Network Working Group
Internet-Draft
Expires: August 22, 2005

P. Eronen
Nokia

February 21, 2005

Mobility Protocol Options for IKEv2 (MOPO-IKE)
draft-eronen-mobike-mopo-02.txt

Status of this Memo

Copyright Notice

Abstract

This document describes a mobility and multihoming extension to the IKEv2 protocol. The main purpose of this extension is to update the (outer) addresses associated with IKE and IPsec Security Associations. The extension also includes features that assist in selecting which addresses to use, and verify return routability during and after updates. It is also able to work together with NAT Traversal in some scenarios.

1. Introduction

1.1 Motivation

1.1.1 Scenario 1: Mobile client

The basic MOBIKE scenario is a mobile device such as a laptop that may have several different network interfaces, and wishes to keep a connection to a VPN gateway working while moving. For instance, a user could start from fixed Ethernet in the office, and then disconnect the laptop and move to office wireless LAN. When leaving the office the laptop could start using GPRS, and switch to a different wireless LAN when the user arrives home.

MOBIKE provides a way to notify the VPN gateway of the user's current IP address without requiring a new IKE SA (which may require user interaction for authentication). Since the IP address used inside the tunnel (assigned by the VPN gateway using Configuration Payloads) does not change, transport layer connections are not broken.

Mobility detection, policy issues such as which interface should be used, and implementation issues -- such as a possible interface (and division of work) between a "mobility and multihoming control module" and "IKE module" inside a host -- are beyond the scope of MOBIKE.

1.1.2 Scenario 2: Mobile client with multihomed gateway

The second scenario has again a mobile laptop, but this time the VPN gateway has several network interfaces.

MOBIKE provides a mechanism for the gateway to notify the client of its addresses (obviously, the client needs to know at least one address to contact the VPN gateway in the first place) and changes in their availability.

MOBIKE also allows the client to determine which of the gateway's addresses can be used when the client changes its own address. This is required since it cannot be assumed that all of the gateway's interfaces are reachable from all networks the client may connect to. The client may also obtain this information in some other way external to MOBIKE, and policy issues about which address should be actually used are beyond the scope of MOBIKE.

1.1.3 Scenario 3: Two multihomed gateways

The third scenario has two VPN gateways connecting, for instance, two different sites of a company. Both gateways have several network connections (possible from different ISPs) to provide better

availability in case of network failures.

Here MOBIKE provides a way to inform the other end of alternative addresses, and switch to some other connection if the currently used path fails. The failures may be detected by either using IKE dead peer detection messages, or some external mechanism beyond the scope of MOBIKE.

1.2 Summary of MOPO-IKE

Assumptions:

- Both the initiator and the responder can have multiple IPv4 and/or IPv6 addresses. It is not assumed that all the paths work; in other words, only some of the initiator's addresses may work with a particular responder address, and vice versa.

- It is assumed that the initiator knows at least one of the responder's IP addresses beforehand, and can reach that address from one of its own addresses when the IKE_SA is created. It is not required that this path continues to work during the whole lifetime of the IKE_SA.

- Some of the paths between the initiator and the responder may contain NATs or stateful packet filters. The protocol assumes that the initiator can send packets to the responder at any time (or in other words, it is the one "behind" the stateful device), but the reverse is not necessarily true.

- Network paths may stop working unexpectedly (causing a need to change to some other path in "break-before-make" fashion), but in some cases, traffic may be moved to a new path while the old path is still working ("make-before-break").

Based on these assumptions, the design goals of MOPO-IKE were to (1) be reasonably simple to implement, especially for nodes whose addresses do not change, (2) be compatible with NAT Traversal, but (3) still be able handle cases where both peers have multiple addresses.

The protocol be summarized as follows:

- If the responder has other addresses than the one used for initial contact, it can inform the initiator of these when the IKE_SA is created.

- The initiator selects which path is used for IPsec SAs, and informs the responder about it. The same path (pair of addresses)

is used in both initiator-to-responder and responder-to-initiator directions. Making the decision on the initiator side is consistent with how normal IKEv2 works, and using this path also for the responder-to-initiator direction makes sense especially with mobile clients where the client is in a better position to decide which network interface should be used for "downlink" traffic.

- When a path used for IPsec SAs is changed, NAT Traversal can be enabled or disabled as needed.

- If the responder's set of addresses changes, it can inform the initiator about this. To allow this feature to work even when there is partial connectivity, the initiator can also inform the responder of other than currently used addresses. This feature does not fully work with all types of NATs and stateful packet filters; all other features do.

- To support smoother make-before-break, and quicker recovery in case of break-before-make, the initiator can determine whether a path works or not before deciding to move the traffic.

- Both the initiator and the responder can optionally verify that the other party can actually receive packets at the claimed address. This "return routability check" can be done immediately after updating the IPsec SAs, or continuously during the connection.

- Both the initiator and the responder can have a policy that prevents the use of paths that contain NATs, IPv4/IPv6 translation agents, or other nodes that modify the addresses in the IP header. This feature is mainly intended for 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 to be an "attack".

1.3 Security association viewpoint

The main purpose of this extension is to modify state associated with IKE_SA and IPsec SAs that is normally initialized when the SA is created, and not changed afterwards.

In particular, this extension considers the following state associated with IKE_SA and outbound IPsec SAs (conceptually speaking; an implementation could store this information in some other way as well):

- IKE_SA
  - local_address (source address for IKE requests)
  - local_port (source port for IKE requests, either 500 or 4500)
  - peer_address (destination address for IKE requests)
  - peer_port (destination port for IKE requests)
- outbound IPsec SAs
  - local_address (tunnel header source address)
  - peer_address (tunnel header destination address)
  - peer_port (destination port if UDP encapsulation is used)
  - udp_encapsulation flag
  - send_keepalives flag
  - automatically_update_peer_address flag

Note that both IKE_SA and outbound IPsec SAs are considered to have a single pair of (source,destination) addresses at a time. These are the addresses used for IKE requests (including retransmissions of previous requests) and outbound ESP/AH packets.

In addition, the IKE_SA contains additional state specific to this extension. This state is used to to store information about addresses that are not currently active (see Section 2.3 for details).

1.4 Terminology

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].

IPsec Security Association (SA)

An ESP or AH Security Association.

Path

A particular combination of source IP address and destination IP address (and possibly ports?).

2. Protocol exchanges

2.1 Signaling support for this specification

Implementations that support this specification MUST include a MOPO_SUPPORTED notification in the IKE_SA_INIT request and response messages.

```
   Initiator                   Responder
  -----------                 -----------
   HDR, SAi1, KEi, Ni,
        N(MOPO_SUPPORTED),
        [N(NAT_DETECTION_*_IP)]  -->


                           <--  HDR, SAr1, KEr, Nr, [CERTREQ],
                                N(MOPO_SUPPORTED),
                                [N(NAT_DETECTION_*_IP)]
```

The MOPO_SUPPORTED notification payload is described in Section 3.

2.2 Additional addresses

Both the initiator and responder MAY include one or more ADDITIONAL_ADDRESS notification payloads 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,
             SAi2, TSi, TSr,
             [N(ADDITIONAL_ADDRESS)*],
             [CP(CFG_REQUEST)] }  -->

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

The recipient stores this information in the "additional_addresses" list, but no other action is taken at this time.

2.3 Changing path of IPsec SAs

This extension is based on the idea that the initiator of the IKE_SA decides what addresses are used in the IPsec SAs. That is, the responder never updates any IPsec SAs without receiving an explicit CHANGE_PATH request from the initiator. (As described below, the responder can, however, update the IKE_SA in some circumstances.)

The description in this section assumes that the initiator has already decided what the new addresses should be. How this decision is made is beyond the scope of this specification. When this decision has been made, the initiator

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

- If NAT Traversal is not enabled, 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.

- When the window size allows, sends an INFORMATIONAL request containing the CHANGE_PATH notification payload (which does not contain any data), and clears the "pending_update" flag.

```
   Initiator                   Responder
  -----------                 -----------
   HDR, SK {N(CHANGE_PATH),
            N(COOKIE2),
            [N(NAT_DETECTION_*),]
            [N(NAT_PREVENTION)]} -->
```

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

When processing an INFORMATIONAL request containing the CHANGE_PATH notification, the responder

- Compares the Message ID with the latest_update_received counter in the IKE_SA. If latest_update_received is greater than the received Message ID, the reply is sent as usual, but no other action is taken; otherwise, updates the latest_update_received counter.

- If the NAT_PREVENTION payload is present, processes it as described in Section 2.7.

- 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 "HDR, SK {COOKIE2, N(UNACCEPTABLE_PATH)}".

- Updates the IP addresses in the IKE_SA and IPsec SAs with the values from the IP header.

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

- Replies with an INFORMATION response:

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

When the initiator receives the reply, it

- If the response contains the NAT_PREVENTED payload, processes it as described in Section 2.7.

- If the response contains an UNACCEPTABLE_PATH notification payload, TBD.

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

2.4 Updating additional addresses

As described in Section 2.2, both the initiator and responder can send a list of additional addresses (in addition to the one used for IKE_SA_INIT/IKE_AUTH exchange) to the initiator in the IKE_AUTH exchange. If this list of addresses changes, a new list can be sent in any INFORMATIONAL exchange request message.

When the responder (of the original IKE_SA) receives an INFORMATIONAL request containing ADDITIONAL_ADDRESS payloads, it simply stores the information, but no other action is taken.

```
   Initiator                  Responder
  -----------                -----------
   HDR, SK { N(ADDITIONAL_ADDRESS)+,
             N(COOKIE2) }  -->

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

When the initiator receives an INFORMATIONAL request containing ADDITIONAL_ADDRESS, it stores the information and also determines whether the currently used path needs to be changed; if it does, the initiator proceeds as described in the previous section.

```
   Initiator                  Responder
  -----------                -----------
                        <--  HDR, SK { N(ADDITIONAL_ADDRESS)+,
                                       N(COOKIE2) }

   HDR, SK { N(COOKIE2) }  -->
```

If the implementation supports window sizes greater than one, it also has to keep track of the Message ID of the latest update it has received, to avoid the situation where new information is overwritten by older.

There is one additional complication: when the responder wants to send a new additional address list, the currently used path may no longer work. In this case, the responder uses the additional address list received from the initiator, the list of its own addresses, and, if necessary, the path testing feature (see Section 2.5) to determine a path that works, updates the addresses in the IKE_SA (but not IPsec SAs), and then sends the INFORMATIONAL request. This is the only time the responder uses the additional address list received from the initiator.

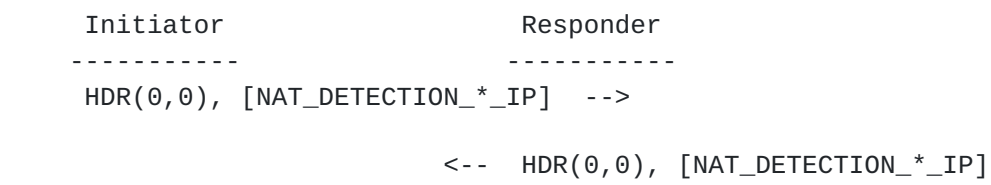
Note that both peers can have their own policies about what addresses or paths are acceptable to use. A minimal "mobile client" could have a policy that says that only the responder's address specified in local configuration is acceptable. This kind of client does not have to send or process ADDITIONAL_ADDRESS notification payloads. Similarly, a simple "VPN gateway" that has only a single address, and is not going to change it, does not need to send or understand ADDITIONAL_ADDRESS notification payloads.

2.5 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, DPD alone does not indicate which of the other paths might work.

The path testing feature allows parties to find out what action is required when no responses are received; that is, to find a path (combination of addresses) that still works. It also removes the need configure information about (lack of) routing relationships in the case where not all possible combinations of addresses work.

MOPO-IKE introduces a new IKEv2 exchange type, PATH_TEST, for testing connectivity. This exchange is not part of any IKE_SA, so it is not cryptographically protected. It also does not result in the responder keeping any state.



The reason for introducing a new exchange type, instead of using INFORMATIONAL exchanges, is to simplify implementations by allowing MOPO-IKE to work with window size 1.

Performing path testing over several different paths is not required if the node has other information that enables it to select which path should be used. In this case, the PATH_TEST exchange can be skipped. Implementations MAY do path testing even if the currently used path is working to e.g. detect when a better but previously unavailable path becomes available, or to speed up recovery in fault situations.

Implementations that perform path testing MUST take steps to avoid causing unnecessary congestion. TBD: add some more details here.

2.6 Return routability check

Both the initiator and the responder can optionally verify that the other party can actually receive packets at the claimed address. This "return routability check" can be done immediately after updating the IPsec SAs, or continuously during the connection. Any INFORMATIONAL exchange can be used for return routability purposes (with one exception, described below): 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 all INFORMATIONAL messages. The sender of an INFORMATIONAL request MUST include a COOKIE2 notification payload, and the recipient of an INFORMATIONAL request MUST copy the payload 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.

There is one additional issue that must be taken into account. If the destination address in the IKE_SA has been updated after the

INFORMATIONAL request was sent, then it is possible that the request has been sent to several different addresses. In this case, receiving the INFORMATIONAL response does not tell which address is the working one; thus, a new INFORMATIONAL request needs to be sent.

2.7 NAT prevention

IKEv2/IPsec implementations that do not support NAT Traversal can, in fact, work across some types of one-to-one "basic" NATs and IPv4/IPv6 translation agents in tunnel mode. Some people feel that this is a problem, since in some sense any modification of the IP addresses could be considered to be an attack.

The "NAT prevention" feature allows both the initiator and responder to have a policy that prevents the use of paths that contain NATs, IPv4/IPv6 translation agents, or other nodes that modify the addresses in the IP header. This feature is mainly intended for 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 to be an "attack".

This specification addresses the issue as follows. 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 message IP header. The NAT_PREVENTION payload is used to guarantee that NATs have not modified the address used in IKE_SA. However, all response messages are still sent to the address and port the corresponding request came from.

The initiator MAY include a NAT_PREVENTION payload in an IKE_SA_INIT request. The responder MUST compare the NAT_PREVENTION payload with the values from the IP header. If they do not match, the responder replies with "HDR(A,0), N(NAT_PREVENTED)" and does not create any state.

If the values do match, the responder initializes (local_address, local_port, peer_address, peer_port) in the to-be-created IKE_SA with values from the IP header. The same applies if neither NAT_PREVENTION nor NAT_DETECTION_*_IP payloads were included, or if the responder does not support NAT Traversal.

If the IKE_SA_INIT request included NAT_DETECTION_*_IP payloads but no NAT_PREVENTION payload, the situation is different since the initiator may at this point change from port 500 to 4500. In this case, the responder initializes (local_address, local_port, peer_address, peer_port) from the first IKE_AUTH request. It may also decide to perform a return routability check soon after the IKE_AUTH exchanges have been completed.

IKEv2 requires that if an IPsec endpoint discovers a NAT between it and its correspondent, it MUST send all subsequent traffic to and from port 4500. 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.

NAT_PREVENTION payloads can also be included when changing the path of IPsec SAs (see Section 2.3). TBD: add better description.

3. Payload formats

3.1 MOPO_SUPPORTED notification payload

The MOPO_SUPPORTED notification payload is included in the IKE_SA_INIT messages to indicate that the implementation supports this specification.

The Notify Message Type for MOPO_SUPPORTED is TBD-BY-IANA (16396..40959). The Protocol ID field is set to one (1), and SPI Size is set to zero. There is no data associated with this Notify type.

3.2 ADDITIONAL_ADDRESS notification payload

Both initiator and responder can include ADDITIONAL_ADDRESS payloads in the IKE_AUTH exchange and INFORMATIONAL exchange request messages; see Section 2.2 and Section 2.4 for more detailed description.

The Notify Message Type for ADDITIONAL_ADDRESS is TBD-BY-IANA (16396..40959). The Protocol ID field is set to one (1), and SPI Size is set to zero. The data associated with this Notify type is either an IPv4 address or an IPv6 address (the type is determined by payload length).

3.3 CHANGE_PATH notification payload

This payload is included in INFORMATIONAL exchange requests sent by the initiator of the IKE_SA to update addresses of the IKE_SA and IPsec SAs (see Section 2.3).

The Notify Message Type for CHANGE_PATH is TBD-BY-IANA (16396..40959). The Protocol ID field is set to one (1), and SPI Size is set to zero. There is no data associated with this Notify type.

3.4 UNACCEPTABLE_PATH notification payload

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

The Notify Message Type for UNACCEPTABLE_PATH is TBD-BY-IANA (40..8191). The Protocol ID field is set to one (1), and SPI Size is set to zero. There is no data associated with this Notify type.

3.5 COOKIE2 notification payload

This payload is included in all INFORMATIONAL exchange messages for return routability check purposes (see Section 2.6).

The data associated with this notification MUST be between 8 and 64 octets in length (inclusive), and MUST be chosen in a way that is unpredictable to the recipient. The Notify Message Type for this message is TBD-BY-IANA (16396..40959). The Protocol ID field is set to one (1), and SPI Size is set to zero.

3.6 NAT_PREVENTION notification payload

See Section 2.7 for a description of this payload.

The data associated with this notification is the SHA-1 hash [FIPS180-2] of the following data: the IP address and port from which the packet was sent, and the IP address and port to which the packet was sent. The Notify Message Type for this message is TBD-BY-IANA (16396..40959). The Protocol ID field is set to one (1), and SPI Size is set to zero.

3.7 NAT_PREVENTED notification payload

See Section 2.7 for a description of this payload.

The Notify Message Type for NAT_PREVENTED is TBD-BY-IANA (40..8191). The Protocol ID field is set to one (1), and SPI Size is set to zero. There is no data associated with this Notify type.

4. Security considerations

The main goal of this specification has been not to reduce any security offered by normal IKEv2.

(TO BE WRITTEN: more text is needed here.)

The return routability check is not inherently incompatible with NATs; as explained in Section 2.7 IKEv2/IPsec can in fact work across some kind of NATs even without NAT Traversal support. In this specification, "NAT prevention", or integrity protection for the addresses in the IP header, is a separate feature.

When NAT Traversal is supported, the peer's address may be updated automatically to allow changes in NAT mappings. The "continued return routability" feature, implemented by the COOKIE2 payload, allows verification of the new address after the change. This limits the duration of any "third party bombing" attack by off-path (relative to the victim) attackers.

5. 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 one new IKEv2 exchange, "PATH_TEST", whose value is to be allocated from the "IKEv2 Exchange Types" namespace. This exchange is described in Section 2.5.

This document also defines several new IKEv2 notification payloads whose values are to be allocated from the "IKEv2 Notification Payload Types" namespace. These notification payloads are described in Section 3.

6. Acknowledgements

Everyone in MOBIKE WG, especially Jari Arkko, Francis Dupont, Paul Hoffman, Tero Kivinen, and Hannes Tschofenig. This document also borrows many ideas and even some text from [AddrMgmt], [SMOBIKE], [Kivinen], and [Design].

7. References

7.1 Normative references

[FIPS180-2] National Institute of Standards and Technology, "Specifications for the Secure Hash Standard", Federal Information Processing Standard (FIPS) Publication 180-2, August 2002.

[IKEv2] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol", draft-ietf-ipsec-ikev2-17 (work in progress), October 2004.

[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.

7.2 Informative references

[AddrMgmt] Dupont, F., "Address Management for IKE version 2", draft-dupont-ikev2-addrmgmt-06 (work in progress), June 2004.

[Design] Kivinen, T. and H. Tschofenig, "Design of the MOBIKE protocol", draft-ietf-mobike-design-01 (work in progress), June 2004.

[Kivinen] Kivinen, T., "MOBIKE protocol", draft-kivinen-mobike-protocol-00 (work in progress), February 2004.

[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.

Author's Address

Pasi Eronen
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.