Network Working Group   V. Smyslov
Internet-Draft   ELVIS-PLUS
Intended status: Standards Track   December 3, 2018
Expires: June 6, 2019

Intermediate Exchange in the IKEv2 Protocol
draft-smyslov-ipsecme-ikev2-aux-02

Abstract

This documents defines a new exchange, called Intermediate Exchange, for the Internet Key Exchange protocol Version 2 (IKEv2). This exchange can be used for transferring large amount of data in the process of IKEv2 Security Association (SA) establishment. Introducing Intermediate Exchange allows re-using existing IKE Fragmentation mechanism, that helps to avoid IP fragmentation of large IKE messages, but cannot be used in the initial IKEv2 exchange.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The Internet Key Exchange protocol version 2 (IKEv2) defined in [RFC7296] uses UDP as a transport for its messages. If size of the messages is large enough, IP fragmentation takes place, that may interfere badly with some network devices. The problem is described in more detail in [RFC7383], which also defines an extension to the IKEv2 called IKE Fragmentation. This extension allows IKE messages to be fragmented at IKE level, eliminating possible issues caused by IP fragmentation. However, the IKE Fragmentation cannot be used in the initial IKEv2 exchange, IKE_SA_INIT. This limitation in most cases is not a problem, since the IKE_SA_INIT messages used to be small enough not to cause IP fragmentation.

Recent progress in Quantum Computing has brought a concern that classical Diffie-Hellman key exchange methods will become insecure in a relatively near future and should be replaced with Quantum Computer (QC) resistant ones. Currently most of QC-resistant key exchange methods have large public keys. If these keys are exchanged in the IKE_SA_INIT, then most probably IP fragmentation will take place, therefore all the problems caused by it will become inevitable.

A possible solution to the problem would be to use TCP as a transport for IKEv2, as defined in [RFC8229]. However this approach has

significant drawbacks and is intended to be a "last resort" when UDP transport is completely blocked by intermediate network devices.

This document defines a new exchange for the IKEv2 protocol, called Intermediate Exchange or INTERMEDIATE. One or more these exchanges may take place right after the IKE_SA_INIT exchange and prior to the IKE_AUTH exchange. The INTERMEDIATE exchange messages can be fragmented using IKE Fragmentation mechanism, so these exchanges may be used to transfer large amounts of data which don't fit into the IKE_SA_INIT exchange without causing IP fragmentation.

While ability to transfer large public keys of QC-resistant key exchange methods is a primary motivation for introducing of the Intermediate Exchange, its application is not limited to this use case. This exchange may be used whenever some data need to be transferred before the IKE_AUTH exchange and for some reason the IKE_SA_INIT exchange is not suited for this purpose. This document defines the INTERMEDIATE exchange without tying it to any specific use case. It is expected that separate specifications will define for which purposes and how the INTERMEDIATE exchange is used in the IKEv2.

2. Terminology and Notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Intermediate Exchange Details

3.1. Support for Intermediate Exchange Negotiation

The initiator indicates its support for Intermediate Exchange by including a notification of type INTERMEDIATE_EXCHANGE_SUPPORTED in the IKE_SA_INIT request message. If the responder also supports this exchange, it includes this notification in the response message.

```
Initiator                                  Responder
-----------                                -----------
HDR, SAi1, KEi, Ni,
[N(INTERMEDIATE_EXCHANGE_SUPPORTED)] -->
                                      <--  HDR, SAr1, KEr, Nr, [CERTREQ],
                                     [N(INTERMEDIATE_EXCHANGE_SUPPORTED)]
```

The INTERMEDIATE_EXCHANGE_SUPPORTED is a Status Type IKEv2 notification. Its Notify Message Type is <TBA by IANA>. Protocol ID

and SPI Size are both set to 0. This specification doesn't define any data this notification may contain, so the Notification Data is left empty. However, future enhancements of this specification may override this. Implementations MUST ignore the non-empty Notification Data if they don't understand its purpose.

3.2. Using Intermediate Exchange

If both peers indicated their support for the Intermediate Exchange, the initiator may use one or more these exchanges to transfer additional data. Using the INTERMEDIATE exchange is optional, the initiator may find it unnecessary after completing the IKE_SA_INIT exchange.

The Intermediate Exchange is denoted as INTERMEDIATE, its Exchange Type is <TBA by IANA>.

```
Initiator                                  Responder
-----------                                -----------
HDR, ..., SK {...}  -->
                                      <--  HDR, ..., SK {...}
```

The initiator may use several INTERMEDIATE exchanges if necessary. Since initiator's Window Size is initially set to one (Section 2.3 of [RFC7296]), these exchanges MUST follow each other and MUST all be completed before the IKE_AUTH exchange is initiated. The IKE SA MUST NOT be considered as established until the IKE_AUTH exchange is successfully completed.

The Message IDs for the INTERMEDIATE exchanges MUST be chosen according to the standard IKEv2 rule, described in the Section 2.2. of [RFC7296], i.e. it is set to 1 for the first INTERMEDIATE exchange, 2 for the next (if any) and so on. The message ID for the first pair of the IKE_AUTH messages is one more than the one that was used in the last INTERMEDIATE exchange.

If the presence of NAT is detected in the IKE_SA_INIT exchange via NAT_DETECTION_SOURCE_IP and NAT_DETECTION_DESTINATION_IP notifications, then the peers MUST switch to port 4500 immediately once this exchange is completed, i.e. in the first INTERMEDIATE exchange.

The content of the INTERMEDIATE exchange messages depends on the data being transferred and will be defined by specifications utilizing this exchange. However, since the main motivation for the INTERMEDIATE exchange is to avoid IP fragmentation when large amount of data need to be transferred prior to IKE_AUTH, the Encrypted payload MUST be present in the INTERMEDIATE exchange messages and

payloads containing large data MUST be placed inside. This will allow IKE Fragmentation [RFC7383] to take place, provided it is supported by the peers and negotiated in the initial exchange.

3.3. The INTERMEDIATE Exchange Protection and Authentication

3.3.1. Protection of the INTERMEDIATE Messages

The keys SK_e[i/r] and SK_a[i/r] for the Encrypted payload in the INTERMEDIATE exchanges are computed in a standard fashion, as defined in the Section 2.14 of [RFC7296]. Every subsequent INTERMEDIATE exchange uses the most recently calculated keys before this exchange is started. The first INTERMEDIATE exchange always uses SK_e[i/r] and SK_a[i/r] keys that were computed as result the IKE_SA_INIT exchange. If this INTERMEDIATE exchange performs additional key exchange resulting in the update of SK_e[i/r] and SK_a[i/r], then these updated keys are used for encryption and authentication of next INTERMEDIATE exchange, otherwise the current keys are used, and so on.

3.3.2. Authentication of the INTERMEDIATE Exchanges

The data transferred in the INTERMEDIATE exchanges must be authenticated in the IKE_AUTH exchange. For this purpose the definition of the blob to be signed (or MAC'ed) from the Section 2.15 of [RFC7296] is modified as follows:

```
InitiatorSignedOctets = RealMsg1 | NonceRData | MACedIDForI [| IntAuth]
ResponderSignedOctets = RealMsg2 | NonceIData | MACedIDForR [| IntAuth]

IntAuth =  IntAuth_1 | [| IntAuth_2 [| IntAuth_3]] ...

IntAuth_1 = IntAuth_1_I | IntAuth_1_R
IntAuth_2 = IntAuth_2_I | IntAuth_2_R
IntAuth_3 = IntAuth_3_I | IntAuth_3_R
...

IntAuth_1_I = prf(SK_pi_1, [IntAuth_1_I_P |] IntAuth_1_I_A)
IntAuth_2_I = prf(SK_pi_2, [IntAuth_2_I_P |] IntAuth_2_I_A)
IntAuth_3_I = prf(SK_pi_3, [IntAuth_3_I_P |] IntAuth_3_I_A)
...

IntAuth_1_R = prf(SK_pr_1, [IntAuth_1_R_P |] IntAuth_1_R_A)
IntAuth_2_R = prf(SK_pr_2, [IntAuth_2_R_P |] IntAuth_2_R_A)
IntAuth_3_R = prf(SK_pr_3, [IntAuth_3_R_P |] IntAuth_3_R_A)
...
```

IntAuth_1_I/IntAuth_1_R, IntAuth_2_I/IntAuth_2_R, IntAuth_3_I/IntAuth_3_R, etc. represent the results of applying the negotiated prf to the content of the INTERMEDIATE messages sent by the initiator (IntAuth_*_I) and by the responder (IntAuth_*_R) in an order of increasing Message IDs (i.e. in an order the INTERMEDIATE exchanges took place). The prf is applied to the two chunks of data: optional IntAuth_*_[I/R]_P and mandatory IntAuth_*_[I/R]_A. The IntAuth_*_[I/R]_A chunk lasts from the first octet of the IKE Header (not including prepended four octets of zeros, if port 4500 is used) to the last octet of the Encrypted Payload header. The IntAuth_*_[I/R]_P chunk is present if the Encrypted payload is not empty. It consists of the not yet encrypted content of the Encrypted payload, excluding Initialization Vector, Padding, Pad Length and Integrity Checksum Data fields (see 3.14 of [RFC7296] for description of the Encrypted payload). In other words, the IntAuth_*_[I/R]_P chunk is the inner payloads of the Encrypted payload in plaintext form.

```
                       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
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ^ ^
  |                       IKE SA Initiator's SPI                  | | |
  |                                                               | | |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ I |
  |                       IKE SA Responder's SPI                  | K |
  |                                                               | E |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+   |
  |  Next Payload | MjVer | MnVer | Exchange Type |     Flags     | H |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ d |
  |                          Message ID                           | r A
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ | |
  |                            Length                             | | |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ v |
  |                                                               |   |
  ~                 Unencrypted payloads (if any)                 ~   |
  |                                                               |   |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ^ |
  | Next Payload  |C|  RESERVED   |         Payload Length        | | |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ E v
  |                     Initialization Vector                     | n
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ c ^
  |                                                               | r |
  ~             Inner payloads (not yet encrypted)                ~   P
  |                                                               | P |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ l v
  |              Padding (0-255 octets)           |  Pad Length   | d
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |
  ~                    Integrity Checksum Data                    ~ |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ v
```

Figure 1: Data to Authenticate in the INTERMEDIATE Exchange Messages

Figure 1 illustrates the layout of the IntAuth_*_[I/R]_P (denoted as P) and the IntAuth_*_[I/R]_A (denoted as A) chunks in case the Encrypted payload is not empty.

The calculations are applied to whole messages only, before possible fragmentation. This ensures that the IntAuth will be the same regardless of whether fragmentation takes place or not ([RFC7383] allows sending first unfragmented message and then trying fragmentation in case of no reply).

Each calculation of IntAuth_*_[I/R] uses its own key SK_p[i/r]_*, which is the most recently updated SK_p[i/r] key available before the corresponded INTERMEDIATE exchange is started. The first INTERMEDIATE exchange always uses SK_p[i/r] key that was computed in

the IKE_SA_INIT as SK_p[i/r]_1. If the first INTERMEDIATE exchange performs additional key exchange resulting in SK_p[i/r] update, then this updated SK_p[i/r] is used as SK_p[i/r]_2, otherwise the original SK_p[i/r] is used, and so on. Note, that if keys are updated then for any given INTERMEDIATE exchange the keys SK_e[i/r] and SK_a[i/r] used for its messages protection (see Section 3.3.1) and the keys SK_p[i/r] for its authentication are always from the same generation.

3.4. Error Handling in the INTERMEDIATE Exchange

Since messages of the INTERMEDIATE exchange are not authenticated until the IKE_AUTH exchange successfully completes, possible errors need to be handled carefully. There is a trade-off between providing a better diagnostics of the problem and a risk to become a part of DoS attack. See Section 2.21.1 and 2.21.2 of [RFC7296] describe how errors are handled in initial IKEv2 exchanges, these considerations are applied to the INTERMEDIATE exchange too.

4. Interaction with other IKEv2 Extensions

The INTERMEDIATE exchanges MAY be used in the IKEv2 Session Resumption [RFC5723] between the IKE_SESSION_RESUME and the IKE_AUTH exchanges.

5. Security Considerations

The data that is transferred by means of the INTERMEDIATE exchanges is not authenticated until the subsequent IKE_AUTH exchange is completed. However, if the data is placed inside the Encrypted payload, then it is protected from passive eavesdroppers. In addition the peers can be certain that they receives messages from the party he/she performed the IKE_SA_INIT with if they can successfully verify the Integrity Checksum Data of the Encrypted payload.

The main application for Intermediate Exchange is to transfer large amount of data before IKE SA is set up without causing IP fragmentation. For that reason it is expected that in most cases IKE Fragmentation will be employed in the INTERMEDIATE exchanges. Section 5 of [RFC7383] contains security considerations for IKE Fragmentation.

Note, that if an attacker was able to break key exchange in real time (e.g. by means of Quantum Computer), then the security of the INTERMEDIATE exchange would degrade. In particular, such an attacker would be able both to read data contained in the Encrypted payload and to forge it. The forgery would become evident in the IKE_AUTH exchange (provided the attacker cannot break employed authentication

mechanism), but the ability to inject forged the INTERMEDIATE exchange messages with valid ICV would allow the attacker to mount Denial-of-Service attack. Moreover, if in this situation the negotiated prf was not secure against preimage attack with known key, then the attacker could forge the INTERMEDIATE exchange messages without later being detected in the IKE_AUTH exchange. To do this the attacker should find the same IntAuth_*_[I|R] value for the forged message as for original.

6. IANA Considerations

This document defines a new Exchange Type in the "IKEv2 Exchange Types" registry:

<TBA>   INTERMEDIATE

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

<TBA>   INTERMEDIATE_EXCHANGE_SUPPORTED

7. Acknowledgements

The idea to use an intermediate exchange between IKE_SA_INIT and IKE_AUTH was first suggested by Tero Kivinen. Scott Fluhrer and Daniel Van Geest identified a possible problem with authentication of the INTERMEDIATE exchange and helped to resolve it.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <https://www.rfc-editor.org/info/rfc7296>.

[RFC7383] Smyslov, V., "Internet Key Exchange Protocol Version 2 (IKEv2) Message Fragmentation", RFC 7383, DOI 10.17487/RFC7383, November 2014, <https://www.rfc-editor.org/info/rfc7383>.

8.2. Informative References

[RFC8229] Pauly, T., Touati, S., and R. Mantha, "TCP Encapsulation of IKE and IPsec Packets", RFC 8229, DOI 10.17487/RFC8229, August 2017, <https://www.rfc-editor.org/info/rfc8229>.

[RFC5723] Sheffer, Y. and H. Tschofenig, "Internet Key Exchange Protocol Version 2 (IKEv2) Session Resumption", RFC 5723, DOI 10.17487/RFC5723, January 2010, <https://www.rfc-editor.org/info/rfc5723>.

Author's Address

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

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