IPSECME
Internet-Draft
Intended status: Standards Track
Expires: January 06, 2014

D. Migault (Ed)
Francetelecom - Orange

July 05, 2013

KEEP_OLD_IKE_SA Extension
draft-mglt-ipsecme-keep-old-ike-sa-00.txt

Abstract

This document considers a VPN Client setting a VPN with a security gateway where at least one of the peer has multiple interfaces.

With the current IKEv2, the outer IP addresses of the VPN are determined by those used by IKEv2 channel. As a result using multiple interface requires to set an IKEv2 channel on each interface, and then on each paths if both the VPN Client and the security gateway have multiple interfaces. Setting multiple IKEv2 channel involves multiple authentications which MAY each require multiple round trips and delay the VPN establishment. In addition multiple authentications unnecessarily load the VPN client and the authentication infrastructure.

This document presents the KEEP_OLD_IKE_SA extension, where an additional IKEv2 channel from an already authenticated IKEv2 channel. The newly created IKEv2 channel is set without the IKEv2 authentication exchange. The newly created IKEv2 channel can then be assigned to another interface using MOBIKE.

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 January 06, 2014.

Table of Contents

1. Requirements notation

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. Introduction

This document considers a VPN End User setting its VPN with a Security Gateway, and at least one of the peers has multiple interfaces. Figure 1 represents the case where the VPN has multiple interfaces, figure 2 represents the case where the Security Gateway has multiple interfaces, and figure 3 represents the case where both the VPN End User and the Security Gateway has multiple interfaces. With figure 1 and figure 2, one of the peer has n = 2 interfaces and the other has a single interface. This results in the creating of up to n = 2 VPNs. With figure 3, the VPN End User has n = 2 interfaces and the Security Gateway has m =2 interfaces. This can lead to up to m x n VPNs.

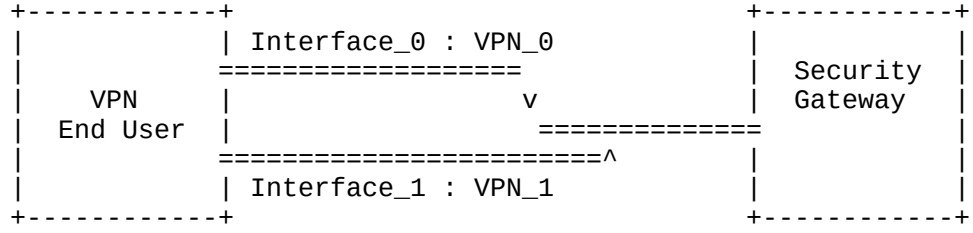

Figure 1: VPN End User with Multiple Interfaces

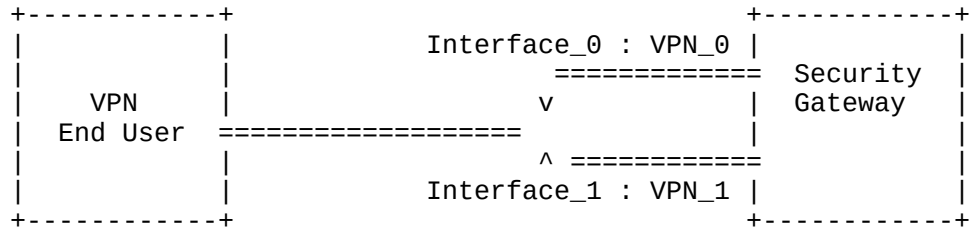

Figure 2: Security Gateway with Multiple Interfaces

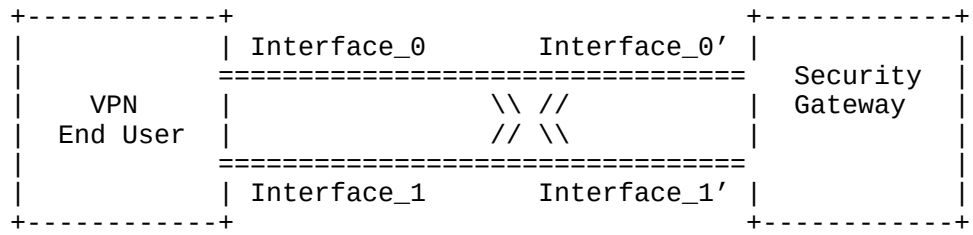

Figure3: VPN End User and Security Gateway

With the current IKEv2 [RFC5996], each VPN requires an IKEv2 channel, and setting an IKEv2 channel requires an authentication. Authentication can involve multiple round trips like EAP-SIM [RFC4186] as well as crypto operations that MAY delay the connectivity.

This document presents the KEEP_OLD_IKE_SA extension. The main idea is that the peer with multiple interfaces sets an first authenticated IKEv2 channel. Then it takes advantage of this authentication and derives as many parallel IKEv2 channels as VPNs. On each IKEv2 channel a VPN is negotiated. This results in parallel VPNS. Then the VPN End User moves the VPNs to their proper places using MOBIKE. Alternatively, the VPN End User can also move the IKEv2 channels and then negotiate the VPNs.

[I-D.mglt-mif-security-requirements] provides a problem statement for IPsec and multiple interfaces.
[I-D.arora-ipsecme-ikev2-alt-tunnel-addresses] and
[I-D.mglt-ipsecme-alternate-outer-address] have been proposed so tunnel outer IP address can differ from those of the IKEv2 channel. The advantage of the KEEP_OLD_IKE_SA extension is that is requires very few modification to already existing IKEv2 implementation. Then, it is reusing already existing and widely deployed protocol such as MOBIKE [RFC4555]. Finally by keeping a dedicated IKEv2 channel for each VPN, it eases reachability tests.

3. Terminology

This section defines terms and acronyms used in this document.

- VPN End User: designates the End User that initiates the VPN with a Security Gateway. This End User may be mobile and moves its VPN from on Security Gateway to the other.

- Security Gateway: designates a point of attachment for the VPN service. In this document, the VPN service is provided by multiple Security Gateways. Each Security Gateway may be considered as a specific hardware.

- Security Association (SA): The Security Association is defined in [RFC4301].

4. Protocol Overview

The goal of the document is to specify how to create a new IKEv2 channel. IKEv2 [RFC5996] specifies the CREATE_CHILD_SA that makes possible to rekey an IKE_SA, create or rekey a new Child SA.

The difference between rekeying an IKE_SA and creating a new IKE_SA is that the old IKE_SA MUST NOT be deleted, either by starting a Delete exchange or removing the IKE_SA without the Delete exchange.

Note that IKEv2 [RFC5996] Section 1.3.2 or Section 2.18 does not explicitly mentions that the old IKE_SA MUST be deleted. However, there are currently no signaling advertising the IKE_SA has not been deleted. The purpose of this document is to avoid this uncertainty when rekeying the IKE_SA. In other words, the document avoids that one peer expects a additional IKE_SA to be created whereas the other simply proceed to a replacement of the old IKE_SA.

Currently, one MAY check whether or not the old IKE_SA has been deleted or not by waiting a for a given time and then initiate and empty INFORMATIONAL exchange using the old IKE_SA. The absence of response MAY indicate the old IKE_SA has been removed.

This document introduces KEEP_OLD_IKE_SA Notify Payload. The initiator sends the KEEP_OLD_IKE_SA Notify Payload in a CREATE_CHILD_SA request for rekeying the IKE_SA. The KEEP_OLD_IKE_SA Notify Payload is placed before the concerned SA and indicates what is expected for the old IKE_SA. Motivation of this draft is to indicate the old IKE_SA MUST NOT be deleted once the new IKE_SA is rekeyed. Alternatively, the initiator MAY use the KEEP_OLD_IKE_SA Notify Payload to indicate the old IKE_SA is not expected to be re-used.

```
Initiator                          Responder
-----------------------------------------------------------------
HDR, SK {N(KEEP_OLD_IKE_SA) SA, Ni, KEi} -->
```

The responder finds a KEEP_OLD_IKE_SA, if it does not understand the extension it ignores the payload as defined in [RFC5996] Section 3.10.1. Similarly, the KEEP_OLD_IKE_SA Notify Payload MUST be ignored if the CREATE_CHILD_SA request does not concern a IKE_SA rekey. If the initiator wants to check whether the IKE_SA has been deleted or not, it SHOULD proceed to additional empty INFORMATIONAL exchange as described in [RFC5996] Section 2.4. In this case, the responder’s response will be:

```
                              <--  HDR, SK {SA, Nr, KEr}
```

In any other case, the responder understands the KEEP_OLD_IKE_SA Notify Payload and the CREATE_CHILD_SA request concerns a IKE_SA rekey. The responder MUST proceed to the IKE_SA rekey. If the KEEP_OLD_IKE_SA indicates the old IKE_SA MUST be kept, the responder MUST keep the old IKE_SA active. Alternatively, if it indicates the old IKE_SA is not supposed to be used, the responder MAY delete the old IKE_SA after a given time out. The responder MUST respond and indicate if the old IKE_SA has been kept or not. The exchange will be:

```
                        <--  HDR, SK { N(KEEP_OLD_IKE_SA)
                                       SA, Nr, KEr}
```

5. Payload Description

Figure 7 illustrates the KEEP_OLD_IKE_SA Notify Payload packet format.

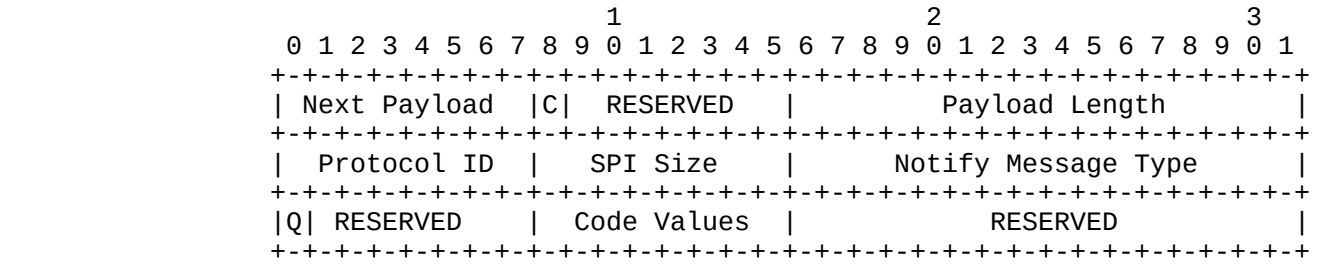

Figure 7: KEEP_OLD_IKE_SA Notify Payload

- Next Payload (1 octet): Indicates the type of payload that follows after the header.

- Critical Bit (1 bit): Indicates how the responder handles the Notify Payload. In this document the Critical Bit is not set.

- RESERVED (7 bits): MUST be set as zero; MUST be ignored on receipt.

- Payload Length (2 octet): Length in octets of the current payload, including the generic payload header.

- Protocol ID (1 octet): set to zero.

- SPI Size (1 octet): set to zero.

- Notify Message Type (2 octets): Specifies the type of notification message. It is set to KEEP_OLD_IKE_SA in this document.

- Question Bit (1 bit): set to one by the initiator and set to zero by the responder.

- RESERVED (7 bits): set to zero.

- Code Values: Code that indicates what action is expected to be done with the newly negotiated IKE_SA.

Code Values	
Keep Old IKE_SA	0
Unused Old IKE_SA	1
Unassigned	2-255

6. IANA Considerations

The new fields and number are the following:

IKEv2 Notify Message Types - Status Types	
KEEP_OLD_IKE_SA	- TBD

7. Security Considerations

The protocol defined in this document does not modifies IKEv2. It signalizes what has been implementation dependent on how to manage an old IKE_SA after a rekey.

8. Acknowledgment

The ideas of this draft came from various inputs from the ipsecme and discussions with Tero Kivinen and Michael Richardson.

9. References

9.1. Normative References

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

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4555] Eronen, P., "IKEv2 Mobility and Multihoming Protocol (MOBIKE)", RFC 4555, June 2006.

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

9.2. Informational References

[I-D.arora-ipsecme-ikev2-alt-tunnel-addresses] Arora, J. and P. Kumar, "Alternate Tunnel Addresses for IKEv2", draft-arora-ipsecme-ikev2-alt-tunnel-addresses-00 (work in progress), April 2010.

[I-D.mglt-ipsecme-alternate-outer-address] Migault, D., "IKEv2 Alternate Outer IP Address Extension", draft-mglt-ipsecme-alternate-outer-address-00 (work in progress), February 2013.

[I-D.mglt-mif-security-requirements] Migault, D. and C. Williams, "IPsec Multiple Interfaces Problem Statement", draft-mglt-mif-security-requirements-03 (work in progress), November 2012.

[RFC4186] Haverinen, H. and J. Salowey, "Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM)", RFC 4186, January 2006.

Appendix A. Document Change Log

[RFC Editor: This section is to be removed before publication]

-00: First version published.

Appendix B. Setting a VPN on Multiple Interfaces

This section is informational and exposes how a VPN End User as illustrated in Figure 1 can builds two VPNs on its two interfaces without multiple authentications. Other cases represented in figure 2 and 3 are similar and can be easily derived from the case. The mechanism is based on the KEEP_OLD_IKE_SA extension and the MOBIKE extension [RFC4555].

B.1. Setting VPN_0

First, the VPN End User negotiates a VPN using one interface. This involves a regular IKEv2 setting. In addition, the VPN End User and

the Security Gateway advertise they support MOBIKE. At the end of the exchange, VPN_0 is set as represented in figure 4.

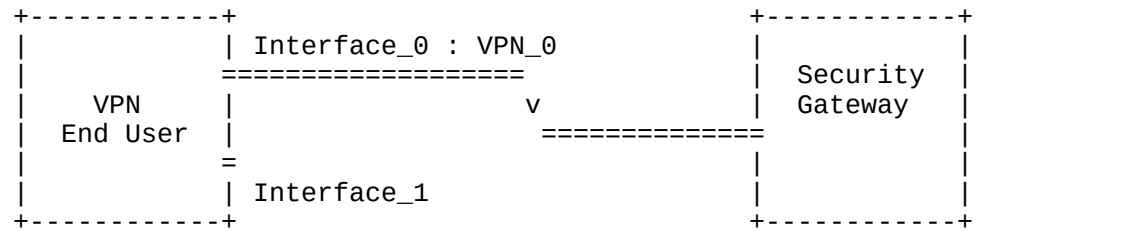

Figure 4: VPN End User Establishing VPN_0

The exchange is completely described in [RFC4555]. First the negotiates the IKE_SA. In the figure below peers also proceed to NAT detection because of the use of MOBIKE.

```
Initiator                          Responder
-----------------------------------------------------------------------
(IP_I1:500 -> IP_R1:500)
HDR, SAi1, KEi, Ni,
     N(NAT_DETECTION_SOURCE_IP),
     N(NAT_DETECTION_DESTINATION_IP)  -->

                       <--  (IP_R1:500 -> IP_I1:500)
                            HDR, SAr1, KEr, Nr,
                                 N(NAT_DETECTION_SOURCE_IP),
                                 N(NAT_DETECTION_DESTINATION_IP)
```

The initiators and the responder proceed to the authentication exchange, advertise they support MOBIKE and negotiate the SA for VPN_0. Optionally, the initiator and the Security Gateway MAY advertise their multiple interfaces using the ADDITIONAL_IP4_ADDRESS and/or ADDITIONAL_IP6_ADDRESS Notify Payload

```
(IP_I1:4500 -> IP_R1:4500)
HDR, SK { IDi, CERT, AUTH,
          CP(CFG_REQUEST),
          SAi2, TSi, TSr,
          N(MOBIKE_SUPPORTED),
          N(ADDITIONAL_IP*_ADDRESS)+ }  -->

                       <--  (IP_R1:4500 -> IP_I1:4500)
                            HDR, SK { IDr, CERT, AUTH,
                                      CP(CFG_REPLY),
```

```
                                    SAr2, TSi, TSr,
                                    N(MOBIKE_SUPPORTED),
                                    N(ADDITIONAL_IP*_ADDRESS)+}
```

B.2. Creating an additional IKEv2 Channel

In our case the the initiator wants to set establish a VPN with its Interface_1 between the VPN End User and the Security Gateway. The VPN End User will first establish a parallel IKE_SA using a CREATE_CHILD_SA that concerns an IKE_SA rekey associated to a KEEP_OLD_IKE_SA Notify Payload. This results in two different IKE_SA between the VPN End User and the Security Gateway. Currently both IKE_SA are set using Interface 0 of the VPN End User.

In this section we consider the creation of the additional IKE_SA as a separate exchange. However, they are several situations where this extra round trips MAY be avoided. First if the VPN End User knows multiple interfaces MAY be involved, it can combine this exchange with the previous one (IKE_AUTH, CREATE_CHILD_SA concerning the creation of the SA). Secondly, the Security Gateway MAY also start the CREATE_CHILD_SA exchange to create an additional IKE_SA. This reduces the delay to half a round trip.

The CREATE_CHILD_SA exchange to create an additional IKE_SA MAY be combined with the IKE_AUTH exchange exchange if the VPN End User estimates with a high probability that multiple interfaces MAY be involved in the communication. This MAY be the case if the VPN End User has multiple interfaces, or if the VPN End User guesses that the Security Gateway has multiple interfaces. In the case the KEEP_OLD_IKE_SA Notify Payload is not supported by the Security Gateway or that the Security Gateway has only one interface, this will result in rekeying the IKE_SA, and thus does not compromise the communication.

Similarly, the CREATE_CHILD_SA exchange to create an additional IKE_SA MAY be initiated by the responder and combined with the IKE_AUTH exchange if the Security Gateway wants to reduce the number of round trips, and supposes the VPN End User will use its multiple interfaces. Note that the Security Gateway knows if multiple interfaces are involved in the communication. What remains uncertain is whether the VPN End User has the ability to use these multiple interfaces simultaneously.

```
Initiator                          Responder
-------------------------------------------------------------------
(IP_I1:4500 -> IP_R1:4500)
HDR, SK { N(KEEP_OLD_IKE_SA),
```

```
            SA, Ni, KEi} -->
                        <--  (IP_R1:4500 -> IP_I1:4500)
                             HDR, SK { N(KEEP_OLD_IKE_SA),
                                       SA, Nr, KEr}
```

B.3. Creation of the Child SA for VPN_1

Once the new IKEv2 channel has been created, the VPN End User MAY initiate a CREATE_CHILD_SA exchange that concerns the creation of a Child SA for VPN_1. The newly created VPN_1 will use Interface_0 of the VPN End User.

It is out of scope of the document to define how the VPN End User MAY handle traffic with its multiple interfaces. The VPN End User MAY use the same IP inner address on its multiple interfaces. In this case, the same Traffic Selectors (that is the IP address used for VPN_0 and VPN_1) MAY match for both VPNs VPN_0 and VPN_1. The End User VPN SHOULD be aware of such match and be able to manage it. It MAY for example use distinct Traffic Selectors on both VPNs using different ports, manage the order of its SPD or have SPD defined per interfaces. Defining these mechanisms are out of scope of this document. Alternatively, the VPN End User MAY uses a different IP address for each interface. In the latter case, if the inner IP address is assigned by the Security Gateway, the Configuration Payload (CP) MUST be placed before the SA Payload as specified in [RFC5996] Section 2.19.

The creation of VPN_1 is performed via the newly created IKE_SA as follows:

```
Initiator                          Responder
-------------------------------------------------------------------
(IP_I1:4500 -> IP_R1:4500)
HDR(new), SK(new) { [CP(CFG_REQUEST)],
          SAi2, TSi, TSr }  -->

                        <--  (IP_R1:4500 -> IP_I1:4500)
                             HDR(new), SK(new) { [CP(CFG_REPLY)],
                                       SAr2, TSi, TSr}
```

The resulting configuration is depicted in figure 5. VPN_0 and VPN_1 have been created, but both are using the same Interface: Interface_0.

```
+------------+                                  +------------+
|            | Interface_0 : VPN_0, VPN_1       |            |
```

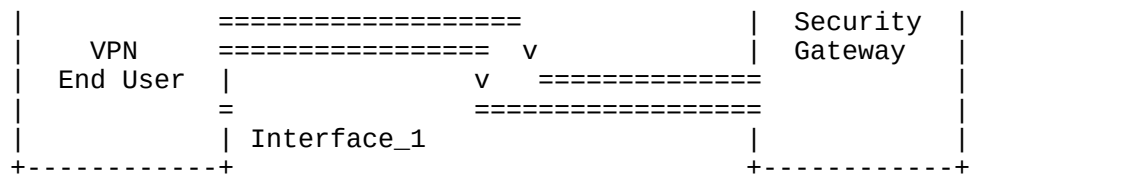

Figure 5: VPN End User Establishing VPN_0 and VPN_1

B.4. Moving VPN_1 on Interface_1

In this section, MOBIKE is used to move VPN_1 on interface_1. The exchange is described in [RFC4555]. All exchanges are using the new IKE_SA. Eventually, the VPN End User MAY check if the Security Gateway is reachable via Interface_1. The exchanges are described below:

```
Initiator                          Responder
-----------------------------------------------------------------------
(IP_I2:4500 -> IP_R1:4500)
HDR(new), SK(new) { N(NAT_DETECTION_SOURCE_IP),
          N(NAT_DETECTION_DESTINATION_IP) }

                        <--  (IP_R2:4500 -> IP_I1:4500)
                             HDR(new), SK(new) {
                                  N(NAT_DETECTION_SOURCE_IP),
                                  N(NAT_DETECTION_DESTINATION_IP) }
```

(This worked, and the initiator requests the peer to switch to new addresses.)

```
(IP_I2:4500 -> IP_R1:4500)
HDR(new), SK(new) { N(UPDATE_SA_ADDRESSES),
          N(NAT_DETECTION_SOURCE_IP),
          N(NAT_DETECTION_DESTINATION_IP),
          N(COOKIE2) }  -->

                        <--  (IP_R1:4500 -> IP_I2:4500)
                             HDR(new), SK(new) {
                                  N(NAT_DETECTION_SOURCE_IP),
                                  N(NAT_DETECTION_DESTINATION_IP),
                                  N(COOKIE2) }
```

This results in the situation as described in figure 6.

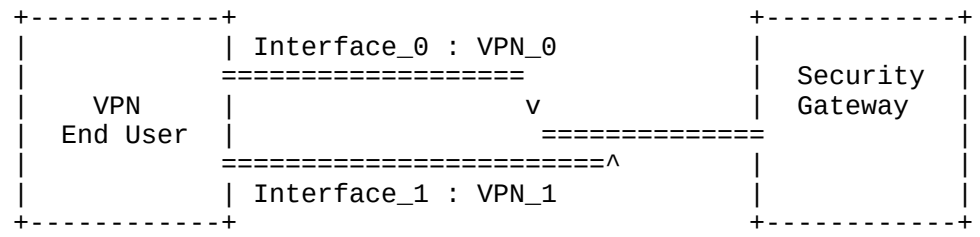

Figure 6: VPN End User with Multiple Interfaces

B.5. Reduced Exchange

The previous sections detail the various exchanges between the VPN End User and the Security Gateway. This section shows an example where the number of exchanges are limited, thus limiting the delay to set up a multiple interface VPN communication.

```
Initiator                          Responder
-----------------------------------------------------------------------

(IP_I1:500 -> IP_R1:500)
HDR, SAi1, KEi, Ni,
     N(NAT_DETECTION_SOURCE_IP),
     N(NAT_DETECTION_DESTINATION_IP)  -->

                      <--  (IP_R1:500 -> IP_I1:500)
                           HDR, SAr1, KEr, Nr,
                                N(NAT_DETECTION_SOURCE_IP),
                                N(NAT_DETECTION_DESTINATION_IP)
(IP_I1:4500 -> IP_R1:4500)
HDR, SK { IDi, CERT, AUTH,
          CP(CFG_REQUEST),
          SAi2, TSi, TSr,
          N(MOBIKE_SUPPORTED),
          N(ADDITIONAL_IP*_ADDRESS)+,
          N(KEEP_OLD_IKE_SA),
          SA, Ni, KEi}                  -->

                      <--  (IP_R1:4500 -> IP_I1:4500)
                           HDR, SK { IDr, CERT, AUTH,
                                     CP(CFG_REPLY),
                                     SAr2, TSi, TSr,
                                     N(MOBIKE_SUPPORTED),
                                     N(ADDITIONAL_IP*_ADDRESS)+},
                                     N(KEEP_OLD_IKE_SA),
                                     SA, Nr, KEr}
```

```
                          <--  (IP_R1:4500 -> IP_I2:4500)
                               HDR(new), SK(new)
                                       { [CP(REQUEST)],
                                         SAi2, TSi, TSr,
                                         N(UPDATE_SA_ADDRESSES)}
   (IP_I2:4500 -> IP_R1:4500)             -->
   HDR(new), SK(new) { [CP(CFG_REPLY)],
             SAr2, TSi, TSr}
```

Author’s Address

Daniel Migault
Francetelecom - Orange
38 rue du General Leclerc
92794 Issy-les-Moulineaux Cedex 9
France

Phone: +33 1 45 29 60 52
Email: mglt.ietf@gmail.com