Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: May 1, 2009

P. Eronen
Nokia
J. Laganier
DOCOMO Euro-Labs
C. Madson
Cisco Systems

October 28, 2008

TOC

IPv6 Configuration in IKEv2
draft-eronen-ipsec-ikev2-ipv6-config-05

Status of this Memo

Abstract

When IKEv2 is used for remote VPN access (client to VPN gateway), the gateway assigns the client an IP address from the internal network using IKEv2 configuration payloads. The configuration payloads specified in RFC 4306 work well for IPv4, but make it difficult to use certain features of IPv6. This document describes the limitations of current IKEv2 configuration payloads for IPv6, and explores possible solutions that would allow IKEv2 to set up full-featured virtual IPv6 interfaces.

Table of Contents

1. Introduction and Problem Statement

TOC

In typical remote access VPN use (client to VPN gateway), the client needs an IP address in the network protected by the security gateway. IKEv2 includes a feature called "configuration payloads" that allows the gateway to dynamically assign a temporary address to the client [IKEv2] (Kaufman, C., “Internet Key Exchange (IKEv2) Protocol,” December 2005.).
For IPv4, the message exchange would look as follows:

```
 Client      Gateway
--------    ---------

 HDR(IKE_SA_INIT), SAi1, KEi, Ni  -->

          <--  HDR(IKE_SA_INIT), SAr1, KEr, Nr, [CERTREQ]

 HDR(IKE_AUTH),
 SK { IDi, CERT, [CERTREQ], AUTH, [IDr],
      CP(CFG_REQUEST) =
         { INTERNAL_IP4_ADDRESS(),
           INTERNAL_IP4_DNS() }, SAi2,
      TSi = (0, 0-65535, 0.0.0.0-255.255.255.255),
      TSr = (0, 0-65535, 0.0.0.0-255.255.255.255) }  -->

        <--  HDR(IKE_AUTH),
             SK { IDr, CERT, AUTH,
                  CP(CFG_REPLY) =
                     { INTERNAL_IP4_ADDRESS(192.0.2.234),
                       INTERNAL_IP4_DNS(10.11.22.33) },
                  SAr2,
                  TSi = (0, 0-65535, 192.0.2.234-192.0.2.234),
                  TSr = (0, 0-65535, 0.0.0.0-255.255.255.255) }
```

Figure 1: IPv4 configuration

The IPv4 case has been implemented by various vendors, and in general works well. IKEv2 also defines almost identical configuration payloads for IPv6:

```
 Client      Gateway
--------    ---------

 HDR(IKE_AUTH),
 SK { IDi, CERT, [CERTREQ], AUTH, [IDr],
      CP(CFG_REQUEST) =
         { INTERNAL_IP6_ADDRESS(),
           INTERNAL_IP6_DNS() }, SAi2,
      TSi = (0, 0-65535,
             0:0:0:0:0:0:0:0 -
             FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF),
      TSr = (0,
             0-65535, 0:0:0:0:0:0:0:0 -
             FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF) }  -->

        <--  HDR(IKE_AUTH),
             SK { IDr, CERT, AUTH,
                  CP(CFG_REPLY) =
                     { INTERNAL_IP6_ADDRESS(2001:DB8:0:1:2:3:4:5,
                                            64),
                       INTERNAL_IP6_DNS(2001:DB8:9:8:7:6:5:4) },
                  SAr2,
                  TSi = (0, 0-65535,
                         2001:DB8:0:1:2:3:4:5 -
                         2001:DB8:0:1:2:3:4:5),
                  TSr = (0, 0-65535,
                         0:0:0:0:0:0:0:0: -
                         FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF) }
```

Figure 2: IPv6 configuration

In other words, IPv6 is basically treated as IPv4 with larger addresses. As noted in [RFC4718] (Eronen, P. and P. Hoffman, “IKEv2 Clarifications and Implementation Guidelines,” October 2006.), this does not fully follow the "normal IPv6 way of doing things". The IPsec tunnels are not full-featured "interfaces" in the IPv6 addressing architecture [IPv6Addr] (Hinden, R. and S. Deering, “IP Version 6 Addressing Architecture,” February 2006.) sense. For example, they do not necessarily have link-local addresses, and this may complicate the use of protocols that assume them.
This document describes what exactly are the limitations of current IKEv2 configuration payloads for IPv6, and explores possible solutions that would allow IKEv2-based VPNs to set up full-featured virtual IPv6 interfaces.

TOC

2. 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] (Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels," March 1997.).

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+").

This document uses the term "virtual interface" when describing how the client uses the IPv6 address(es) assigned by the gateway. While existing IPsec documents do not use this term, it is not a new concept. In order to use the address assigned by the VPN gateway, current VPN clients already create a local "virtual interface" (as only addresses assigned to interfaces can be used, e.g., as source addresses for TCP connections). Note that this definition of "interface" is not necessarily identical with what some particular implementation calls "interface".

3. Current Limitations

TOC

This section explores the limitations of the current IPv6 configuration mechanism.

The IKEv2 specification does not always fully describe the semantics associated with configuration payloads, only their on-the-wire format. This section assumes the semantics implied by Figure 2 (IPv6 configuration). It is possible that many of the limitations described here could be solved by specifying additional semantics for these configuration payloads.

3.1. Multiple Prefixes

TOC

In Figure 2 (IPv6 configuration) only a single IPv6 address (from a single prefix) is assigned. The specification does allow the client to include multiple INTERNAL_IP6_ADDRESS attributes in its request, but the gateway cannot assign more addresses than the client requested. Multiple prefixes are useful for site renumbering, host-based site multihoming [SHIM6] (Nordmark, E. and M. Bagnulo, "Shim6: Level 3 Multihoming Shim Protocol for IPv6," February 2008.), and unique local IPv6 addresses [RFC4193] (Hinden, R. and B. Haberman, "Unique Local IPv6 Unicast Addresses," October 2005.). In all of these cases, the gateway has better information on how many different addresses (from different prefixes) the client should be assigned.

3.2. Link-Local Addresses

TOC

The IPv6 addressing architecture [IPv6Addr] (Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture," February 2006.) specifies that "IPv6 addresses of all types are assigned to interfaces, not nodes. [..] All interfaces are required to have at least one Link-Local unicast address".
Currently, the virtual interface created by IKEv2 configuration payloads does not have link-local addresses. This violates [IPv6Addr] (Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture," February 2006.) and prevents the use of protocols that require link-local addresses, such as [MLDv2] (Vida, R. and L. Costa, "Multicast Listener Discovery Version 2 (MLDv2) for IPv6," June 2004.) and [DHCPv6] (Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)," July 2003.)

3.3. Receiving Multicast Traffic

TOC

Even if MLD would work, the traffic selectors negotiated in Figure 2 (IPv6 configuration) do not allow receiving multicast traffic.

3.4. Interface Identifier Selection

TOC

In the message exchange shown in Figure 2 (IPv6 configuration), the gateway chooses the interface ID used by the client. It is also possible for the client to request a specific interface ID; the gateway then chooses the prefix part.
This approach complicates the use of Cryptographically Generates Addresses [CGA] (Aura, T., "Cryptographically Generated Addresses (CGA)," March 2006.). With CGAs, the interface ID cannot be calculated before the prefix is known. The client could first obtain a non-CGA address to determine the prefix, and then send a separate CFG_REQUEST to obtain a CGA address with the same prefix. However, this approach requires that the IKEv2 software component provides an interface to the component managing CGAs; an ugly implementation dependency that would be best avoided.
Similar concerns apply to other cases where the client has some interest in what interface ID is being used, such as Hash-Based Addresses [HBA] (Bagnulo, M., "Hash Based Addresses (HBA)," December 2007.) and privacy addresses [RFC4941] (Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6," September 2007.).
Without CGAs and HBAs, VPN clients are not able to fully use IPv6 features such as [SHIM6] (Nordmark, E. and M. Bagnulo, "Shim6: Level 3

Multihoming Shim Protocol for IPv6," February 2008.) or enhanced Mobile IPv6 route optimization [RFC4866] (Arkko, J., Vogt, C., and W. Haddad, "Enhanced Route Optimization for Mobile IPv6," May 2007.).

3.5. Sharing VPN Access

TOC

Some VPN clients may want to share the VPN connection with other devices (e.g., from a cell phone to a laptop, or vice versa) via some local area network connection (such as Wireless LAN or Bluetooth). It is to be determined how common this use case would actually be; e.g., how likely it is that security policies would allow this. Quite obviously sharing of VPN access requires more than one address (unless NAT is used). However, the current model where each address is requested separately is probably complex to integrate with a local area network that uses stateless address autoconfiguration. Thus, obtaining a whole prefix for the VPN client, and advertising that to the local link (something resembling [NDProxy] (Thaler, D., Talwar, M., and C. Patel, "Neighbor Discovery Proxies (ND Proxy)," April 2006.)) would be preferable. With DHCPv6 prefix delegation [RFC3633] (Troan, O. and R. Droms, "IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6," December 2003.), even [NDProxy] (Thaler, D., Talwar, M., and C. Patel, "Neighbor Discovery Proxies (ND Proxy)," April 2006.) and associated multi-link subnet issues would be avoided.

3.6. Additional Information

TOC

The original 3GPP standards for IPv6 assigned a single IPv6 address to each mobile phone, resembling current IKEv2 payloads. [RFC3314] (Wasserman, M., "Recommendations for IPv6 in Third Generation Partnership Project 3GPP) Standards," September 2002.) describes the problems with this approach, and caused 3GPP to change the specifications to assign unique /64 prefix(es) for each phone. If the VPN client is assigned IPv6 address(es) from prefix(es) that are shared with other VPN clients, this results in some kind of multi-link subnet. [Multilink] (Thaler, D., "Multi-Link Subnet Issues," June 2007.) describes issues associated with multi-link subnets, and recommends that they should be avoided.

4. Design Goals

TOC

TOC

4.1. Main Requirements

*A VPN client can obtain several addresses from a given prefix; the interface IDs can be selected by the client, and may depend on the prefix.

*A VPN client can be assigned multiple prefixes for use on the client-gateway link. The client does not have to know beforehand how many prefixes are needed.

*The solution should avoid periodic messages over the VPN tunnel.

*The solution should avoid Duplicate Address Detection (DAD) over the VPN tunnel.

*Multicast works. That is, the client is able to send multicast packets (tunneled to the gateway via unicast), join multicast groups using [MLDv2] (Vida, R. and L. Costa, “Multicast Listener Discovery Version 2 (MLDv2) for IPv6,” June 2004.), and receive multicast packets (tunneled from the gateway to the client via unicast).

*It should be possible to share the VPN access over a local area network connection, without requiring anything special from other hosts in the local network (beyond minimal IPv6 node requirements specified in [RFC4294] (Loughney, J., Ed., “IPv6 Node Requirements,” April 2006.)).

*Re-authentication works: the client can start a new IKE SA and continue using the same "virtual link" (with same addresses, etc.).

*Compatibility with other IPsec uses: Configuring a virtual IPv6 link should not prevent the peers from using IPsec/IKEv2 for other uses.

*Compatibility with current IPv6 configuration: Although the current IPv6 mechanism is not widely implemented, new solutions should not preclude its use (e.g., by defining incompatible semantics for the existing payloads).

*Compatibility with current IPv4 configuration: it should be possible to use the existing IPv4 configuration mechanism within the same IKE SA.

*(Optional/To be determined) When the client is also a router (to some local network), it should be able to use DHCPv6 prefix delegation [RFC3633] (Troan, O. and R. Droms, “IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6,” December 2003.) over the virtual link.

4.2. Desirable Non-Functional Properties

TOC

Note that the following desirable properties may be somewhat conflicting.

*Re-use existing mechanisms, such as [AUTOCONF] (Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration," September 2007.) and [DHCPv6] (Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)," July 2003.) as much as possible; as explained in [IPConfig] (Aboba, B., Thaler, D., and L. Andersson, "Principles of Internet Host Configuration," May 2008.), creating IKEv2-specific mechanisms should be avoided.

*Avoid the Not Invented Here (NIH) syndrome: There were several proposals how to do IP address configuration in IKEv2, and the IPsec WG chose one of them. Any significant changes should be motivated by real technical needs, not by dislike of the proposal that was chosen.

4.3. Implementation Considerations

TOC

The solution should have clean implementation dependencies. In particular, it should not require significant modifications to the core IPv6 stack (typically part of the operating system), or require the IKE implementor to re-implement parts of the IPv6 stack (to, e.g., have access or control to functionality that is currently not exposed by public interfaces of the IPv6 stack).

4.4. Non-Goals

TOC

Mobile IPv6 already defines how it interacts with IPsec/IKEv2 [RFC4877] (Devarapalli, V. and F. Dupont, "Mobile IPv6 Operation with IKEv2 and the Revised IPsec Architecture," April 2007.), and the intent of this document is not to change that interaction in any way.

TOC

5. Solution Discussion

Assigning a new IPv6 address to the client creates a new "virtual IPv6 interface", and "virtual link" between the client and the gateway. We will assume that the virtual link has the following properties:

- The link and its interfaces are created and destroyed by the IKEv2 process.

- The IPv6 addresses and prefixes are assigned to the link and its interfaces by IKEv2 messages, and are removed once they are no longer used by any IKE SA. An IKEv2 implementation may delay removal of the IPv6 addresses and prefixes for a period of time to allow upper layer protocol communications (e.g., a TCP connection) to survive an IKE SA re-authentication that would use the same addresses and prefixes.

- The link is not an IPsec SA; at any time, there can be zero or more IPsec SAs covering traffic on this link.

- The link is not a single IKE SA; to support reauthentication, it must be possible to identify the same link in another IKE SA.

- It is TBD whether a single IKE SA needs to support multiple virtual links. (Possibly not; if multiple virtual links are needed, multiple IKE_SAs could be used.)

- Not all IPsec-protected traffic between the peers is necessarily related to the virtual link (although in the simplest VPN client-to-gateway scenario it will be).

Given these assumptions and the goals described in the previous section, it seems that the most important design choices to be made are the following:

- What link/subnet model is used: in other words, the relationships between VPN clients, IPv6 subnet prefixes, and link-local traffic (especially link-local multicast).

- How information about the IPv6 prefix(es) is distributed from the gateway to the clients.

- How to ensure unique IPv6 addresses for each client, and keep forwarding state up-to-date accordingly..

- How layer 3 access control is done; in other words, where the mechanisms for preventing address spoofing by clients are placed architecturally.

Each of these is discussed next in turn.

5.1. Link Model

TOC

There are at least three main choices how to organize the relationships between VPN clients, IPv6 subnet prefixes, and link-local traffic:

* Point-to-point link model: each VPN client is assigned one or more IPv6 prefixes; these prefixes are not shared with other clients, and there is no link-local traffic between different VPN clients connected to the same gateway.

* Multi-access link model: multiple VPN clients share the same IPv6 prefix. Link-local multicast packets sent by one VPN client will be received by other VPN clients (VPN gateway will forward the packets, possibly with MLD snooping to remove unnecessary packets).

* "Router aggregation" link model: one form of "multi-link" subnet [Multilink] (Thaler, D., “Multi-Link Subnet Issues,” June 2007.) where multiple VPN clients share the same IPv6 prefix. Link-local multicast will not be received by other VPN clients.

In the multi-access link model, VPN clients who are idle (i.e., not currently sending or receiving application traffic) could receive significant amounts of multicast packets from other clients (depending on how many other clients are connected). This is especially undesirable when the clients are battery-powered; for example, a PDA which keeps the VPN connection to corporate intranet active 24/7. For this reason, we will not consider the multi-access link model in the rest of this document.

5.2. Distributing Prefix Information

TOC

Some types of addresses, such as CGAs, require knowledge about the prefix before an address can be generated. The prefix information could be distributed to clients in the following ways:

* IKEv2 messages (Configuration Payloads).

* Router Advertisement messages (sent over the IPsec tunnel).

* DHCPv6 messages (sent over the IPsec tunnel).

TOC

5.3. Unique Address Allocation

In the "multi-access" and "router aggregation" link models (where a single IPv6 prefix is shared between multiple VPN clients) mechanisms are needed to ensure that one VPN client does not use an address already used by some other client. Also, the VPN gateway has to know which client is using which addresses in order to correctly forward traffic.
The main choices seem to be the following:

- Clients receive the address(es) they are allowed to use in IKEv2 messages (Configuration Payloads). In this case, keeping track of which client is using which address is trivial.

- Clients receive the address(es) they are allowed to use in DHCPv6 messages sent over the IPsec tunnel. In case the DHCPv6 server is not integrated with the VPN gateway, the gateway may need to work as a relay agent to keep track of which client is using which address (and update its forwarding state accordingly).

- Clients can use stateless address autoconfiguration to configure addresses and perform Duplicate Address Detection (DAD). This is easy to do in multi-access link model, and can be made to work with router aggretation link model if the VPN gateway traps NS messages and spoofs NA replies. The gateway keeps track of which client is using which address (and updates its forwarding state accordingly) by trapping these NS/NA messages.

In the point-to-point link model, the client can simply use any address from the prefix, and the VPN gateway only needs to know which client is using which prefix in order to forward packets correctly.

5.4. Layer 3 Access Control

TOC

It is almost always desirable to prevent one VPN client from sending packets with a source address that is used by another VPN client. In order to correctly forward packets destined to clients, the VPN gateway obviously has to know which client is using which address; the question is therefore where, architecturally, the mechanisms for ingress filtering are placed.

- Layer 3 access control enforced by IPsec SAD/SPD: the addresses/ prefixes assigned to a VPN client are reflected in the traffic selectors used in IPsec Security Association and Security Policy Database entries, as negotiated in IKEv2.

*The ingress filtering capability could be placed outside IPsec; the traffic selectors in SAD/SPD entries would cover traffic that would be dropped later by ingress filtering.

The former approach is used by the current IPv4 solution.

5.5. Other Considerations

TOC

VPN gateway state

In some combinations of design choices, the amount of state information required in the VPN gateway depends not only on the number of clients, but also on the number of addresses used by one client. With privacy addresses and potentially some uses of Cryptographically Generated Addresses (CGAs), a single client could have a large number of different addresses (especially if different privacy addresses are used with different destinations).

Virtual link identifier

Reauthentication requires a way to uniquely identify the virtual link when a second IKE SA is created. Some possible alternatives are the IKE SPIs of the IKE SA where the virtual link was "created" (assuming we can't have multiple virtual links within the same IKE SA), a new identifier assigned when the link is created, or any unique prefix or address that remains assigned to the link for its entire lifetime. Currently, Section 6 (Solution Sketch) proposes that the gateway assigns a new IKEv2 Link ID when the link is created. The client treats the Link ID as an opaque octet string; the gateway uses it to identify relevant local state when reauthentication is done.

Note that the link is not uniquely identified by the IKE peer identities (because IDi is often a user identity that can be used on multiple hosts at the same time), or the outer IP addresses of the peers (due to NAT Traversal and [MOBIKE] (Eronen, P., “IKEv2 Mobility and Multihoming Protocol (MOBIKE),” June 2006.)).

Prefix lifetime

Prefixes could remain valid either for the lifetime of the IKE SA, until explicitly cancelled, or for an explicitly specified time. Currently, Section 6 (Solution Sketch) proposes that prefixes remain valid for the lifetime of the IKE SA (and its continuations via rekeying, but not reauthentication). If necessary, the VPN gateway can thus add or remove prefixes by triggering reauthentication. It is assumed that adding or removing prefixes is a relatively rare situation, and thus this draft does not currently specify more complex solutions (such as explicit prefix lifetimes, or use of CFG_SET/CFG_ACK).

Compatibility with other IPsec uses

> Compatibility with other IPsec uses probably requires that when a CHILD_SA is created, both peers can determine whether the CHILD_SA applies to the virtual interface (at the end of the virtual link), or the real interfaces IKEv2 messages are being sent over. This is required to select the correct SPD to be used for traffic selector narrowing and SA authorization in general.
>
> One straight-forward solution would be to add an extra payload to CREATE_CHILD_SA requests, containing the virtual link identifier. Requests not containing this payload would refer to the real link (over which IKEv2 messages are being sent).
>
> Another solution is to require that the peer requesting a CHILD_SA proposes traffic selectors that identify the link. For example, if TSi includes the peer's "outer" IP address, it's probably related to the real interface, not the virtual one. Or if TSi includes any of the prefixes assigned by the gateway (or the link-local or multicast prefix), it is probably related to the virtual interface.
>
> These heuristics can work in many situations, but have proved inadequate in the context of IPv6-in-IPv4 tunnels [RFC4891] (Graveman, R., Parthasarathy, M., Savola, P., and H. Tschofenig, “Using IPsec to Secure IPv6-in-IPv4 Tunnels,” May 2007.) and Provider Provisioned VPNs [VLINK] (Duffy, M., “Framework for IPsec Protected Virtual Links for PPVPNs,” October 2002.) [RFC3884] (Touch, J., Eggert, L., and Y. Wang, “Use of IPsec Transport Mode for Dynamic Routing,” September 2004.), and Mobile IPv6 [RFC4877] (Devarapalli, V. and F. Dupont, “Mobile IPv6 Operation with IKEv2 and the Revised IPsec Architecture,” April 2007.). Thus, currently Section 6 (Solution Sketch) proposes including the virtual link identifier in all CREATE_CHILD_SA requests that apply to the virtual interface.

Example about other IPsec uses:

> If a VPN gateway receives a CREATE_CHILD_SA request associated with a physical Ethernet interface, requesting SA for (TSi=FE80::something, dst=*), it would typically reject the request (or in other words, narrow it to an empty set) because it doesn't have SPD/PAD entries that would allow joe.user@example.com to request such CHILD_SAs.
>
> (However, it might have SPD/PAD entries that would allow "neighboring-router.example.com" to create such SAs, for protecting e.g. some routing protocol that uses link-local addresses.)
>
> However, the virtual interface created when joe.user@example.com

> authenticated and sent INTERNAL_IP6_LINK would have a different SPD/PAD, which would allow joe.user@example.com to create this SA.

6. Solution Sketch

TOC

This solution is basically a combination of (1) point-to-point link model, (2) prefix information distributed in IKEv2 messages, and (3) access control enforced by IPsec SAD/SPD.
(Second preliminary version, based on discussions with Tero Kivinen.)

6.1. Initial Exchanges

TOC

1) During IKE_AUTH, the client sends a new configuration attribute, INTERNAL_IP6_LINK, which requests a virtual link to be configured. The attribute contains the client's interface ID for link-local address (other addresses may use other interface IDs). Typically, the client would also ask for DHCPv6 server address; this is used only for configuration, not address assignment.
To handle backward compatibility between a client that supports the extended address configuration mechanism hereby specified and a VPN gateway that does not, this specification RECOMMENDS that the VPN client includes as well the INTERNAL_IP6_ADDRESS configuration attribute to allow graceful fallback to the existing address configuration mechanism specified in the IKEv2 specification [IKEv2] (Kaufman, C., “Internet Key Exchange (IKEv2) Protocol,” December 2005.), unless it knows for sure that the VPN gateway supports the extended mechanism hereby specified (e.g., via configuration.)

```
        CP(CFG_REQUEST) =
           { INTERNAL_IP6_LINK(Client's Link-Local Interface ID)
             INTERNAL_IP6_ADDRESS()
             INTERNAL_IP6_DHCP() }
        TSi = (0, 0-65535, 0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
        TSr = (0, 0-65535, 0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)  -->
```

To handle backward compatibility between a VPN gateway that supports the extended address configuration mechanism hereby specified and a client that does not, if the client has not sent the INTERNAL_IP6_LINK configuration attribute the VPN gateway MUST NOT include the INTERNAL_IP6_LINK configuration attribute in its reply and should fallback to the address configuration mechanism specified in the IKEv2

specification [IKEv2] (Kaufman, C., “Internet Key Exchange (IKEv2) Protocol,” December 2005.).
If the client has sent the INTERNAL_IP6_LINK configuration attribute, the VPN gateway SHOULD ignore any INTERNAL_IP6_ADDRESS configuration attribute present in the request.
The VPN gateway MUST choose for itself a link-local interface identifier different than the client's one, i.e., accept the link-local interface identifier proposed by the client. In case the VPN gateway cannot accept the link-local interface identifier the client proposed, the VPN gateway MUST fail the IPv6 address assignment by including a NOTIFY payload with the INTERNAL_ADDRESS_FAILURE message, i.e., the IKE_SA can be created but no CHILD_SA will be created.
The VPN Gateway then replies with an INTERNAL_IP6_LINK configuration attribute that contains the IKEv2 Link ID (which will be used for reauthentication and CREATE_CHILD_SA messages), the client's link local interface identier, and zero or more INTERNAL_IP6_PREFIX attributes. The traffic selectors proposed by the initiator are also narrowed to contain only the assigned prefixes, and the client link-local address formed from the well-known link-local subnet prefix and the client link-local interface identifier.

```
        CP(CFG_REPLY) =
           { INTERNAL_IP6_LINK(Client's Link-Local Interface ID,
                               IKEv2 Link ID)
             INTERNAL_IP6_PREFIX(Prefix1/64),
             [INTERNAL_IP6_PREFIX(Prefix2/64),...],
             [INTERNAL_IP6_DHCP(Address) ]
        TSi = ((0, 0-65535,
                FE80::<Client's Interface ID> -
                FE80::<Client's Interface ID>)
               (0, 0-65535,
                Prefix1::0 -
                Prefix1::FFFF:FFFF:FFFF:FFFF),
               [(0, 0-65535,
                 Prefix2::0 -
                 Prefix2::FFFF:FFFF:FFFF:FFFF), ...])
        TSr = (0, 0-65535,
               0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
```

At this point, the client can configure 1) its link-local address from the well-known link-local subnet prefix (FE80::/64) and the assigned client's link-local interface identifier, and 2) other non-link-local unicast addresses from the assigned prefixes and any proper interface identifier [IPv6Addr] (Hinden, R. and S. Deering, “IP Version 6 Addressing Architecture,” February 2006.). The VPN gateway MUST NOT simultaneously assign the same prefixes to any other client, and MUST NOT itself configure addresses from these prefixes. Thus, the client does not have to perform Duplicate Address Detection (DAD). (This

approach is based on [IPv6PPP] (Varada, S., Haskins, D., and E. Allen, "IP Version 6 over PPP," September 2007.).)
The prefixes remain valid through the lifetime of the IKE SA (and its continuations via rekeying). If the VPN gateway needs to remove a prefix it has previously assigned, or assign a new prefix, it can do so by triggering reauthentication.
2) The client also contacts the DHCPv6 server. This is the RECOMMENDED way to obtain additional configuration parameters (such as the DNS server), as it allows easier extensibility and more options (such as the domain search list for DNS).

6.2. Reauthentication

TOC

When the client performs reauthentication (and wants to continue using the same "virtual link"), it includes the IKEv2 Link ID given by the gateway in the INTERNAL_IP6_LINK attribute.

```
        CP(CFG_REQUEST) =
           { INTERNAL_IP6_LINK(Client's Link Local Interface ID,
                               IKEv2 Link ID)
             INTERNAL_IP6_DHCP() }
        TSi = (0, 0-65535, 0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
        TSr = (0, 0-65535, 0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)  -->
```

The gateway uses the Link ID to look up relevant local state, verifies that the authenticated peer identity associated with the link is correct, and continues the handshake as usual.

6.3. Creating CHILD_SAs

TOC

As described in the previous section, both peers need to be able to determine whether a CHILD_SA applies to the virtual interfaces, or the real interfaces IKEv2 messages are being sent over.
Currently, this document proposes using an explicit indication instead of relying on heuristics: the peers MUST include a LINK_ID notification, containing the IKEv2 Link ID, in all CREATE_CHILD_SA requests, including rekeys, that are related to the virtual link. The LINK_ID notification is not included in the CREATE_CHILD_SA response, or when doing IKE_SA rekeying.

TOC

6.4. Multicast

(The details of multicast use are to-be-determined.)
One way would be to create an SA for receiving multicast packets:

```
        TSi = (0, 0-65535,
               FF00:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
        TSr = (0, 0-65535,
               0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF) -->

        <--
        TSi = (0, 0-65535,
               FF00:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
        TSr = (0, 0-65535,
               0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
```

...and then use MLD as usual.

6.5. Relationship to Neighbor Discovery

TOC

Neighbor Discovery [IPv6ND] (Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)," September 2007.) specifies the following mechanisms:
Router Discovery, Prefix Discovery, Parameter Discovery,and Address Autoconfiguration are not used, as the necessary functionality is implemented in IKEv2 layer.
Address Resolution, Next-hop Determination, and Redirect are not used, as the virtual link does not have link-local addresses, and is a point-to-point link.
Neighbor Unreachability Detection could be used, but is a bit redundant given IKEv2 Dead Peer Detection.
Duplicate Address Detection is not needed, because this is a point-to-point link, where the VPN gateway does not assign any addresses from the global unicast prefixes, and link-local interface identifier is negotiated separately.

6.6. Relationship to Existing IKEv2 Payloads

TOC

The mechanism described in this document is not intended to be used at the same time as the existing INTERNAL_IP6_ADDRESS attribute. For compatibility with gateways implementing only INTERNAL_IP6_ADDRESS, the VPN client MAY include attributes for both mechanisms in CFG_REQUEST.

The capabilities and preferences of the VPN gateway will then determine which is used.
All other attributes except INTERNAL_IP6_ADDRESS (and INTENAL_ADDRESS_EXPIRY) from [IKEv2] (Kaufman, C., "Internet Key Exchange (IKEv2) Protocol," December 2005.) remain valid, including the somewhat confusingly named INTERNAL_IP6_SUBNET (see Section 6.3 of [RFC4718] (Eronen, P. and P. Hoffman, "IKEv2 Clarifications and Implementation Guidelines," October 2006.) for discussion).

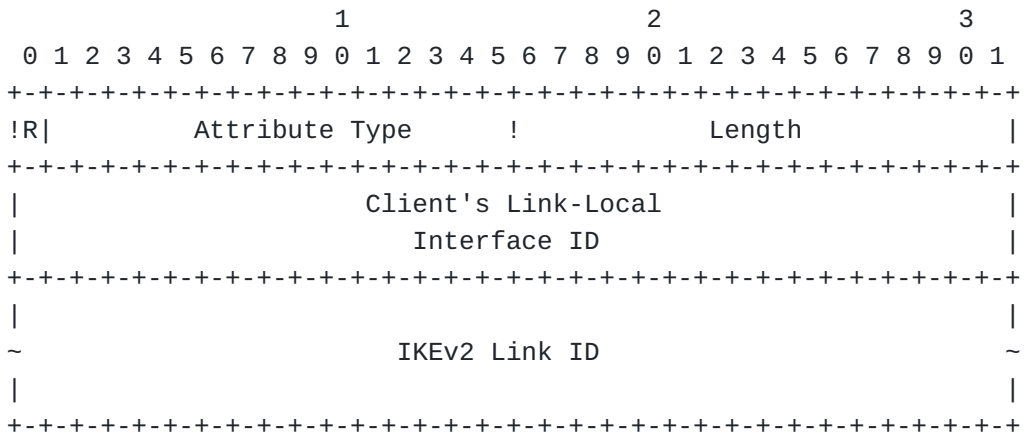
7. Payload Formats

TOC

7.1. INTERNAL_IP6_LINK Configuration Attribute

TOC

The INTERNAL_IP6_LINK configuration attribute is formatted as follows:



*Reserved (1 bit) - See [IKEv2] (Kaufman, C., "Internet Key Exchange (IKEv2) Protocol," December 2005.).

*Attribute Type (15 bits) - INTERNAL_IP6_LINK (TBD1).

*Length (2 octets) - Length in octets of the Value field (Client's Link-Local Interface ID and IKEv2 Link ID); 8 or more.

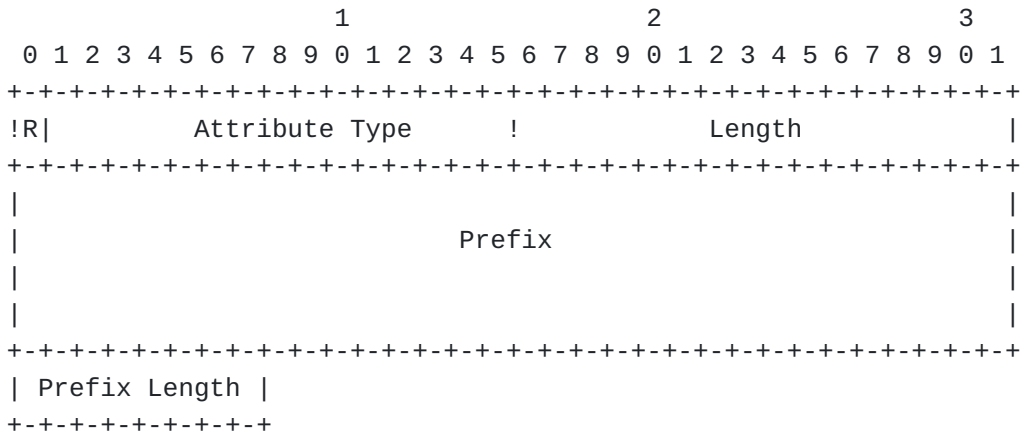
*Link-Local Interface ID (8 octets) - The Interface ID used for link-local address (by the party that sent this attribute).

*IKEv2 Link ID (variable length) - The link ID (may be empty when the client does not yet know the link ID).

TOC

7.2. INTERNAL_IP6_PREFIX Configuration Attribute

The INTERNAL_IP6_PREFIX configuration attribute is formatted as follows:



*Reserved (1 bit) - See [IKEv2] (Kaufman, C., "Internet Key Exchange (IKEv2) Protocol," December 2005.).

*Attribute Type (15 bits) - INTERNAL_IP6_PREFIX (TBD2).

*Length (2 octets) - Length in octets of the Value field; in this case, 17.

*Prefix (16 octets) - An IPv6 prefix assigned to the virtual link. The low order bits of the prefix field which are not part of the prefix MUST be set to zero by the sender and MUST be ignored by the receiver.

*Prefix Length (1 octets) - The length of the prefix in bits; usually 64.

7.3. LINK_ID Notify Payload

TOC

The LINK_ID notification is included in CREATE_CHILD_SA requests to indicate that the SA being created is related to the virtual link. If this notification is not included, the CREATE_CHILD_SA requests is related to the physical interface.
The Notify Message Type for LINK_ID is TBD3. The Protocol ID and SPI Size fields are set to zero. The data associated with this notification is the IKEv2 Link ID returned in the INTERNAL_IP6_LINK_ID configuration attribute.

8. IANA Considerations

TOC

This document defines two new IKEv2 configuration attributes, whose values are to be allocated (have been allocated) from the "IKEv2 Configuration Payload Attribute Types" namespace [IKEv2] (Kaufman, C., "Internet Key Exchange (IKEv2) Protocol," December 2005.):

Value	Attribute Type	Multi-Valued	Length	Reference
TBD1	INTERNAL_IP6_LINK	NO	8 or more	[this doc]
TBD2	INTERNAL_IP6_PREFIX	YES	17 octets	[this doc]

This document also defines one new IKEv2 notification, whose value is to be allocated (has been allocated) from the "IKEv2 Notify Message Types - Status Types" namespace [IKEv2] (Kaufman, C., "Internet Key Exchange (IKEv2) Protocol," December 2005.):

Value	Notify Messages - Status Types	Reference
TBD3	LINK_ID	[this doc]

This document does not create any new namespaces to be maintained by IANA.

9. Security Considerations

TOC

To be written. (The security consideration should be pretty much the same as for current configuration payloads.)
Assigning each client a unique prefix makes using randomized interface identifiers [RFC4941] (Narten, T., Draves, R., and S. Krishnan, "Privacy Extensions for Stateless Address Autoconfiguration in IPv6," September 2007.) ineffective from privacy point of view: the client is still uniquely identified by the prefix. In some environments, it may be preferable to assign a VPN client the same prefixes each time a VPN connection is established; other environments may prefer assigning a different prefix every time for privacy reasons. (This is basically a similar trade-off as in Mobile IPv6 -- using the same Home Address forever is simpler than changing it often, but has privacy implications.)

10. Acknowledgments

TOC

The author would like to thank Patrick Irwin, Tero Kivinen, Julien Laganier, Chinh Nguyen, Mohan Parthasarathy, Yaron Sheffer, Hemant

Singh, Dave Thaler, Yinghzhe Wu, and Fan Zhao for their valuable comments.

Many of the challenges associated with IPsec-protected "virtual interfaces" have been identified before: for example, in the context of protecting IPv6-in-IPv4 tunnels with IPsec [RFC4891] (Graveman, R., Parthasarathy, M., Savola, P., and H. Tschofenig, "Using IPsec to Secure IPv6-in-IPv4 Tunnels," May 2007.), Provider Provisioned VPNs [VLINK] (Duffy, M., "Framework for IPsec Protected Virtual Links for PPVPNs," October 2002.) [RFC3884] (Touch, J., Eggert, L., and Y. Wang, "Use of IPsec Transport Mode for Dynamic Routing," September 2004.), and Mobile IPv6 [RFC4877] (Devarapalli, V. and F. Dupont, "Mobile IPv6 Operation with IKEv2 and the Revised IPsec Architecture," April 2007.). Some of the limitations of assigning a single IPv6 address were identified in [RFC3314] (Wasserman, M., "Recommendations for IPv6 in Third Generation Partnership Project 3GPP) Standards," September 2002.).

11. References

TOC

11.1. Normative References

TOC

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

[IPv6Addr] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture," RFC 4291, February 2006.

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

11.2. Informative References

TOC

[AUTOCONF] Thomson, S., Narten, T., and T. Jinmei, "IPv6 Stateless Address Autoconfiguration," RFC 4862, September 2007.

[CGA] Aura, T., "Cryptographically Generated Addresses (CGA)," RFC 3972, March 2006.

[DHCPv6] Droms, R., Bound, J., Volz, B., Lemon, T., Perkins, C., and M. Carney, "Dynamic Host Configuration Protocol for IPv6 (DHCPv6)," RFC 3315, July 2003.

[HBA] Bagnulo, M., "Hash Based Addresses (HBA)," draft-ietf-shim6-hba-05 (work in progress), December 2007.

[IPConfig] Aboba, B., Thaler, D., and L. Andersson, "Principles of Internet Host Configuration," draft-iab-ip-config-04 (work in progress), May 2008.

[IPv6ND] Narten, T., Nordmark, E., Simpson, W., and H. Soliman, "Neighbor Discovery for IP version 6 (IPv6)," RFC 4861, September 2007.

[IPv6PPP] Varada, S., Haskins, D., and E. Allen, “IP Version 6 over PPP,” RFC 5072, September 2007.

[MLDv2] Vida, R. and L. Costa, “Multicast Listener Discovery Version 2 (MLDv2) for IPv6,” RFC 3810, June 2004.

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

[Multilink] Thaler, D., “Multi-Link Subnet Issues,” RFC 4903, June 2007.

[NDProxy] Thaler, D., Talwar, M., and C. Patel, “Neighbor Discovery Proxies (ND Proxy),” RFC 4389, April 2006.

[RFC3314] Wasserman, M., “Recommendations for IPv6 in Third Generation Partnership Project 3GPP) Standards,” RFC 3314, September 2002.

[RFC3456] Patel, B., Aboba, B., Kelly, S., and V. Gupta, “Dynamic Host Configuration Protocol (DHCPv4) Configuration of IPsec Tunnel Mode,” RFC 3456, January 2003.

[RFC3633] Troan, O. and R. Droms, “IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6,” RFC 3633, December 2003.

[RFC3884] Touch, J., Eggert, L., and Y. Wang, “Use of IPsec Transport Mode for Dynamic Routing,” RFC 3884, September 2004.

[RFC4193] Hinden, R. and B. Haberman, “Unique Local IPv6 Unicast Addresses,” RFC 4193, October 2005.

[RFC4294] Loughney, J., Ed., “IPv6 Node Requirements,” RFC 4294, April 2006.

[RFC4718] Eronen, P. and P. Hoffman, “IKEv2 Clarifications and Implementation Guidelines,” RFC 4718, October 2006.

[RFC4866] Arkko, J., Vogt, C., and W. Haddad, “Enhanced Route Optimization for Mobile IPv6,” RFC 4866, May 2007.

[RFC4877] Devarapalli, V. and F. Dupont, “Mobile IPv6 Operation with IKEv2 and the Revised IPsec Architecture,” RFC 4877, April 2007.

[RFC4891] Graveman, R., Parthasarathy, M., Savola, P., and H. Tschofenig, “Using IPsec to Secure IPv6-in-IPv4 Tunnels,” RFC 4891, May 2007.

[RFC4941] Narten, T., Draves, R., and S. Krishnan, “Privacy Extensions for Stateless Address Autoconfiguration in IPv6,” RFC 4941, September 2007.

[SHIM6] Nordmark, E. and M. Bagnulo, “Shim6: Level 3 Multihoming Shim Protocol for IPv6,” draft-ietf-shim6-proto-10 (work in progress), February 2008.

[VLINK] Duffy, M., “Framework for IPsec Protected Virtual Links for PPVPNs,” draft-duffy-ppvpn-ipsec-vlink-00 (work in progress), October 2002.

TOC

Appendix A. Alternative Solution Sketches

A.1. Version -00 Sketch

TOC

The -00 version of this draft contained the following solution sketch, which is basically a combination of (1) point-to-point link model, (2) prefix information distributed in Neighbor Advertisements, and (3) access control enforced outside IPsec.

1) During IKE_AUTH, client sends a new configuration attribute, INTERNAL_IP6_LINK_ID, which requests a virtual link to be created. The attribute contains the client's interface ID for link-local address (other addresses may use other interface IDs).

```
        CP(CFG_REQUEST) =
           { INTERNAL_IP6_LINK_ID(Link-Local Interface ID) }
        TSi = (0, 0-65535, 0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
        TSr = (0, 0-65535, 0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)  -->
```

The VPN gateway replies with its own link-local interface ID (which MUST be different from the client's) and an IKEv2 Link ID (which will be used for reauthentication).

```
        CP(CFG_REPLY) =
           { INTERNAL_IP6_LINK_ID(Link-Local Interface ID, IKEv2 Link ID) }
        TSi = (0, 0-65535, 0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
        TSr = (0, 0-65535, 0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
```

At this point, both peers configure the virtual interface with the link-local addresses.

2) The next step is IPv6 stateless address autoconfiguration; that is, Router Solicitation and Router Advertisement messages sent over the IPsec SA.

```
       ESP(Router Solicitation:
           src=:
           dst=FF02:0:0:0:0:0:0:2)  -->

       <-- ESP(Router Advertisement:
               src=FE80::<Gateway's Interface ID>
               dst=FF02:0:0:0:0:0:0:1,
               Prefix1, [Prefix2...])
```

After receiving the Router Advertisement, the client can configure unicast addresses from the advertised prefixes, using any interface ID. The VPN gateway MUST NOT simultaneously assign the same prefixes to any other client, and MUST NOT itself configure addresses from these prefixes. Thus, the client does not have to perform Duplicate Address Detection (DAD).
3) Reauthentication works basically the same way as in Section 6.2 (Reauthentication); the client includes the IKEv2 Link ID in the INTERNAL_IP6_LINK_ID attribute.
4) Creating and rekeying IPsec SAs works basically the same way as in Section 6.3 (Creating CHILD_SAs); the client includes the IKEv2 Link ID in those CHILD_SA requests that are related to the virtual link.
Comments: This was changed in -01 draft based on feedback from VPN vendors: while the solution looks nice on paper, it is claimed to be unneccessarily complex to implement when the IKE implementation and IPv6 stack are from different companies. Furthermore, enforcing access control outside IPsec is a significant architectural change compared to current IPv4 solutions.

A.2. Router Aggregation Sketch #1

TOC

The following solution was sketched during the IETF 70 meeting in Vancouver together with Hemant Singh. It combines the (1) router aggregation link model, (2) prefix information distributed in IKEv2 messages, (3) unique address allocation with stateless address autoconfiguration (with VPN gateway trapping NS messages and spoofing NA replies), and (4) access control enforced (partly) outside IPsec.
1) During IKE_AUTH, the client sends a new configuration attribute, INTERNAL_IP6_LINK_ID, which requests a virtual link to be created. The attribute contains the client's interface ID for link-local address (other addresses may use other interface IDs).

```
CP(CFG_REQUEST) =
   { INTERNAL_IP6_LINK_ID(Link-Local Interface ID) }
TSi = (0, 0-65535, 0:0:0:0:0:0:0:0 -
       FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
TSr = (0, 0-65535, 0:0:0:0:0:0:0:0 -
       FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)  -->
```

The VPN gateway replies with its own link-local interface ID (which MUST be different from the client's), an IKEv2 Link ID (which will be used for reauthentication and CREATE_CHILD_SA messages), and zero or more INTERNAL_IP6_PREFIX attributes. The traffic selectors proposed by the initiator are also narrowed to contain only the assigned prefixes (and the link-local prefix).

```
CP(CFG_REPLY) =
   { INTERNAL_IP6_LINK_ID(Link-Local Interface ID, IKEv2 Link ID),
     INTERNAL_IP6_PREFIX(Prefix1/64),
     [INTERNAL_IP6_PREFIX(Prefix2/64),...],
     INTERNAL_IP6_DHCP(Address) ]
TSi = ((0, 0-65535,
        FE80::<Client's Interface ID> -
        FE80::<Client's Interface ID>)
       (0, 0-65535,
        Prefix1::0 -
        Prefix1::FFFF:FFFF:FFFF:FFFF),
       [(0, 0-65535,
         Prefix2::0 -
         Prefix2::FFFF:FFFF:FFFF:FFFF), ...])
TSr = (0, 0-65535,
       0:0:0:0:0:0:0:0 -
       FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
```

2) The client now configures tentative unicast addresses from the prefixes given by the gateway, and performs Duplicate Address Detection (DAD) for them.
The Neighbor Solicitation messages are processed by the VPN gateway: if the target address is already in use by some other VPN client, the gateway replies with a Neighbor Advertisement. If the target address is not already in use, the VPN gateway notes that it is now being used by this client, and updates its forwarding state accordingly.
Comments: The main disadvantages of this solution are non-standard processing of NS messages (which are used to update the gateway's forwarding state), and performing access control partly outside IPsec.

A.3. Router Aggregation Sketch #2

TOC

This is basically similar to the version -00 sketch described with above, but uses router aggregation link model. In other words, it combines (1) router aggregation link model, (2) prefix information distributed in Neighbor Advertisements, (3) unique address allocation with stateless address autoconfiguration (with VPN gateway trapping NS messages and spoofing NA replies), and (4) access control enforced outside IPsec.
1) During IKE_AUTH, client sends a new configuration attribute, INTERNAL_IP6_LINK_ID, which requests a virtual link to be created. The attribute contains the client's interface ID for link-local address (other addresses may use other interface IDs).

```
        CP(CFG_REQUEST) =
           { INTERNAL_IP6_LINK_ID(Link-Local Interface ID) }
        TSi = (0, 0-65535, 0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
        TSr = (0, 0-65535, 0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)  -->
```

The VPN gateway replies with its own link-local interface ID (which MUST be different from the client's) and an IKEv2 Link ID (which will be used for reauthentication).

```
        CP(CFG_REPLY) =
           { INTERNAL_IP6_LINK_ID(Link-Local Interface ID, IKEv2 Link ID) }
        TSi = (0, 0-65535, 0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
        TSr = (0, 0-65535, 0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
```

At this point, both peers configure the virtual interface with the link-local addresses.

2) The next step is IPv6 stateless address autoconfiguration; that is, Router Solicitation and Router Advertisement messages sent over the IPsec SA.

```
       ESP(Router Solicitation:
           src=:
           dst=FF02:0:0:0:0:0:0:2)  -->

       <-- ESP(Router Advertisement:
               src=FE80::<Gateway's Interface ID>
               dst=FF02:0:0:0:0:0:0:1,
               Prefix1, [Prefix2...])
```

3) The client now configures tentative unicast addresses from the prefixes given by the gateway, and performs Duplicate Address Detection (DAD) for them.

The Neighbor Solicitation messages are processed by the VPN gateway: if the target address is already in use by some other VPN client, the gateway replies with a Neighbor Advertisement. If the target address is not already in use, the VPN gateway notes that it is now being used by this client, and updates its forwarding state accordingly.

Comments: The main disadvantages of this solution are non-standard processing of NS messages (which are used to update the gateway's forwarding state), and performing access control outside IPsec.

TOC

A.4. IPv4-like Sketch

This sketch resembles the current IPv4 configuration payloads, and it combines (1) router aggregation link model, (2) prefix information distributed in IKEv2 messages, (3) unique address allocation with IKEv2 messages, and (4) access control enforced by IPsec SAD/SPD.
1) During IKE_AUTH, the client sends a new configuration attribute, INTERNAL_IP6_LINK_ID, which requests a virtual link to be created. The attribute contains the client's interface ID for link-local address (other addresses may use other interface IDs).

```
        CP(CFG_REQUEST) =
           { INTERNAL_IP6_LINK_ID(Link-Local Interface ID) }
        TSi = (0, 0-65535,
               0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
        TSr = (0, 0-65535,
               0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)  -->
```

The VPN gateway replies with its own link-local interface ID (which MUST be different from the client's), an IKEv2 Link ID (which will be used for reauthentication and CREATE_CHILD_SA messages), and zero or more INTERNAL_IP6_ADDRESS2 attributes. Each attribute contains one address from a particular prefix.

```
        CP(CFG_REPLY) =
           { INTERNAL_IP6_LINK_ID(Link-Local Interface ID, IKEv2 Link ID),
             INTERNAL_IP6_ADDRESS2(Prefix1+Client's Interface ID1),
             [INTERNAL_IP6_ADDRESS2(Prefix2+Client's Interface ID2),...],
        TSi = ((0, 0-65535,
                FE80::<Client's Link-Local Interface ID> -
                FE80::<Client's Link-Local Interface ID>)
               (0, 0-65535,
                Prefix1::<Client's Interface ID1> -
                Prefix1::<Client's Interface ID1>),
               [(0, 0-65535,
                 Prefix2::<Client's Interface ID2> -
                 Prefix2::<Client's Interface ID2>), ...])
        TSr = (0, 0-65535,
               0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
```

Since the VPN gateway keeps track of address uniqueness, there is no need to perform Duplicate Address Detection.
2) If the client wants additional addresses later (for example, with specific interface ID), it requests them in a separate CREATE_CHILD_SA exchange. For example:

```
        CP(CFG_REQUEST) =
           { INTERNAL_IP6_ADDRESS2(Prefix1+Client's Interface ID3) }
        TSi = (0, 0-65535,
               Prefix1::0 -
               Prefix1::FFFF:FFFF:FFFF:FFFF>),
        TSr = (0, 0-65535,
               0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)  -->
```

If the requested address is not currently in use by some other client, the VPN gateway simply returns the same address, and traffic selectors narrowed appropriately.

```
        CP(CFG_REQUEST) =
           { INTERNAL_IP6_ADDRESS2(Prefix1+Client's Interface ID3) }
        TSi = ((0, 0-65535,
                Prefix1::<Client's Interface ID3> -
                Prefix1::<Client's Interface ID3>),
        TSr = (0, 0-65535,
               0:0:0:0:0:0:0:0 -
               FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF:FFFF)
```

Comments: The main advantage of this solution is that it's quite close to the current IPv4 way of doing things. By adding explicit link creation (with Link ID for reauthentication/SPD selection, and link-local addresses), and slightly changing the semantics (and also name) of INTERNAL_IP6_ADDRESS attribute (can return more attributes than was asked), we get much of the needed functionality.
The biggest disadvantages are probably potentially complex implementation dependency for interface ID selection (see Section 3.4 (Interface Identifier Selection)), and the multi-link subnet model.

A.5. Sketch Based on RFC 3456

TOC

For completeness: a solution modeled after [RFC3456] (Patel, B., Aboba, B., Kelly, S., and V. Gupta, “Dynamic Host Configuration Protocol (DHCPv4) Configuration of IPsec Tunnel Mode,” January 2003.) would combine (1) router aggregation link model, (2) prefix information distribution and unique address allocation with DHCPv6, and (3) access control enforced by IPsec SAD/SPD.

Authors' Addresses

TOC

Pasi Eronen
Nokia Research Center
P.O. Box 407

FIN-00045 Nokia Group
Finland
Email: pasi.eronen@nokia.com

Julien Laganier
DOCOMO Communications Laboratories Europe GmbH
Landsberger Strasse 312
Munich D-80687
Germany
Phone: +49 89 56824 231
Email: julien.laganier.IETF@googlemail.com

Cheryl Madson
Cisco Systems, Inc.
510 MacCarthy Drive
Milpitas, CA
USA
Email: cmadson@cisco.com

Full Copyright Statement

TOC

Intellectual Property