MOBIKE Working Group
Internet-Draft
Expires: April 27, 2006

P. Eronen, Ed.
Nokia

October 24, 2005

IKEv2 Mobility and Multihoming Protocol (MOBIKE)
draft-ietf-mobike-protocol-05.txt

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 hosts to update the (outer) IP addresses associated with IKEv2 and tunnel mode IPsec Security Associations. A mobile 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, the IPsec and IKE 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. 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 also 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 leaving the office the laptop could start using GPRS; when the user arrives home, the laptop could switch the 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 discovering the addresses when the IKE_SA is first established. This implies that 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 unidirectional paths.

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.

Readers are encouraged to 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 talks about updating the source/destination addresses of an IPsec SA, it means updating IPsec-related state so that outgoing ESP/AH packets use those addresses in the tunnel header. Depending on how the nominal division between Security Association Database (SAD), Security Policy Database (SPD), and Peer Authorization Database (PAD) described in [IPsecArch] is 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).

Also, 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 Runs

A simple MOBIKE exchange in a mobile scenario is illustrated below:

```
      Initiator                  Responder
     -----------                -----------
   1) (IP_I1:500 -> IP_R1:500)
      HDR, SAi1, KEi, Ni,
           N(NAT_DETECTION_*_IP)  -->

                              <--  (IP_R1:500 -> IP_I1:500)
                                   HDR, SAr1, KEr, Nr,
                                        N(NAT_DETECTION_*_IP)

   2) (IP_I1:500 -> IP_R1:500)
      HDR, SK { IDi, CERT, AUTH,
                CP(CFG_REQUEST),
                SAi2, TSi, TSr,
                N(MOBIKE_SUPPORTED) }  -->

                              <--  (IP_R1:500 -> IP_I1:500)
                                   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_I2:500 -> IP_R1:500)
      HDR, SK { N(UPDATE_SA_ADDRESSES),
                N(NAT_DETECTION_*_IP) }  -->

                              <-- (IP_R1:500 -> IP_I2:500)
                                  HDR, SK { N(NAT_DETECTION_*_IP) }

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

   4)                         <-- (IP_R1:500 -> IP_I2:500)
                                  HDR, SK { N(COOKIE2) }

      (IP_I2:500 -> IP_R1:500)
      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 so 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_*_IP)  -->

                          <--  (IP_R1:500 -> IP_I1:500)
                               HDR, SAr1, KEr, Nr,
                                    N(NAT_DETECTION_*_IP)

2) (IP_I1:500 -> IP_R1:500)
   HDR, SK { IDi, CERT, AUTH,
             CP(CFG_REQUEST),
             SAi2, TSi, TSr,
             N(MOBIKE_SUPPORTED) }  -->

                          <--  (IP_R1:500 -> IP_I1:500)
                               HDR, SK { IDr, CERT, AUTH,
                                         CP(CFG_REPLY),
                                         SAr2, TSi, TSr,
                                         N(MOBIKE_SUPPORTED),
                                         N(ADDITIONAL_IPV4_ADDRESS) }
```

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

```
3) (IP_I1:500 -> IP_R1:500)
   HDR, SK { N(NAT_DETECTION_*_IP) }  -->

   (IP_I1:500 -> IP_R1:500)
   HDR, SK { N(NAT_DETECTION_*_IP) }  -->

   ...
```

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

```
4) (IP_I1:500 -> IP_R2:500)
   HDR, SK { N(NAT_DETECTION_*_IP) }

                          <--  (IP_R2:500 -> IP_I1:500)
                               HDR, SK { N(NAT_DETECTION_*_IP) }
```

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

```
5) (IP_I1:500 -> IP_R2:500)
   HDR, SK { N(UPDATE_SA_ADDRESSES),
             N(NAT_DETECTION_*_IP),
             N(COOKIE2) }  -->

                          <--  (IP_R2:500 -> IP_I1:500)
                               HDR, SK { N(NAT_DETECTION_*_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 talks simply about 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 in 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 does not fully support the case where the responder's addresses change.

"Does not fully support" means 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 as follows: If the IKE_SA_INIT request contains the NAT_DETECTION_*_IP notifications and the responder supports NAT Traversal, the values are initialized from the IP header of the first IKE_AUTH request. Otherwise, the values are initialized from the IP header of the IKE_SA_INIT request.

The addresses are taken from the IKE_AUTH request when NAT Traversal is being used 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).

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).

```
   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 usually never updates 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 ADDITIONAL_IP4/6_ADDRESS or NO_ADDITIONAL_ADDRESS notifications is received. This means the peer's addresses may have changed. This is particularly important if the announced set of address 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_*_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_*_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, it 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 current addresses, or disconnect.

- It updates the IPsec SAs associated with this IKE_SA with the new addresses (unless this was already done before sending the request).

- If NAT Traversal is supported and NAT detection payloads were included, it 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_IP4/6_ADDRESS notifications or the NO_ADDITIONAL_ADDRESSES notification. 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 ADDITIONAL_*_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 ADDITIONAL_IP4/6_ADDRESS or NO_ADDITIONAL_ADDRESS 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 need to process ADDITIONAL_*_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., the address is included in the peer's certificate), 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 4.6 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, NAT_DETECTION_*_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 NAT_DETECTION_*_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 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 IKE_SA_INIT request and all INFORMATIONAL requests that contain UPDATE_SA_ADDRESSES and/or ADDITIONAL_IP4/6_ADDRESS notifications) 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 (and in the case of the IKE_SA_INIT exchange, no state is created at the responder). 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 request, it SHOULD retry the operation several times using new IKE_SA_INIT/INFORMATIONAL requests. This ensures that an attacker who is able to modify only a single packet does not unnecessarily cause a path to remain unused.

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.

4. Payload Formats

This specification defines several new IKEv2 Notify payload types. The UNACCEPTABLE_ADDRESSES and UNEXPECTED_NAT_DETECTED notifications are "error types"; the other notifications are "status types". See [IKEv2] Section 3.10 for a general description of the Notify payload.

4.1. 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 TBD-BY-IANA1. 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. ADDITIONAL_IP4/6_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 TBD-BY-IANA2 and TBD-BY-IANA3, 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.

4.3. 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 TBD-BY-IANA4. The Protocol ID and SPI Size fields are set to zero. There is no data associated with this Notify type.

4.4. 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 TBD-BY-IANA5. The Protocol ID and SPI Size fields are set to zero. There is no data associated with this Notify type.

4.5. 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 TBD-BY-IANA6. The Protocol ID and SPI Size fields are set to zero. There is no data associated with this Notify type.

4.6. 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 TBD-BY-IANA7. The Protocol ID and SPI Size fields are set to zero.

4.7. NO_NATS_ALLOWED Notify Payload

See Section 3.9 for a description of this notification.

The data field of this notification contains the following information: the IP address from which the packet was sent (4 or 16 bytes), the port from which the packet was sent (2 bytes, network byte order), the IP addresss to which the packet was sent (4 or 16 bytes), and the port to which the packet was sent (2 bytes, network byte order). The total length of the data field is thus 12 bytes for IPv4 and 36 bytes for IPv6. The Notify Message Type for this message is TBD-BY-IANA8. The Protocol ID and SPI Size fields are set to zero.

4.8. UNEXPECTED_NAT_DETECTED Notify Payload

See Section 3.9 for a description of this notification.

The Notify Message Type for UNEXPECTED_NAT_DETECTED is TBD-BY-IANA9. The Protocol ID and SPI Size fields are set to zero. There is no

data associated with this Notify type.

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. Such modifications can only be performed by attackers who are on the path and capable of modifying the When this notification is used, communication through NATs and other address translators is impossible, so it is sent only when not doing NAT Traversal.

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 eavesdropped along the way.

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 level of required protection depends on, for instance, the current location of the VPN client.

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.

5.3. Denial-of-Service Attacks Against Third Parties

Traffic redirection may be performed not just to gain access to the traffic (not very interesting because it is encrypted) or to deny service to the peers, but also to cause a denial-of-service attack for 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.

Normally, such attacks would expire in a short time frame due to the lack of responses (such as transport layer acknowledgements) from the victim. However, as described in [Aura02], malicious participants would typically be able to spoof such acknowledgements and maintain the traffic flow for an extended period of time. For instance, if the attacker opened the TCP stream itself before redirecting it to the victim, the attacker becomes aware of the sequence number space used in this particular session.

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 either 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.

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 ADDITIONAL_IP4/6_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/MOBIKE and sends ADDITIONAL_IP4/6_ADDRESS notifications, host B receives additional information it might not otherwise know. If host A used the cellular connection for the IKEv2/MOBIKE 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/MOBIKE 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_IP4/6_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_IP4/6_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 are to be allocated from the "IKEv2 Notify Message Types" namespace.

Notify Message	Value
MOBIKE_SUPPORTED	TBD-BY-IANA1 (16396..40959)
ADDITIONAL_IP4_ADDRESS	TBD-BY-IANA2 (16396..40959)
ADDITIONAL_IP6_ADDRESS	TBD-BY-IANA3 (16396..40959)
NO_ADDITIONAL_ADDRESSES	TBD-BY-IANA4 (16396..40959)
UPDATE_SA_ADDRESSES	TBD-BY-IANA5 (16396..40959)
UNACCEPTABLE_ADDRESSES	TBD-BY-IANA6 (40..8191)
COOKIE2	TBD-BY-IANA7 (16396..40959)
NO_NATS_ALLOWED	TBD-BY-IANA8 (16396..40959)
UNEXPECTED_NAT_DETECTED	TBD-BY-IANA9 (40..8191)

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, Francis Dupont, Paul Hoffman, Tero Kivinen, and Hannes Tschofenig. This document also incorporates ideas and text from earlier MOBIKE 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", draft-ietf-ipsec-ikev2-17 (work in progress), October 2004.

[IPsecArch] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", draft-ietf-ipsec-rfc2401bis-06 (work in progress), March 2005.

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

[UDPEncap] Huttunen, A., Swander, B., Volpe, V., DiBurro, L., and M. Stenberg, "UDP Encapsulation of IPsec ESP Packets", RFC 3948, January 2005.

8.2. Informative References

[AddrMgmt] Dupont, F., "Address Management for IKE version 2", draft-dupont-ikev2-addrmgmt-07 (work in progress), May 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", draft-dupont-mipv6-3bombing-02 (work in progress), June 2005.

[DNA4] Aboba, B., "Detecting Network Attachment (DNA) in IPv4", draft-ietf-dhc-dna-ipv4-15 (work in progress), August 2005.

[DNA6] Narayanan, S., Daley, G., and N. Montavont, "Detecting Network Attachment in IPv6 - Best Current Practices for hosts", draft-ietf-dna-hosts-01 (work in progress), June 2005.

[Design] Kivinen, T. and H. Tschofenig, "Design of the MOBIKE protocol", draft-ietf-mobike-design-02 (work in progress), February 2005.

[ICMPAttacks] Gont, F., "ICMP attacks against TCP", draft-gont-tcpm-icmp-attacks-03 (work in progress), December 2004.

[Kivinen] Kivinen, T., "MOBIKE protocol", draft-kivinen-mobike-protocol-00 (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)", draft-eronen-mobike-mopo-02 (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)", draft-eronen-mobike-simple-00 (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. Changelog

(This section should be removed by the RFC editor.)

Changes from -04 to -05:

- Editorial fixes and clarifications (issue 44, 47, 48, 49, 50, 53, 62, 66, 67, 68, 69).
- Ignore data in MOBIKE_SUPPORTED (issue 61).

Changes from -03 to -04:

- Copy-editing done by the RFC editor.

Changes from -02 to -03:

- Editorial fixes and clarifications (issues 42 and 43).
- Clarified IANA considerations (issue 42).
- Added security considerations about address and topology disclosure (issue 42).
- Added a suggestion about retransmission timeout (issue 42).

- Change dynamic address updates: MUST NOT do them based on IKEv2 packets, MAY do based on ESP (issue 34).
- Mandate NAT prohibition if not doing NAT traversal (issue 41).
- Clarified security considerations related to NATs (issue 41).
- Don't use SHA-1 in NO_NATS_ALLOWED, just send the addresses (issue 42).
- Added a short section about path testing.
- Added an example protocol run in Section 1.

Changes from -01 to -02:

- Moved MOBIKE_SUPPORTED from IKE_SA_INIT to IKE_AUTH (issues 35, 37).
- Changed terminology related to NAT prohibition (issues 22, 24).
- Rewrote much of the ADDITIONAL_*_ADDRESS text, added NO_ADDITIONAL_ADDRESSES notification.
- Use NAT detection payloads to detect changes in NAT mappings (issue 34).
- Removed separate PATH_TEST message (issue 34).
- Clarified processing of UNACCEPTABLE_ADDRESSES when request has been sent using several different addresses (issue 36).
- Clarified changing of ports 500/4500 (issue 33).
- Updated security considerations (issues 27 and 28).
- No need to include COOKIE2 in non-RR messages (issue 32).
- Many editorial fixes and clarifications (issue 38, 40).
- Use the terms initiator and responder more consistently.
- Clarified that this document does not solve all problems in MOBIKE WG charter (issue 40).

Changes from -00 to -01:

- Editorial fixes and small clarifications (issues 21, 25, 26, 29).
- Use Protocol ID zero for notifications (issue 30).
- Separate ADDITIONAL_*_ADDRESS payloads for IPv4 and IPv6 (issue 23).
- Use the word "path" only in senses that include the route taken (issue 29).

Author's Address

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

Email: pasi.eronen@nokia.com

Intellectual Property Statement

Disclaimer of Validity

Copyright Statement

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.