Network Working Group
Request for Comments: 4555
Category: Standards Track

P. Eronen, Ed.
Nokia

June 2006

IKEv2 Mobility and Multihoming Protocol (MOBIKE)

Status of This Memo

Copyright Notice

Abstract

This document describes the MOBIKE protocol, a mobility and multihoming extension to Internet Key Exchange (IKEv2). MOBIKE allows the IP addresses associated with IKEv2 and tunnel mode IPsec Security Associations to change. A mobile Virtual Private Network (VPN) client could use MOBIKE to keep the connection with the VPN gateway active while moving from one address to another. Similarly, a multihomed host could use MOBIKE to move the traffic to a different interface if, for instance, the one currently being used stops working.

Table of Contents

1. Introduction

1.1. Motivation

IKEv2 is used for performing mutual authentication, as well as establishing and maintaining IPsec Security Associations (SAs). In the base IKEv2 protocol [IKEv2], the IKE SAs and tunnel mode IPsec SAs are created implicitly between the IP addresses that are used when the IKE_SA is established. These IP addresses are then used as the outer (tunnel header) addresses for tunnel mode IPsec packets (transport mode IPsec SAs are beyond the scope of this document). Currently, it is not possible to change these addresses after the IKE_SA has been created.

There are scenarios where these IP addresses might change. One example is mobility: a host changes its point of network attachment and receives a new IP address. Another example is a multihoming host that would like to change to a different interface if, for instance, the currently used interface stops working for some reason.

Although the problem can be solved by creating new IKE and IPsec SAs when the addresses need to be changed, this may not be optimal for several reasons. In some cases, creating a new IKE_SA may require user interaction for authentication, such as entering a code from a token card. Creating new SAs often involves expensive calculations and possibly a large number of round-trips. For these reasons, a mechanism for updating the IP addresses of existing IKE and IPsec SAs is needed. The MOBIKE protocol described in this document provides such a mechanism.

The main scenario for MOBIKE is enabling a remote access VPN user to move from one address to another without re-establishing all security associations with the VPN gateway. For instance, a user could start from fixed Ethernet in the office and then disconnect the laptop and move to the office’s wireless LAN. When the user leaves the office, the laptop could start using General Packet Radio Service (GPRS); when the user arrives home, the laptop could switch to the home wireless LAN. MOBIKE updates only the outer (tunnel header) addresses of IPsec SAs, and the addresses and other traffic selectors used inside the tunnel stay unchanged. Thus, mobility can be (mostly) invisible to applications and their connections using the VPN.

MOBIKE also supports more complex scenarios where the VPN gateway also has several network interfaces: these interfaces could be connected to different networks or ISPs, they may be a mix of IPv4 and IPv6 addresses, and the addresses may change over time. Furthermore, both parties could be VPN gateways relaying traffic for other parties.

1.2. Scope and Limitations

This document focuses on the main scenario outlined above and supports only tunnel mode IPsec SAs.

The mobility support in MOBIKE allows both parties to move, but does not provide a "rendezvous" mechanism that would allow simultaneous movement of both parties or discovery of the addresses when the IKE_SA is first established. Therefore, MOBIKE is best suited for situations where the address of at least one endpoint is relatively stable and can be discovered using existing mechanisms such as DNS (see Section 3.1).

MOBIKE allows both parties to be multihomed; however, only one pair of addresses is used for an SA at a time. In particular, load balancing is beyond the scope of this specification.

MOBIKE follows the IKEv2 practice where a response message is sent to the same address and port from which the request was received. This implies that MOBIKE does not work over address pairs that provide only unidirectional connectivity.

Network Address Translators (NATs) introduce additional limitations beyond those listed above. For details, refer to Section 2.3.

The base version of the MOBIKE protocol does not cover all potential future use scenarios, such as transport mode, application to securing SCTP, or optimizations desirable in specific circumstances. Future extensions may be defined later to support additional requirements. Please consult the MOBIKE design document [Design] for further information and rationale behind these limitations.

1.3. Terminology and Notation

When messages containing IKEv2 payloads are described, optional payloads are shown in brackets (for instance, "[FOO]"), and a plus sign indicates that a payload can be repeated one or more times (for instance, "FOO+"). To provide context, some diagrams also show what existing IKEv2 payloads would typically be included in the exchanges. These payloads are shown for illustrative purposes only; see [IKEv2] for an authoritative description.

When this document describes updating the source/destination addresses of an IPsec SA, it means updating IPsec-related state so that outgoing Encapsulating Security Payload (ESP)/Authentication Header (AH) packets use those addresses in the tunnel header. Depending on how the nominal divisions between Security Association Database (SAD), Security Policy Database (SPD), and Peer Authorization Database (PAD) described in [IPsecArch] are actually implemented, an implementation can have several different places that have to be updated.

In this document, the term "initiator" means the party who originally initiated the first IKE_SA (in a series of possibly several rekeyed IKE_SAs); "responder" is the other peer. During the lifetime of the IKE_SA, both parties may initiate INFORMATIONAL or CREATE_CHILD_SA exchanges; in this case, the terms "exchange initiator" and "exchange responder" are used. The term "original initiator" (which in [IKEv2] refers to the party who started the latest IKE_SA rekeying) is not 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 [KEYWORDS].

2. Protocol Overview

2.1. Basic Operation

MOBIKE allows both parties to have several addresses, and there are up to N*M pairs of IP addresses that could potentially be used. The decision of which of these pairs to use has to take into account several factors. First, the parties may have preferences about which interface should be used due to, for instance, performance and cost reasons. Second, the decision is constrained by the fact that some of the pairs may not work at all due to incompatible IP versions, outages in the network, problems at the local link at either end, and so on.

MOBIKE solves this problem by taking a simple approach: the party that initiated the IKE_SA (the "client" in a remote access VPN scenario) is responsible for deciding which address pair is used for the IPsec SAs and for collecting the information it needs to make this decision (such as determining which address pairs work or do not work). The other party (the "gateway" in a remote access VPN scenario) simply tells the initiator what addresses it has, but does not update the IPsec SAs until it receives a message from the initiator to do so. This approach applies to the addresses in the IPsec SAs; in the IKE_SA case, the exchange initiator can decide which addresses are used.

Making the decision at the initiator is consistent with how normal IKEv2 works: the initiator decides which addresses it uses when contacting the responder. It also makes sense, especially when the initiator is a mobile node: it is in a better position to decide which of its network interfaces should be used for both upstream and downstream traffic.

The details of exactly how the initiator makes the decision, what information is used in making it, how the information is collected, how preferences affect the decision, and when a decision needs to be changed are largely beyond the scope of MOBIKE. This does not mean that these details are unimportant: on the contrary, they are likely to be crucial in any real system. However, MOBIKE is concerned with these details only to the extent that they are visible in IKEv2/IPsec messages exchanged between the peers (and thus need to be standardized to ensure interoperability).

Many of these issues are not specific to MOBIKE, but are common with the use of existing hosts in dynamic environments or with mobility protocols such as Mobile IP [MIP4] [MIP6]. A number of mechanisms already exist or are being developed to deal with these issues. For instance, link-layer and IP-layer mechanisms can be used to track the status of connectivity within the local link [RFC2461]; movement detection is being specified for both IPv4 and IPv6 in [DNA4], [DNA6], and so on.

Naturally, updating the addresses of IPsec SAs has to take into account several security considerations. MOBIKE includes two features designed to address these considerations. First, a "return routability" check can be used to verify the addresses provided by the peer. This makes it more difficult to flood third parties with large amounts of traffic. Second, a "NAT prohibition" feature ensures that IP addresses have not been modified by NATs, IPv4/IPv6 translation agents, or other similar devices. This feature is enabled only when NAT Traversal is not used.

2.2. Example Protocol Exchanges

A simple MOBIKE exchange in a mobile scenario is illustrated below. The notation is based on [IKEv2], Section 1.2. In addition, the source/destination IP addresses and ports are shown for each packet: here IP_I1, IP_I2, IP_R1, and IP_R2 represent IP addresses used by the initiator and the responder.

```
   Initiator                  Responder
  -----------                -----------
1) (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)

2) (IP_I1:4500 -> IP_R1:4500)
   HDR, SK { IDi, CERT, AUTH,
             CP(CFG_REQUEST),
             SAi2, TSi, TSr,
             N(MOBIKE_SUPPORTED) }  -->

                          <--  (IP_R1:4500 -> IP_I1:4500)
                               HDR, SK { IDr, CERT, AUTH,
                                         CP(CFG_REPLY),
                                         SAr2, TSi, TSr,
                                         N(MOBIKE_SUPPORTED) }

(Initiator gets information from lower layers that its attachment
point and address have changed.)

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

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

(Responder verifies that the initiator has given it a correct IP
address.)

4)                        <-- (IP_R1:4500 -> IP_I2:4500)
                              HDR, SK { N(COOKIE2) }

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

Step 1 is the normal IKE_INIT exchange. In step 2, the peers inform each other that they support MOBIKE. In step 3, the initiator notices a change in its own address, and informs the responder about

this by sending an INFORMATIONAL request containing the UPDATE_SA_ADDRESSES notification. The request is sent using the new IP address. At this point, it also starts to use the new address as a source address in its own outgoing ESP traffic. Upon receiving the UPDATE_SA_ADDRESSES notification, the responder records the new address and, if it is required by policy, performs a return routability check of the address. When this check (step 4) completes, the responder starts to use the new address as the destination for its outgoing ESP traffic.

Another protocol run in a multihoming scenario is illustrated below. In this scenario, the initiator has one address but the responder has two.

```
   Initiator                  Responder
  -----------                -----------
1) (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)

2) (IP_I1:4500 -> IP_R1:4500)
   HDR, SK { IDi, CERT, AUTH,
             CP(CFG_REQUEST),
             SAi2, TSi, TSr,
             N(MOBIKE_SUPPORTED) }  -->

                         <--  (IP_R1:4500 -> IP_I1:4500)
                              HDR, SK { IDr, CERT, AUTH,
                                        CP(CFG_REPLY),
                                        SAr2, TSi, TSr,
                                        N(MOBIKE_SUPPORTED),
                                        N(ADDITIONAL_IP4_ADDRESS) }
```

(The initiator suspects a problem in the currently used address pair and probes its liveness.)

```
   3) (IP_I1:4500 -> IP_R1:4500)
      HDR, SK { N(NAT_DETECTION_SOURCE_IP),
                N(NAT_DETECTION_DESTINATION_IP) }  -->

      (IP_I1:4500 -> IP_R1:4500)
      HDR, SK { N(NAT_DETECTION_SOURCE_IP),
                N(NAT_DETECTION_DESTINATION_IP) }  -->

      ...
```

(Eventually, the initiator gives up on the current address pair and tests the other available address pair.)

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

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

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

```
   5) (IP_I1:4500 -> IP_R2:4500)
      HDR, SK { N(UPDATE_SA_ADDRESSES),
                N(NAT_DETECTION_SOURCE_IP),
                N(NAT_DETECTION_DESTINATION_IP),
                N(COOKIE2) }  -->

                              <--  (IP_R2:4500 -> IP_I1:4500)
                                   HDR, SK { N(NAT_DETECTION_SOURCE_IP),
                                             N(NAT_DETECTION_DESTINATION_IP),
                                             N(COOKIE2) }
```

2.3. MOBIKE and Network Address Translation (NAT)

In some MOBIKE scenarios, the network may contain NATs or stateful packet filters (for brevity, the rest of this document simply describes NATs). The NAT Traversal feature specified in [IKEv2] allows IKEv2 to work through NATs in many cases, and MOBIKE can leverage this functionality: when the addresses used for IPsec SAs are changed, MOBIKE can enable or disable IKEv2 NAT Traversal, as needed.

Nevertheless, there are some limitations because NATs usually introduce an asymmetry into the network: only packets coming from the "inside" cause state to be created. This asymmetry leads to

restrictions on what MOBIKE can do. To give a concrete example, consider a situation where both peers have only a single address, and the initiator is behind a NAT. If the responder's address now changes, it needs to send a packet to the initiator using its new address. However, if the NAT is, for instance, of the common "restricted cone" type (see [STUN] for one description of different NAT types), this is not possible. The NAT will drop packets sent from the new address (unless the initiator has previously sent a packet to that address -- which it cannot do until it knows the address).

For simplicity, MOBIKE does not attempt to handle all possible NAT-related scenarios. Instead, MOBIKE assumes that if NATs are present, the initiator is the party "behind" the NAT, and the case where the responder's addresses change is not fully supported (meaning that no special effort is made to support this functionality). Responders may also be unaware of NATs or specific types of NATs they are behind. However, when a change has occurred that will cause a loss of connectivity, MOBIKE responders will still attempt to inform the initiator of the change. Depending on, for instance, the exact type of NAT, it may or may not succeed. However, analyzing the exact circumstances when this will or will not work is not done in this document.

3. Protocol Exchanges

3.1. Initial IKE Exchange

The initiator is responsible for finding a working pair of addresses so that the initial IKE exchange can be carried out. Any information from MOBIKE extensions will only be available later, when the exchange has progressed far enough. Exactly how the addresses used for the initial exchange are discovered is beyond the scope of this specification; typical sources of information include local configuration and DNS.

If either or both of the peers have multiple addresses, some combinations may not work. Thus, the initiator SHOULD try various source and destination address combinations when retransmitting the IKE_SA_INIT request.

3.2. Signaling Support for MOBIKE

Implementations that wish to use MOBIKE for a particular IKE_SA MUST include a MOBIKE_SUPPORTED notification in the IKE_AUTH exchange (in case of multiple IKE_AUTH exchanges, in the message containing the SA payload).

The format of the MOBIKE_SUPPORTED notification is described in Section 4.

3.3. Initial Tunnel Header Addresses

When an IPsec SA is created, the tunnel header IP addresses (and port, if doing UDP encapsulation) are taken from the IKE_SA, not the IP header of the IKEv2 message requesting the IPsec SA. The addresses in the IKE_SA are initialized from the IP header of the first IKE_AUTH request.

The addresses are taken from the IKE_AUTH request because IKEv2 requires changing from port 500 to 4500 if a NAT is discovered. To simplify things, implementations that support both this specification and NAT Traversal MUST change to port 4500 if the correspondent also supports both, even if no NAT was detected between them (this way, there is no need to change the ports later if a NAT is detected on some other path).

3.4. Additional Addresses

Both the initiator and responder MAY include one or more ADDITIONAL_IP4_ADDRESS and/or ADDITIONAL_IP6_ADDRESS notifications in the IKE_AUTH exchange (in case of multiple IKE_AUTH exchanges, in the message containing the SA payload). Here "ADDITIONAL_*_ADDRESS" means either an ADDITIONAL_IP4_ADDRESS or an ADDITIONAL_IP6_ADDRESS notification.

```
   Initiator                  Responder
  -----------                -----------
   HDR, SK { IDi, [CERT], [IDr], AUTH,
             [CP(CFG_REQUEST)]
             SAi2, TSi, TSr,
             N(MOBIKE_SUPPORTED),
             [N(ADDITIONAL_*_ADDRESS)+] }  -->

                        <--  HDR, SK { IDr, [CERT], AUTH,
                                       [CP(CFG_REPLY)],
                                       SAr2, TSi, TSr,
                                       N(MOBIKE_SUPPORTED)
                                       [N(ADDITIONAL_*_ADDRESS)+] }
```

The recipient stores this information, but no other action is taken at this time.

Although both the initiator and responder maintain a set of peer addresses (logically associated with the IKE_SA), it is important to note that they use this information for slightly different purposes.

The initiator uses the set of responder addresses as an input to its address selection policy; it may, at some later point, decide to move the IPsec traffic to one of these addresses using the procedure described in Section 3.5. The responder normally does not use the set of initiator addresses for anything: the addresses are used only when the responder's own addresses change (see Section 3.6).

The set of addresses available to the peers can change during the lifetime of the IKE_SA. The procedure for updating this information is described in Section 3.6.

Note that if some of the initiator's interfaces are behind a NAT (from the responder's point of view), the addresses received by the responder will be incorrect. This means the procedure for changing responder addresses described in Section 3.6 does not fully work when the initiator is behind a NAT. For the same reason, the peers also SHOULD NOT use this information for any other purpose than what is explicitly described either in this document or a future specification updating it.

3.5. Changing Addresses in IPsec SAs

In MOBIKE, the initiator decides what addresses are used in the IPsec SAs. That is, the responder does not normally update any IPsec SAs without receiving an explicit UPDATE_SA_ADDRESSES request from the initiator. (As described below, the responder can, however, update the IKE_SA in some circumstances.)

The reasons why the initiator wishes to change the addresses are largely beyond the scope of MOBIKE. Typically, triggers include information received from lower layers, such as changes in IP addresses or link-down indications. Some of this information can be unreliable: for instance, ICMP messages could be spoofed by an attacker. Unreliable information SHOULD be treated only as a hint that there might be a problem, and the initiator SHOULD trigger Dead Peer Detection (that is, send an INFORMATIONAL request) to determine if the current path is still usable.

Changing addresses can also be triggered by events within IKEv2. At least the following events can cause the initiator to re-evaluate its local address selection policy, possibly leading to changing the addresses.

- An IKEv2 request has been re-transmitted several times, but no valid reply has been received. This suggests the current path is no longer working.

- An INFORMATIONAL request containing an ADDITIONAL_IP4_ADDRESS, ADDITIONAL_IP6_ADDRESS, or NO_ADDITIONAL_ADDRESSES notification is received. This means the peer's addresses may have changed. This is particularly important if the announced set of addresses no longer contains the currently used address.

- An UNACCEPTABLE_ADDRESSES notification is received as a response to address update request (described below).

- The initiator receives a NAT_DETECTION_DESTINATION_IP notification that does not match the previous UPDATE_SA_ADDRESSES response (see Section 3.8 for a more detailed description).

The description in the rest of this section assumes that the initiator has already decided what the new addresses should be. When this decision has been made, the initiator:

- Updates the IKE_SA with the new addresses, and sets the "pending_update" flag in the IKE_SA.

- Updates the IPsec SAs associated with this IKE_SA with the new addresses (unless the initiator's policy requires a return routability check before updating the IPsec SAs, and the check has not been done for this responder address yet).

- If the IPsec SAs were updated in the previous step: If NAT Traversal is not enabled, and the responder supports NAT Traversal (as indicated by NAT detection payloads in the IKE_SA_INIT exchange), and the initiator either suspects or knows that a NAT is likely to be present, enables NAT Traversal (that is, enables UDP encapsulation of outgoing ESP packets and sending of NAT-Keepalive packets).

- If there are outstanding IKEv2 requests (requests for which the initiator has not yet received a reply), continues retransmitting them using the addresses in the IKE_SA (the new addresses).

- When the window size allows, sends an INFORMATIONAL request containing the UPDATE_SA_ADDRESSES notification (which does not contain any data), and clears the "pending_update" flag. The request will be as follows:

```
   Initiator                  Responder
  -----------                -----------
   HDR, SK { N(UPDATE_SA_ADDRESSES),
             [N(NAT_DETECTION_SOURCE_IP),
              N(NAT_DETECTION_DESTINATION_IP)],
             [N(NO_NATS_ALLOWED)],
             [N(COOKIE2)] } -->
```

- If a new address change occurs while waiting for the response, starts again from the first step (and ignores responses to this UPDATE_SA_ADDRESSES request).

When processing an INFORMATIONAL request containing the UPDATE_SA_ADDRESSES notification, the responder:

- Determines whether it has already received a newer UPDATE_SA_ADDRESSES request than this one (if the responder uses a window size greater than one, it is possible that requests are received out of order). If it has, a normal response message (described below) is sent, but no other action is taken.

- If the NO_NATS_ALLOWED notification is present, processes it as described in Section 3.9.

- Checks that the (source IP address, destination IP address) pair in the IP header is acceptable according to local policy. If it is not, replies with a message containing the UNACCEPTABLE_ADDRESSES notification (and possibly COOKIE2).

- Updates the IP addresses in the IKE_SA with the values from the IP header. (Using the address from the IP header is consistent with normal IKEv2, and allows IKEv2 to work with NATs without needing unilateral self-address fixing [UNSAF].)

- Replies with an INFORMATIONAL response:

```
   Initiator                  Responder
  -----------                -----------
                        <--  HDR, SK { [N(NAT_DETECTION_SOURCE_IP),
                                     N(NAT_DETECTION_DESTINATION_IP)],
                                     [N(COOKIE2)] }
```

- If necessary, initiates a return routability check for the new initiator address (see Section 3.7) and waits until the check is completed.

- Updates the IPsec SAs associated with this IKE_SA with the new addresses.

- If NAT Traversal is supported and NAT detection payloads were included, enables or disables NAT Traversal.

When the initiator receives the reply:

- If an address change has occurred after the request was first sent, no MOBIKE processing is done for the reply message because a new UPDATE_SA_ADDRESSES is going to be sent (or has already been sent, if window size greater than one is in use).

- If the response contains the UNEXPECTED_NAT_DETECTED notification, the initiator processes the response as described in Section 3.9.

- If the response contains an UNACCEPTABLE_ADDRESSES notification, the initiator MAY select another addresses and retry the exchange, keep on using the previously used addresses, or disconnect.

- It updates the IPsec SAs associated with this IKE_SA with the new addresses (unless this was already done earlier before sending the request; this is the case when no return routability check was required).

- If NAT Traversal is supported and NAT detection payloads were included, the initiator enables or disables NAT Traversal.

There is one exception to the rule that the responder never updates any IPsec SAs without receiving an UPDATE_SA_ADDRESSES request. If the source address that the responder is currently using becomes unavailable (i.e., sending packets using that source address is no longer possible), the responder is allowed to update the IPsec SAs to use some other address (in addition to initiating the procedure described in the next section).

3.6. Updating Additional Addresses

As described in Section 3.4, both the initiator and responder can send a list of additional addresses in the IKE_AUTH exchange. This information can be updated by sending an INFORMATIONAL exchange request message that contains either one or more ADDITIONAL_IP4_ADDRESS/ADDITIONAL_IP6_ADDRESS notifications or the NO_ADDITIONAL_ADDRESSES notification.

If the exchange initiator has only a single IP address, it is placed in the IP header, and the message contains the NO_ADDITIONAL_ADDRESSES notification. If the exchange initiator has several addresses, one of them is placed in the IP header, and the rest in ADDITIONAL_IP4_ADDRESS/ADDITIONAL_IP6_ADDRESS notifications.

The new list of addresses replaces the old information (in other words, there are no separate add/delete operations; instead, the complete list is sent every time these notifications are used).

The message exchange will look as follows:

```
   Initiator                  Responder
  -----------                -----------
   HDR, SK { [N(ADDITIONAL_*_ADDRESS)+],
             [N(NO_ADDITIONAL_ADDRESSES)],
             [N(NO_NATS_ALLOWED)],
             [N(COOKIE2)] }  -->

                          <--  HDR, SK { [N(COOKIE2)] }
```

When a request containing an ADDITIONAL_IP4_ADDRESS, ADDITIONAL_IP6_ADDRESS, or NO_ADDITIONAL_ADDRESSES notification is received, the exchange responder:

- Determines whether it has already received a newer request to update the addresses (if a window size greater than one is used, it is possible that the requests are received out of order). If it has, a response message is sent, but the address set is not updated.

- If the NO_NATS_ALLOWED notification is present, processes it as described in Section 3.9.

- Updates the set of peer addresses based on the IP header and the ADDITIONAL_IP4_ADDRESS, ADDITIONAL_IP6_ADDRESS, and NO_ADDITIONAL_ADDRESSES notifications.

- Sends a response.

The initiator MAY include these notifications in the same request as UPDATE_SA_ADDRESSES.

If the request to update the addresses is retransmitted using several different source addresses, a new INFORMATIONAL request MUST be sent.

There is one additional complication: when the responder wants to update the address set, the currently used addresses may no longer work. In this case, the responder uses the additional address list received from the initiator, and the list of its own addresses, to determine which addresses to use for sending the INFORMATIONAL request. This is the only time the responder uses the additional address list received from the initiator.

Note that both peers can have their own policies about what addresses are acceptable to use, and certain types of policies may simplify implementation. For instance, if the responder has a single fixed address, it does not need to process the ADDITIONAL_IP4_ADDRESS and ADDITIONAL_IP6_ADDRESS notifications it receives (beyond ignoring unrecognized status notifications, as already required in [IKEv2]). Furthermore, if the initiator has a policy saying that only the responder address specified in local configuration is acceptable, it does not have to send its own additional addresses to the responder (since the responder does not need them except when changing its own address).

3.7. Return Routability Check

Both parties can optionally verify that the other party can actually receive packets at the claimed address. By default, this "return routability check" SHOULD be performed. In environments where the peer is expected to be well-behaved (many corporate VPNs, for instance), or the address can be verified by some other means (e.g., a certificate issued by an authority trusted for this purpose), the return routability check MAY be omitted.

The check can be done before updating the IPsec SAs, immediately after updating them, or continuously during the connection. By default, the return routability check SHOULD be done before updating the IPsec SAs, but in some environments it MAY be postponed until after the IPsec SAs have been updated.

Any INFORMATIONAL exchange can be used for return routability purposes, with one exception (described later in this section): when a valid response is received, we know the other party can receive packets at the claimed address.

To ensure that the peer cannot generate the correct INFORMATIONAL response without seeing the request, a new payload is added to INFORMATIONAL messages. The sender of an INFORMATIONAL request MAY include a COOKIE2 notification, and if included, the recipient of an INFORMATIONAL request MUST copy the notification as-is to the response. When processing the response, the original sender MUST verify that the value is the same one as sent. If the values do not match, the IKE_SA MUST be closed. (See also Section 4.2.5 for the format of the COOKIE2 notification.)

The exception mentioned earlier is as follows: If the same INFORMATIONAL request has been sent to several different addresses (i.e., the destination address in the IKE_SA has been updated after the request was first sent), receiving the INFORMATIONAL response does not tell which address is the working one. In this case, a new INFORMATIONAL request needs to be sent to check return routability.

3.8. Changes in NAT Mappings

IKEv2 performs Dead Peer Detection (DPD) if there has recently been only outgoing traffic on all of the SAs associated with the IKE_SA.

In MOBIKE, these messages can also be used to detect if NAT mappings have changed (for example, if the keepalive interval is too long, or the NAT box is rebooted). More specifically, if both peers support both this specification and NAT Traversal, the NAT_DETECTION_SOURCE_IP and NAT_DETECTION_DESTINATION_IP notifications MAY be included in any INFORMATIONAL request; if the request includes them, the responder MUST also include them in the response (but no other action is taken, unless otherwise specified).

When the initiator is behind a NAT (as detected earlier using the NAT_DETECTION_SOURCE_IP and NAT_DETECTION_DESTINATION_IP notifications), it SHOULD include these notifications in DPD messages and compare the received NAT_DETECTION_DESTINATION_IP notifications with the value from the previous UPDATE_SA_ADDRESSES response (or the IKE_SA_INIT response). If the values do not match, the IP address and/or port seen by the responder has changed, and the initiator SHOULD send UPDATE_SA_ADDRESSES as described in Section 3.5. If the initiator suspects that the NAT mapping has changed, it MAY also skip the detection step and send UPDATE_SA_ADDRESSES immediately. This saves one roundtrip if the NAT mapping has indeed changed.

Note that this approach to detecting NAT mapping changes may cause an extra address update when the IKE_SA is rekeyed. This is because the NAT_DETECTION_DESTINATION_IP hash also includes the IKE Security Parameter Indexes (SPIs), which change when performing rekeying. This unnecessary update is harmless, however.

When MOBIKE is in use, the dynamic updates (specified in [IKEv2], Section 2.23), where the peer address and port are updated from the last valid authenticated packet, work in a slightly different fashion. The host not behind a NAT MUST NOT use these dynamic updates for IKEv2 packets, but MAY use them for ESP packets. This ensures that an INFORMATIONAL exchange that does not contain UPDATE_SA_ADDRESSES does not cause any changes, allowing it to be used for, e.g., testing whether a particular path works.

3.9. NAT Prohibition

Basic IKEv2/IPsec without NAT Traversal support may work across some types of one-to-one "basic" NATs and IPv4/IPv6 translation agents in tunnel mode. This is because the IKEv2 integrity checksum does not cover the addresses in the IP header. This may be considered a problem in some circumstances, because in some sense any modification of the IP addresses can be considered an attack.

This specification addresses the issue by protecting the IP addresses when NAT Traversal has not been explicitly enabled. This means that MOBIKE without NAT Traversal support will not work if the paths contain NATs, IPv4/IPv6 translation agents, or other nodes that modify the addresses in the IP header. This feature is mainly intended for IPv6 and site-to-site VPN cases, where the administrators may know beforehand that NATs are not present, and thus any modification to the packet can be considered an attack.

More specifically, when NAT Traversal is not enabled, all messages that can update the addresses associated with the IKE_SA and/or IPsec SAs (the first IKE_AUTH request and all INFORMATIONAL requests that contain any of the following notifications: UPDATE_SA_ADDRESSES, ADDITIONAL_IP4_ADDRESS, ADDITIONAL_IP6_ADDRESS, NO_ADDITIONAL_ADDRESSES) MUST also include a NO_NATS_ALLOWED notification. The exchange responder MUST verify that the contents of the NO_NATS_ALLOWED notification match the addresses in the IP header. If they do not match, a response containing an UNEXPECTED_NAT_DETECTED notification is sent. The response message is sent to the address and port that the corresponding request came from, not to the address contained in the NO_NATS_ALLOWED notification.

If the exchange initiator receives an UNEXPECTED_NAT_DETECTED notification in response to its INFORMATIONAL request, it SHOULD retry the operation several times using new INFORMATIONAL requests. Similarly, if the initiator receives UNEXPECTED_NAT_DETECTED in the IKE_AUTH exchange, it SHOULD retry IKE_SA establishment several times, starting from a new IKE_SA_INIT request. This ensures that an attacker who is able to modify only a single packet does not unnecessarily cause a path to remain unused. The exact number of retries is not specified in this document because it does not affect interoperability. However, because the IKE message will also be rejected if the attacker modifies the integrity checksum field, a reasonable number here would be the number of retries that is being used for normal retransmissions.

If an UNEXPECTED_NAT_DETECTED notification is sent, the exchange responder MUST NOT use the contents of the NO_NATS_ALLOWED notification for any other purpose than possibly logging the information for troubleshooting purposes.

3.10. Path Testing

IKEv2 Dead Peer Detection allows the peers to detect if the currently used path has stopped working. However, if either of the peers has several addresses, Dead Peer Detection alone does not tell which of the other paths might work.

If required by its address selection policy, the initiator can use normal IKEv2 INFORMATIONAL request/response messages to test whether a certain path works. Implementations MAY do path testing even if the path currently being used is working to, for example, detect when a better (but previously unavailable) path becomes available.

3.11. Failure Recovery and Timeouts

In MOBIKE, the initiator is responsible for detecting and recovering from most failures.

To give the initiator enough time to detect the error, the responder SHOULD use relatively long timeout intervals when, for instance, retransmitting IKEv2 requests or deciding whether to initiate Dead Peer Detection. While no specific timeout lengths are required, it is suggested that responders continue retransmitting IKEv2 requests for at least five minutes before giving up.

3.12. Dead Peer Detection

MOBIKE uses the same Dead Peer Detection method as normal IKEv2, but as addresses may change, it is not sufficient to just verify that the peer is alive, but also that it is synchronized with the address updates and has not, for instance, ignored an address update due to failure to complete return routability test. This means that when there are incoming IPsec packets, MOBIKE nodes SHOULD inspect the addresses used in those packets and determine that they correspond to those that should be employed. If they do not, such packets SHOULD NOT be used as evidence that the peer is able to communicate with this node and or that the peer has received all address updates.

4. Payload Formats

This specification defines several new IKEv2 Notify payload types. See [IKEv2], Section 3.10, for a general description of the Notify payload.

4.1. Notify Messages - Error Types

4.1.1. UNACCEPTABLE_ADDRESSES Notify Payload

The responder can include this notification in an INFORMATIONAL exchange response to indicate that the address change in the corresponding request message (which contained an UPDATE_SA_ADDRESSES notification) was not carried out.

The Notify Message Type for UNACCEPTABLE_ADDRESSES is 40. The Protocol ID and SPI Size fields are set to zero. There is no data associated with this Notify type.

4.1.2. UNEXPECTED_NAT_DETECTED Notify Payload

See Section 3.9 for a description of this notification.

The Notify Message Type for UNEXPECTED_NAT_DETECTED is 41. The Protocol ID and SPI Size fields are set to zero. There is no data associated with this Notify type.

4.2. Notify Messages - Status Types

4.2.1. MOBIKE_SUPPORTED Notify Payload

The MOBIKE_SUPPORTED notification is included in the IKE_AUTH exchange to indicate that the implementation supports this specification.

The Notify Message Type for MOBIKE_SUPPORTED is 16396. The Protocol ID and SPI Size fields are set to zero. The notification data field MUST be left empty (zero-length) when sending, and its contents (if any) MUST be ignored when this notification is received. This allows the field to be used by future versions of this protocol.

4.2.2. ADDITIONAL_IP4_ADDRESS and ADDITIONAL_IP6_ADDRESS Notify Payloads

Both parties can include ADDITIONAL_IP4_ADDRESS and/or ADDITIONAL_IP6_ADDRESS notifications in the IKE_AUTH exchange and INFORMATIONAL exchange request messages; see Section 3.4 and Section 3.6 for more detailed description.

The Notify Message Types for ADDITIONAL_IP4_ADDRESS and ADDITIONAL_IP6_ADDRESS are 16397 and 16398, respectively. The Protocol ID and SPI Size fields are set to zero. The data associated with these Notify types is either a four-octet IPv4 address or a 16-octet IPv6 address.

4.2.3. NO_ADDITIONAL_ADDRESSES Notify Payload

The NO_ADDITIONAL_ADDRESSES notification can be included in an INFORMATIONAL exchange request message to indicate that the exchange initiator does not have addresses beyond the one used in the exchange (see Section 3.6 for more detailed description).

The Notify Message Type for NO_ADDITIONAL_ADDRESSES is 16399. The Protocol ID and SPI Size fields are set to zero. There is no data associated with this Notify type.

4.2.4. UPDATE_SA_ADDRESSES Notify Payload

This notification is included in INFORMATIONAL exchange requests sent by the initiator to update addresses of the IKE_SA and IPsec SAs (see Section 3.5).

The Notify Message Type for UPDATE_SA_ADDRESSES is 16400. The Protocol ID and SPI Size fields are set to zero. There is no data associated with this Notify type.

4.2.5. COOKIE2 Notify Payload

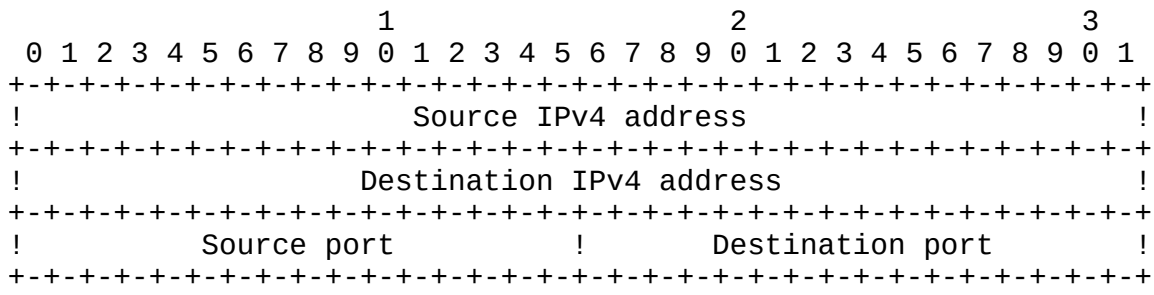
This notification MAY be included in any INFORMATIONAL request for return routability check purposes (see Section 3.7). If the INFORMATIONAL request includes COOKIE2, the exchange responder MUST copy the notification to the response message.

The data associated with this notification MUST be between 8 and 64 octets in length (inclusive), and MUST be chosen by the exchange initiator in a way that is unpredictable to the exchange responder. The Notify Message Type for this message is 16401. The Protocol ID and SPI Size fields are set to zero.

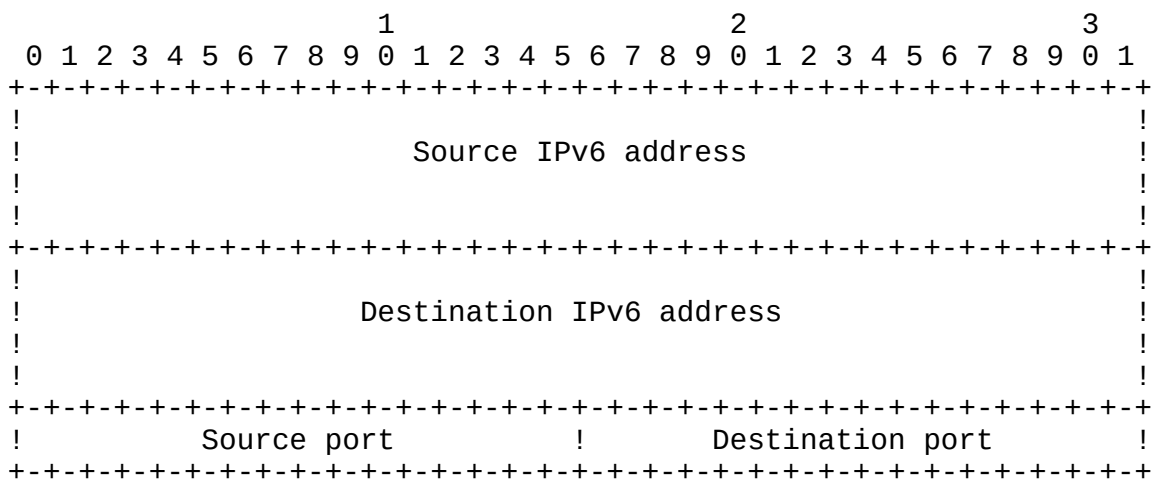
4.2.6. NO_NATS_ALLOWED Notify Payload

See Section 3.9 for a description of this notification.

The Notify Message Type for this message is 16402. The notification data contains the IP addresses and ports from/to which the packet was sent. For IPv4, the notification data is 12 octets long and is defined as follows:



For IPv6, the notification data is 36 octets long and is defined as follows:



The Protocol ID and SPI Size fields are set to zero.

5. Security Considerations

The main goals of this specification are to maintain the security offered by usual IKEv2 procedures and to counter mobility-related threats in an appropriate manner. This section describes new security considerations introduced by MOBIKE. See [IKEv2] for security considerations for IKEv2 in general.

5.1. Traffic Redirection and Hijacking

MOBIKE payloads relating to updating addresses are encrypted, integrity protected, and replay protected using the IKE_SA. This assures that no one except the participants can, for instance, give a control message to change the addresses.

However, as with normal IKEv2, the actual IP addresses in the IP header are not covered by the integrity protection. This means that a NAT between the parties (or an attacker acting as a NAT) can modify the addresses and cause incorrect tunnel header (outer) IP addresses to be used for IPsec SAs. The scope of this attack is limited mainly to denial of service because all traffic is protected using IPsec.

This attack can only be launched by on-path attackers that are capable of modifying IKEv2 messages carrying NAT detection payloads (such as Dead Peer Detection messages). By modifying the IP header of these packets, the attackers can lead the peers to believe a new NAT or a changed NAT binding exists between them. The attack can continue as long as the attacker is on the path, modifying the IKEv2 messages. If this is no longer the case, IKEv2 and MOBIKE mechanisms designed to detect NAT mapping changes will eventually recognize that the intended traffic is not getting through, and will update the addresses appropriately.

MOBIKE introduces the NO_NATS_ALLOWED notification that is used to detect modification, by outsiders, of the addresses in the IP header. When this notification is used, communication through NATs and other address translators is impossible, so it is sent only when not doing NAT Traversal. This feature is mainly intended for IPv6 and site-to-site VPN cases, where the administrators may know beforehand that NATs are not present.

5.2. IPsec Payload Protection

The use of IPsec protection on payload traffic protects the participants against disclosure of the contents of the traffic, should the traffic end up in an incorrect destination or be subject to eavesdropping.

However, security associations originally created for the protection of a specific flow between specific addresses may be updated by MOBIKE later on. This has to be taken into account if the (outer) IP address of the peer was used when deciding what kind of IPsec SAs the peer is allowed to create.

For instance, the level of required protection might depend on the current location of the VPN client, or access might be allowed only from certain IP addresses.

It is recommended that security policies, for peers that are allowed to use MOBIKE, are configured in a manner that takes into account that a single security association can be used at different times through paths of varying security properties.

This is especially critical for traffic selector authorization. The (logical) Peer Authorization Database (PAD) contains the information used by IKEv2 when determining what kind of IPsec SAs a peer is allowed to create. This process is described in [IPsecArch], Section 4.4.3. When a peer requests the creation of an IPsec SA with some traffic selectors, the PAD must contain "Child SA Authorization Data" linking the identity authenticated by IKEv2 and the addresses permitted for traffic selectors. See also [Clarifications] for a more extensive discussion.

It is important to note that simply sending IKEv2 packets using some particular address does not automatically imply a permission to create IPsec SAs with that address in the traffic selectors. However, some implementations are known to use policies where simply being reachable at some address X implies a temporary permission to create IPsec SAs for address X. Here "being reachable" usually means the ability to send (or spoof) IP packets with source address X and receive (or eavesdrop) packets sent to X.

Using this kind of policies or extensions with MOBIKE may need special care to enforce the temporary nature of the permission. For example, when the peer moves to some other address Y (and is no longer reachable at X), it might be necessary to close IPsec SAs with traffic selectors matching X. However, these interactions are beyond the scope of this document.

5.3. Denial-of-Service Attacks against Third Parties

Traffic redirection may be performed not just to gain access to the traffic or to deny service to the peers, but also to cause a denial-of-service attack on a third party. For instance, a high-speed TCP session or a multimedia stream may be redirected towards a victim host, causing its communications capabilities to suffer.

The attackers in this threat can be either outsiders or even one of the IKEv2 peers. In usual VPN usage scenarios, attacks by the peers can be easily dealt with if the authentication performed in the initial IKEv2 negotiation can be traced to persons who can be held responsible for the attack. This may not be the case in all scenarios, particularly with opportunistic approaches to security.

If the attack is launched by an outsider, the traffic flow would normally stop soon due to the lack of responses (such as transport layer acknowledgements). However, if the original recipient of the flow is malicious, it could maintain the traffic flow for an extended period of time, since it often would be able to send the required acknowledgements (see [Aura02] for more discussion).

It should also be noted, as shown in [Bombing], that without ingress filtering in the attacker's network, such attacks are already possible simply by sending spoofed packets from the attacker to the victim directly. Furthermore, if the attacker's network has ingress filtering, this attack is largely prevented for MOBIKE as well. Consequently, it makes little sense to protect against attacks of similar nature in MOBIKE. However, it still makes sense to limit the amplification capabilities provided to attackers, so that they cannot use stream redirection to send a large number of packets to the victim by sending just a few packets themselves.

This specification includes return routability tests to limit the duration of any "third party bombing" attacks by off-path (relative to the victim) attackers. The tests are authenticated messages that the peer has to respond to, and can be performed before the address change takes effect, immediately afterwards, or even periodically during the session. The tests contain unpredictable data, and only someone who has the keys associated with the IKE SA and has seen the request packet can properly respond to the test.

The duration of the attack can also be limited if the victim reports the unwanted traffic to the originating IPsec tunnel endpoint using ICMP error messages or INVALID_SPI notifications. As described in [IKEv2], Section 2.21, this SHOULD trigger a liveness test, which also doubles as a return routability check if the COOKIE2 notification is included.

5.4. Spoofing Network Connectivity Indications

Attackers may spoof various indications from lower layers and the network in an effort to confuse the peers about which addresses are or are not working. For example, attackers may spoof link-layer error messages in an effort to cause the parties to move their traffic elsewhere or even to disconnect. Attackers may also spoof

information related to network attachments, router discovery, and address assignments in an effort to make the parties believe they have Internet connectivity when, in reality, they do not.

This may cause use of non-preferred addresses or even denial of service.

MOBIKE does not provide any protection of its own for indications from other parts of the protocol stack. These vulnerabilities can be mitigated through the use of techniques specific to the other parts of the stack, such as validation of ICMP errors [ICMPAttacks], link layer security, or the use of [SEND] to protect IPv6 Router and Neighbor Discovery.

Ultimately, MOBIKE depends on the delivery of IKEv2 messages to determine which paths can be used. If IKEv2 messages sent using a particular source and destination addresses reach the recipient and a reply is received, MOBIKE will usually consider the path working; if no reply is received even after retransmissions, MOBIKE will suspect the path is broken. An attacker who can consistently control the delivery or non-delivery of the IKEv2 messages in the network can thus influence which addresses actually get used.

5.5. Address and Topology Disclosure

MOBIKE address updates and the ADDITIONAL_IP4_ADDRESS/ ADDITIONAL_IP6_ADDRESS notifications reveal information about which networks the peers are connected to.

For example, consider a host A with two network interfaces: a cellular connection and a wired Ethernet connection to a company LAN. If host A now contacts host B using IKEv2 and sends ADDITIONAL_IP4_ADDRESS/ADDITIONAL_IP6_ADDRESS notifications, host B receives additional information it might not otherwise know. If host A used the cellular connection for the IKEv2 traffic, host B can also see the company LAN address (and perhaps further guess that host A is used by an employee of that company). If host A used the company LAN to make the connection, host B can see that host A has a subscription from this particular cellular operator.

These additional addresses can also disclose more accurate location information than just a single address. Suppose that host A uses its cellular connection for IKEv2 traffic, but also sends an ADDITIONAL_IP4_ADDRESS notification containing an IP address corresponding to, say, a wireless LAN at a particular coffee shop location. It is likely that host B can now make a much better guess at A’s location than would be possible based on the cellular IP address alone.

Furthermore, as described in Section 3.4, some of the addresses could also be private addresses behind a NAT.

In many environments, disclosing address information is not a problem (and indeed it cannot be avoided if the hosts wish to use those addresses for IPsec traffic). For instance, a remote access VPN client could consider the corporate VPN gateway sufficiently trustworthy for this purpose. Furthermore, the ADDITIONAL_IP4_ADDRESS and ADDITIONAL_IP6_ADDRESS notifications are sent encrypted, so the addresses are not visible to eavesdroppers (unless, of course, they are later used for sending IKEv2/IPsec traffic).

However, if MOBIKE is used in some more opportunistic approach, it can be desirable to limit the information that is sent. Naturally, the peers do not have to disclose any addresses they do not want to use for IPsec traffic. Also, as noted in Section 3.6, an initiator whose policy is to always use the locally configured responder address does not have to send any ADDITIONAL_IP4_ADDRESS/ ADDITIONAL_IP6_ADDRESS payloads.

6. IANA Considerations

This document does not create any new namespaces to be maintained by IANA, but it requires new values in namespaces that have been defined in the IKEv2 base specification [IKEv2].

This document defines several new IKEv2 notifications whose values have been allocated from the "IKEv2 Notify Message Types" namespace.

Notify Messages - Error Types	Value
UNACCEPTABLE_ADDRESSES	40
UNEXPECTED_NAT_DETECTED	41

Notify Messages - Status Types	Value
MOBIKE_SUPPORTED	16396
ADDITIONAL_IP4_ADDRESS	16397
ADDITIONAL_IP6_ADDRESS	16398
NO_ADDITIONAL_ADDRESSES	16399
UPDATE_SA_ADDRESSES	16400
COOKIE2	16401
NO_NATS_ALLOWED	16402

These notifications are described in Section 4.

7. Acknowledgements

This document is a collaborative effort of the entire MOBIKE WG. We would particularly like to thank Jari Arkko, Tuomas Aura, Marcelo Bagnulo, Stephane Beaulieu, Elwyn Davies, Lakshminath Dondeti, Francis Dupont, Paul Hoffman, James Kempf, Tero Kivinen, Pete McCann, Erik Nordmark, Mohan Parthasarathy, Pekka Savola, Bill Sommerfeld, Maureen Stillman, Shinta Sugimoto, Hannes Tschofenig, and Sami Vaarala. This document also incorporates ideas and text from earlier MOBIKE-like protocol proposals, including [AddrMgmt], [Kivinen], [MOPO], and [SMOBIKE], and the MOBIKE design document [Design].

8. References

8.1. Normative References

[IKEv2] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol", RFC 4306, December 2005.

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

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

8.2. Informative References

[AddrMgmt] Dupont, F., "Address Management for IKE version 2", Work in Progress, November 2005.

[Aura02] Aura, T., Roe, M., and J. Arkko, "Security of Internet Location Management", Proc. 18th Annual Computer Security Applications Conference (ACSAC), December 2002.

[Bombing] Dupont, F., "A note about 3rd party bombing in Mobile IPv6", Work in Progress, December 2005.

[Clarifications] Eronen, P. and P. Hoffman, "IKEv2 Clarifications and Implementation Guidelines", Work in Progress, February 2006.

[DNA4] Aboba, B., Carlson, J., and S. Cheshire, "Detecting Network Attachment in IPv4 (DNAv4)", RFC 4436, March 2006.

[DNA6] Narayanan, S., Daley, G., and N. Montavont, "Detecting Network Attachment in IPv6 - Best Current Practices for hosts", Work in Progress, October 2005.

[Design] Kivinen, T. and H. Tschofenig, "Design of the MOBIKE protocol", Work in Progress, January 2006.

[ICMPAttacks] Gont, F., "ICMP attacks against TCP", Work in Progress, October 2005.

[Kivinen] Kivinen, T., "MOBIKE protocol", Work in Progress, February 2004.

[MIP4] Perkins, C., "IP Mobility Support for IPv4", RFC 3344, August 2002.

[MIP6] Johnson, D., Perkins, C., and J. Arkko, "Mobility Support in IPv6", RFC 3775, June 2004.

[MOPO] Eronen, P., "Mobility Protocol Options for IKEv2 (MOPO-IKE)", Work in Progress, February 2005.

[RFC2461] Narten, T., Nordmark, E., and W. Simpson, "Neighbor Discovery for IP Version 6 (IPv6)", RFC 2461, December 1998.

[SEND] Arkko, J., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, March 2005.

[SMOBIKE] Eronen, P. and H. Tschofenig, "Simple Mobility and Multihoming Extensions for IKEv2 (SMOBIKE)", Work in Progress, March 2004.

[STUN] Rosenberg, J., Weinberger, J., Huitema, C., and R. Mahy, "STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs)", RFC 3489, March 2003.

[UNSAF] Daigle, L., "IAB Considerations for UNilateral Self-Address Fixing (UNSAF) Across Network Address Translation", RFC 3424, November 2002.

Appendix A. Implementation Considerations

A.1. Links from SPD Cache to Outbound SAD Entries

[IPsecArch], Section 4.4.2, says that "For outbound processing, each SAD entry is pointed to by entries in the SPD-S part of the SPD cache". The document does not specify how exactly this "pointing" is done, since this is an implementation detail that does not have to be standardized.

However, it is clear that the links between the SPD cache and the SAD have to be done correctly to ensure that outbound packets are sent over the right SA. Some implementations are known to have problems in this area.

In particular, simply storing the (remote tunnel header IP address, remote SPI) pair in the SPD cache is not sufficient, since the pair does not always uniquely identify a single SAD entry. For instance, two hosts behind the same NAT can accidentally happen to choose the same SPI value. The situation can also occur when a host is assigned an IP address previously used by some other host, and the SAs associated with the old host have not yet been deleted by Dead Peer Detection. This may lead to packets being sent over the wrong SA or, if the key management daemon ensures the pair is unique, denying the creation of otherwise valid SAs.

Storing the remote tunnel header IP address in the SPD cache may also complicate the implementation of MOBIKE, since the address can change during the lifetime of the SA. Thus, we recommend implementing the links between the SPD cache and the SAD in a way that does not require modification when the tunnel header IP address is updated by MOBIKE.

A.2. Creating Outbound SAs

When an outbound packet requires IPsec processing but no suitable SA exists, a new SA will be created. In this case, the host has to determine (1) who is the right peer for this SA, (2) whether the host already has an IKE_SA with this peer, and (3) if no IKE_SA exists, the IP address(es) of the peer for contacting it.

Neither [IPsecArch] nor MOBIKE specifies how exactly these three steps are carried out. [IPsecArch], Section 4.4.3.4, says:

For example, assume that IKE A receives an outbound packet destined for IP address X, a host served by a security gateway. RFC 2401 [RFC2401] and this document do not specify how A determines the address of the IKE peer serving X. However, any peer contacted by A as the presumed representative for X must be registered in the PAD in order to allow the IKE exchange to be authenticated. Moreover, when the authenticated peer asserts that it represents X in its traffic selector exchange, the PAD will be consulted to determine if the peer in question is authorized to represent X.

In step 1, there may be more than one possible peer (e.g., several security gateways that are allowed to represent X). In step 3, the host may need to consult a directory such as DNS to determine the peer IP address(es).

When performing these steps, implementations may use information contained in the SPD, the PAD, and possibly some other implementation-specific databases. Regardless of how exactly the steps are implemented, it is important to remember that IP addresses can change, and that an IP address alone does not always uniquely identify a single IKE peer (for the same reasons as why the combination of the remote IP address and SPI does not uniquely identify an outbound IPsec SA; see Appendix A.1). Thus, in steps 1 and 2 it may be easier to identify the "right peer" using its authenticated identity instead of its current IP address. However, these implementation details are beyond the scope of this specification.

Author’s Address

Pasi Eronen (editor)
Nokia Research Center
P.O. Box 407
FIN-00045 Nokia Group
Finland

EMail: pasi.eronen@nokia.com

Full Copyright Statement

Intellectual Property

Acknowledgement

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).