Workgroup: ipsecme
Internet-Draft:
draft-mglt-ipsecme-multiple-child-sa-01
Published: 9 November 2022
Intended Status: Standards Track
Expires: 13 May 2023
Authors: D. Migault S. Klassert
Ericsson Secunet

Negotiation of multiple Child Security Association with the Internet Key Exchange Protocol Version 2 (IKEv2)

Abstract

IPsec packet processing with one Security Association (SA) per core is more efficient than having a SA shared by the multiple cores.

This document optimizes the negotiation of multiple unidirectional SAs so that each peer can assign one unidirectional SA per core.

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 https://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 13 May 2023.

Table of Contents

1. Requirements 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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Introduction

IPsec processing (on Linux) is more efficient with SA attached to a given core as opposed to a SA shared by multiple cores. Suppose an initiator and a responder respectively with n and p cores establish an IPsec protected communication defined by Traffic Selectors (TSi, TSr). IPsec processing performance may be increased if the initiator (resp. the responder) processes IPsec packets via n (resp. p) distinct unidirectional SAs rather than having a SA shared by the n (resp p) cores.

Optimally the number of SAs is expected to be equal to the number of cores which can be different for each peer. When peers have a different number of cores, the number of SA is expected to be equal to the highest number of cores to minimize context switching and the minimum number of cores to optimize memory space. In fact, having fewer SAs than the number of cores may result in switching the SA context to unused cores. On the other hand, having a greater number of SAs results in a core sharing multiple SAs for the same purpose, which does not improve performances at the cost of an additional SA stored in the kernel.

Currently Child SA are agreed with IKEv2 [RFC7296] CREATE_CHILD_SA exchange. Additional Child SAs (in our case n or p) would require n or p CREATE_CHILD_SA exchanges that add multiple round trips carrying similar payloads (TSi, TSr, SA) which is not necessary.

This document describes the MULTIPLE_CHILD_SA Notify Payload used in a CREATE_CHILD_SA to indicate the support of Multiple SA Extension as well as to agree on the additional number negotiated SA. Section Section 4 describes how SAs are generated.

3. Protocol Exchange

Note for the WG: Because the CREATE_CHILD_SA happens in the IKE_AUTH exchange which is usually used to advertise the supported extensions, the current protocol does not advertise or negotiate the support of the extension in a separate exchange.

The support for Multiple Child SA extension as well as the number of additional Child SAs is performed during the CREATE_CHILD_SA exchange via the MULTIPLE_CHILD_SA Notify Payload.

The initiator indicates in a single MULTIPLE_CHILD_SA notification, the requested additional number of SA (nChildSAi), the maximum number of Child SA (maxChildSA) he commits to generate as well as an ordered list of maxChildSA SPI (SPIi)for potentially accepted additional SA by the responder.

It is RECOMMENDED that maxChildSA balances the limitations of the initiator while enabling responders to optimize their IPsec processing as well. Setting nChildSAi to n and maxChildSA to 2 * n seems a reasonable comprise for communications between nodes of similar capacities.

```
initiator                          responder
-------------------------------------------------------------------
HDR, SK {IDi, [CERT,] [CERTREQ,]
    [IDr,] AUTH, SAi2, TSi, TSr,
    N(MULTIPLE_CHILD_SA(nChildSAi, maxChildSA=2n, SPIi))}  -->
```

Upon receiving a request for the CREATE_CHILD_SA exchange, the responder builds the CREATE_CHILD_SA Response. The MULTIPLE_CHILD_SA Notify Payload is processed only when the CREATE_CHILD_SA can be successfully completed and that the responder supports the Multiple Child SA extension. Otherwise the MULTIPLE_CHILD_SA Notify Payload is ignored. Only the first encountered MULTIPLE_CHILD_SA notification is considered, others are ignored.

Upon receiving the MULTIPLE_CHILD_SA Notify Payload, a responder indicates the accepted number of additional SA (nChildSAr) it is willing to generate. nChildSAr MUST be equal or greater to 0 and lower or equal to maxChildSA.

The responder generates an ordered list of nChildSAr SPIs (SPIr), returns to the initiator nChildSAr, maxChildSA set to zero and SPIr. The responder populates the nChildSAr additional Child SAs from

SAr2, TSi, TSr, nChildSAr, SPIi, SPIr and KEYMAT as defined in [RFC7296] section 2.17 for the Child SA and as defined in Section 4 for the other additional Child SAs.

```
<--  HDR, SK {IDr, [CERT,] AUTH,
         SAr2, TSi, TSr,
          N(MULTIPLE_CHILD_SA(nChildSAr, SPIr))}
```

If the CREATE_CHILD_SA is processed correctly by the initiator, the initiator checks nChildSAr is lower or equal to maxChildSA initially provided. The value of maxChildSA carried by the notification is ignored. Additional Child SAs are populated as defined in [RFC7296] section 2.17 for the Child SA and as defined in {keying-mat} for the other additional Child SAs.

4. Generating Keying Material for Child Sas

This section details how each peers derives the cryptographic material for nChildSAr Child SAs from SAi2, SAr2, TSi, TSr, nChildSAr, SPIi, SPIr and KEYMAT.

The initiator and the responder generates the first Child SA as defined by the CREATE_CHILD_SA in [RFC7296] and the cryptographic material is derived as defined in [RFC7296] Section 2.17.

Upon receiving the MULTIPLE_CHILD_SA Extension, each peer generates the remaining SAs by repeating a CREATE_CHILD_SA negotiation nChildSAr times. While this is implementation dependent how the nChildSAr set of SAs are generated, the resulting SAs MUST ended in the same result as described below:

While SPIi and SPIr are not empty: * Take the first SPI of SPIi (SPIi[0]), and remove that value from SPIi. SPIi length is decreased by one. * Replace SPI value in SA2i by SPIi[0] * Take the first SPI of SPIr (SPIr[0]), and remove that value from SPIr. SPIr length is decreased by one. * Replace SPI value in SA2r by SPIr[0] * Generates the SAs as described in [RFC7296] section 2.17.

Note for the WG: The handling of MULTIPLE_CHILD_SA is based on information exchanged during the CREATE_CHILD_SA exchange. It woudl be better to have the MULTIPLE_CHILD_SA Payload BEFORE the CREATE_IKE_SA.

5. Error Handling

There may be conditions when the responder for some reason is unable or unwilling to create additional Child SAs. This inability may be temporary or permanent.

Temporary inability occurs when the responder doesn't have enough resources at the moment to generate Child SAs. In this case, the responder SHOULD reject the request to clone the IKE SA with the TEMPORARY_FAILURE notification.

<-- HDR, SK {N(TEMPORARY_FAILURE)}

After receiving this notification, the initiator MAY retry its request after waiting some period of time. See Section 2.25 of [RFC7296] for details.

In some cases, the responder may have restrictions on the number of coexisting SAs with one peer. These restrictions may be either implicit (some devices may have enough resources to handle only a few SAs) or explicit (provided by some configuration parameter). If the initiator wants more SAs than the responder is able or is configured to handle, the responder SHOULD reject the request with the NO_ADDITIONAL_SAS notification as defined in [RFC7296].

<-- HDR, SK {N(NO_ADDITIONAL_SAS)}

This condition is considered permanent and the initiator SHOULD NOT retry creating Child SAs until some of the existing SAs with the responder are deleted. This condition is considered permanent and the initiator SHOULD NOT retry cloning an IKE SA until some of the existing SAs with the responder are deleted.

6. Payload Description

Figure 1 illustrates the Notify Payload packet format as described in Section 3.10 of [RFC7296] used for both the MULTIPLE_CHILD_SA notifications.

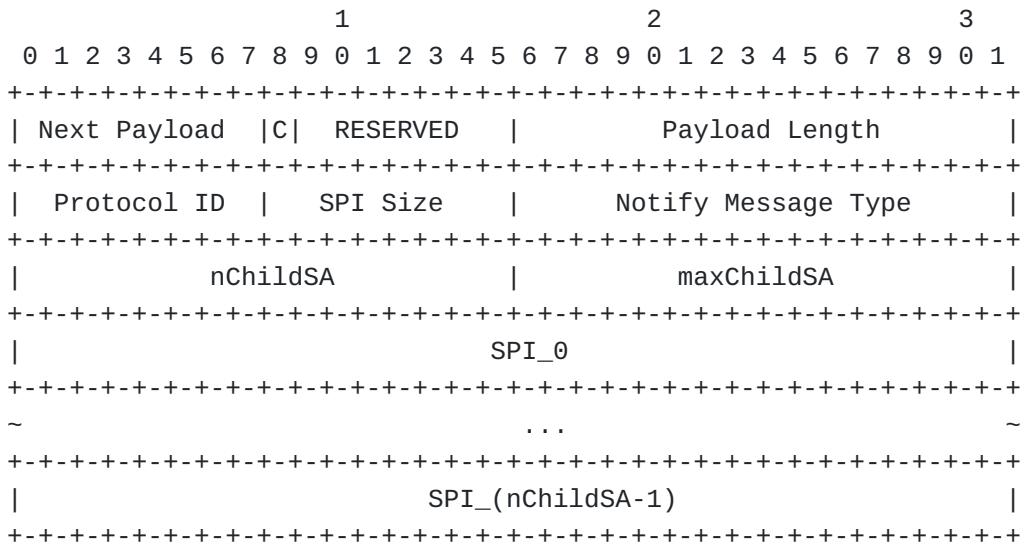

Figure 1: Notify Payload

The fields Next Payload, Critical Bit, RESERVED, and Payload Length are defined in [RFC7296]. Specific fields defined in this document are:

*Protocol ID (1 octet): Set to zero.

*Security Parameter Index (SPI) Size (1 octet): Set to zero.

*Notify Message Type (2 octets): Specifies the type of notification message. It is set to TBD1 for the MULTIPLE_CHILD_SA notification.

*nChildSA (2 octets): number of set of SAs. The value set by the initiator is nChildSAi and the one set by the responder is nChildSAr.

*maxChildSA (2 octets): Maximum number of acceptable set of SAs. This value is set by the initiator and set to zero by the responder.

*SPI_0... SPI_(nChildSA-1): the list of nChildSA SPIs. The list is designated as SPIi when sent by th einitiator and as SPIr when sent by the responder.

7. IANA Considerations

IANA has allocated two values in the "IKEv2 Notify Message Types - Status Types" registry:

Value	Notify Messages - Status Types
TBD1	MULTIPLE_CHILD_SA

8. Security Consideration

The protocol defined in this document does not modify IKEv2. Security considerations. Generating multiple SA are mostly equivalent as the CREATE_CHILD_SA exchange described in [RFC7296].

9. 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>.

[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>.

[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>.

Authors' Addresses

Daniel Migault
Ericsson
8275 Trans Canada Route
Saint Laurent, QC 4S 0B6
Canada

Email: daniel.migault@ericsson.com

Steffen Klassert
Secunet

Email: steffen.klassert@secunet.com