IP Datagrams as threshold when constructing fragmented response.

Otherwise for messages to be sent over IPv6 it is RECOMMENDED to use value 1280 bytes as a maximum IP Datagram size ([RFC2460]). For messages to be sent over IPv4 it is RECOMMENDED to use value 576 bytes as a maximum IP Datagram size.

For IPv4 Encrypted Payload content size is less than IP Datagram size by the sum of the following values:

- IPv4 header size (typically 20 bytes, up to 60 if IP options are present)

- UDP header size (8 bytes)
- non-ESP marker size (4 bytes if present)
- IKE Header size (28 bytes)
- Encrypted Payload header size (4 bytes)
- IV size (varying)
- padding and its size (at least 1 byte)
- ICV size (varying)

The sum may be estimated as 61..105 bytes + IV + ICV + padding. For IPv6 this estimation is difficult as there may be varying IPv6 Extension headers included.

According to [RFC0791] the minimum IPv4 datagram size that is guaranteed not to be further fragmented is 68 bytes, but it is generally impossible to use such small value for solution, described in this document. Using 576 bytes is a compromise - the value is large enough for the presented solution and small enough to avoid IP fragmentation in most situations. Sender MAY use other values if they are appropriate.

Initiator MAY try to discover path MTU by using several values of fragmentation threshold, provided that it starts with larger values and fragments message again with next smaller value if it doesn't receive response in a reasonable time after several retransmissions. In this case using next smaller value MUST result in increasing Total Fragments field.

2.5.2. Fragmenting Messages containing unencrypted Payloads

Currently no one of IKEv2 Exchanges defines messages, containing both unencrypted payloads and payloads, protected by Encrypted Payload. But IKEv2 doesn't forbid such messages. If some future IKEv2 extension defines such a message and it needs to be fragmented, all unprotected payloads MUST be in the first fragment, along with Encrypted Fragment Payload, which MUST be present in any IKE Fragment Message.

Below is an example of fragmenting message, containing both encrypted and unencrypted Payloads.

HDR(MID=n), PLD0, SK(NextPld=PLD1) {PLD1 ... PLDN}

Original Message

```
HDR(MID=n), PLD0, SKF(NextPld=PLD1, Frag#=1, TotalFrags=m) {...},
HDR(MID=n), SKF(NextPld=0, Frag#=2, TotalFrags=m) {...},
...
HDR(MID=n), SKF(NextPld=0, Frag#=m, TotalFrags=m) {...}
```

IKE Fragment Messages

Note, that the size of each IP Datagram bearing IKE Fragment Messages SHOULD not exceed fragmentation threshold, including the very first, which contains unprotected Payloads. This will reduce the size of Encrypted Fragment Payload content in the first IKE Fragment Message to accommodate unprotected Payloads. In extreme cases Encrypted Fragment Payload will contain no data, but it is still MUST be present in the message, because only its presence allows receiver to distinguish IKE Fragment Message from regular IKE message.

2.6. Receiving IKE Fragment Message

Receiver identifies IKE Fragment Message by the presence of Encrypted Fragment Payload in it. Note, that it is possible for this payload to be not the first (and the only) payload in the message (see Section 2.5.2). But for all currently defined IKEv2 exchanges this payload will be the first and the only payload in the message.

Upon receiving IKE Fragment Message the following actions are performed:

- Check message validity - in particular, check whether values of Fragment Number and Total Fragments in Encrypted Fragment Payload are valid. If not - message MUST be silently discarded.

- Check, that this IKE Fragment Message is new for the receiver and not a replay. If IKE Fragment message with the same Message ID, same Fragment Number and same Total Fragments fields was already received and successfully processed, this message is considered a replay and MUST be discarded.

- Verify IKE Fragment Message authenticity by checking ICV in Encrypted Fragment Payload. If ICV check fails message MUST be silently discarded.

- If reassembling isn't finished yet and Total Fragments field in received IKE Fragment Message is greater than this field in previously received fragments, receiver MUST discard all received fragments and start reassembling over with just received IKE

Fragment Message.

- Store message in the list waiting for the rest of fragments to arrive.

When all IKE Fragment Messages (as indicated in the Total Fragments field) are received, content of their Encrypted Fragment Payloads is decrypted and merged together to form content of original Encrypted Payload, and, therefore, along with IKE Header, original message. Then it is processed as if it was received, verified and decrypted as as regular unfragmented message.

2.6.1. Changes in Replay Protection Logic

According to [RFC5996] IKEv2 MUST reject message with the same Message ID as it has seen before (taking into consideration Response bit). This logic has already been updated by [RFC6311], which deliberately allows any number of messages with zero Message ID. This document also updates this logic: if message contains Encrypted Fragment Payload, the values of Fragment Number and Total Fragments fields from this payload MUST be used along with Message ID to detect retransmissions and replays.

If Responder receives IKE Fragment Message after it received, successfully verified and processed regular message with the same Message ID, it means that response message didn't reach Initiator and it activated IKE Fragmentation. If Fragment Number in Encrypted Fragment Payload in this message is equal to 1, Responder MUST fragment its response and retransmit it back to Initiator in fragmented form.

If Responder receives a replay IKE Fragment Message for already reassembled, verified and processed fragmented message, it MUST retransmit response back to Initiator, but only if Fragment Number field in Encrypted Fragment Payload is equal to 1 and MUST silently discard received message otherwise.

3. Interaction with other IKE extensions

IKE Fragmentation is compatible with most of defined IKE extensions, like IKE Session Resumption [RFC5723], Quick Crash Detection Method [RFC6290] and so on. It neither affect their operation, nor is affected by them. It is believed that IKE Fragmentation will also be compatible with most future IKE extensions, if they follow general principles of formatting, sending and receiving IKE messages, described in [RFC5996].

The notable exception that requires a special care is [RFC6311] - Protocol Support for High Availability of IKEv2. As it deliberately allows any number of synchronization Exchanges to have the same Message ID - zero, standard replay detection logic, based on checking Message ID is not applicable for such messages, and receiver has to check message content to detect replays. When implementing IKE Fragmentation along with [RFC6311], IKE Message ID Synchronization messages MUST NOT be sent fragmented to simplify receiver's task of detecting replays. Fortunately, these messages are small and there is no point in fragmenting them anyway.

4. Security Considerations

Most of the security considerations for IKE Fragmentation are the same as those for base IKEv2 protocol described in [RFC5996]. This extension introduces Encrypted Fragment Payload to protect content of IKE Message Fragment. This allows receiver to individually check authenticity of fragments, thus protecting itself from Denial of Service attack.

5. IANA Considerations

This document defines new Payload in the "IKEv2 Payload Types" registry:

<TBA>	Encrypted Fragment Payload	SKF

This document also defines new Notify Message Types in the "Notify Messages Types - Status Types" registry:

<TBA>	IKE_FRAGMENTATION_SUPPORTED

6. References

6.1. Normative References

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

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

[RFC6311] Singh, R., Kalyani, G., Nir, Y., Sheffer, Y., and D. Zhang, "Protocol Support for High Availability of IKEv2/ IPsec", RFC 6311, July 2011.

6.2. Informative References

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

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

[RFC5723] Sheffer, Y. and H. Tschofenig, "Internet Key Exchange Protocol Version 2 (IKEv2) Session Resumption", RFC 5723, January 2010.

[RFC6290] Nir, Y., Wierbowski, D., Detienne, F., and P. Sethi, "A Quick Crash Detection Method for the Internet Key Exchange Protocol (IKE)", RFC 6290, June 2011.

Author's Address

Valery Smyslov
ELVIS-PLUS
PO Box 81
Moscow (Zelenograd) 124460
RU

Phone: +7 495 276 0211
Email: svan@elvis.ru