

IPSECME WG
Internet-Draft
Intended status: Informational
Expires: October 25, 2015

A. Dodd-Noble
S. Gundavelli
Cisco
J. Korhonen
F. Baboescu
Broadcom Corporation
B. Weis
Cisco
April 23, 2015

3GPP IMS Option for IKEv2
draft-gundavelli-ipsecme-3gpp-ims-options-05.txt

Abstract

This document defines two new configuration attributes for Internet Key Exchange Protocol version 2 (IKEv2). These attributes can be used for carrying the IPv4 address and IPv6 address of the Proxy-Call Session Control Function (P-CSCF). When an IPsec gateway delivers these attributes to an IPsec client, the IPsec client can obtain the IPv4 and/or IPv6 address of the P-CSCF server located in the 3GPP network.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on October 25, 2015.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Conventions and Terminology	4
2.1. Conventions	4
2.2. Terminology	4
3. P_CSCF_IP4_ADDRESS Configuration Attribute	5
4. P_CSCF_IP6_ADDRESS Configuration Attribute	5
5. Example Scenario	6
6. IANA Considerations	7
7. Security Considerations	8
8. Acknowledgements	8
9. References	8
9.1. Normative References	8
9.2. Informative References	8
Authors' Addresses	9

1. Introduction

The Third Generation Partnership Project (3GPP) S2b reference point [TS23402], specified by the 3GPP system architecture defines a mechanism for allowing a mobile node (MN) attached in an untrusted non-3GPP IP Access Network to securely connect to a 3GPP network and access IP services. In this scenario, the mobile node establishes an IPsec ESP tunnel [RFC4303] to the security gateway called evolved packet data gateway (ePDG) and which in turn establishes a Proxy Mobile IPv6 (PMIPv6) [RFC5213] or GPRS Tunneling Protocol (GTP) [TS23402] tunnel to the packet data gateway (PGW) [TS23402] where the mobile node's session is anchored.

The below figure shows the interworking option for non-3GPP access over an untrusted-access network. The mobile access gateway (MAG) and the local mobility anchor (LMA) functions are defined in [RFC5213]. The ePDG and PGW functions are defined in [TS23402]. IPsec ESP tunnel is between the MN and the ePDG and PMIP or GTP tunnel between the ePDG and the PGW.

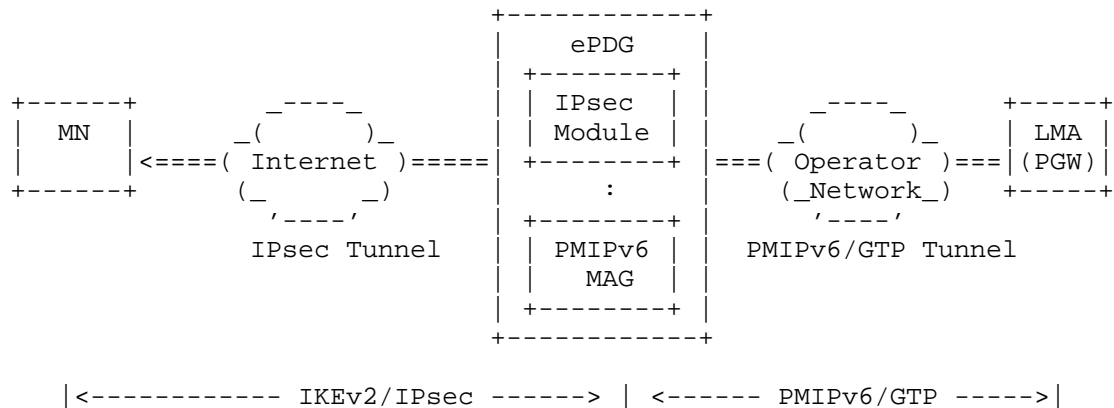


Figure 1: Exchange of IPv4 Traffic Offload Selectors

A mobile node in this scenario may potentially need to access the IP Multimedia Subsystem (IMS) services in the 3GPP network [TS23228] and [TS24229]. Currently, there are no attributes in IKEv2 [RFC7296] that can be used for carrying these information elements. In the absence of these attributes the mobile node needs to be statically configured with this information and this is proving to be an operational challenge. Any other approaches such as using DNS, or DHCP for discovering these functions would result in obtaining

configuration in the access network and not in the home network. Given that the above referenced 3GPP interface is primarily for allowing the mobile node to connect to the 3GPP network through an untrusted-access network, the access network may not have any relation with the home network provider and may be unable to deliver the mobile node's home network configuration.

This specification therefore defines two new IKEv2 attributes [RFC7296] that allows an IPsec gateway to provide the IPv4 and/or IPv6 address of the P-CSCF server. These attributes can be exchanged by IKEv2 peers as part of the configuration payload exchange. The attributes follow the configuration attribute format defined in Section 3.15.1 of [RFC7296]. Furthermore, providing the P-CSCF server address(es) in IKEv2 as standard attribute(s) enables clients to directly access IMS services behind a VPN gateway without going through the 3GPP specific interfaces.

2. Conventions and Terminology

2.1. Conventions

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 RFC 2119 [RFC2119].

2.2. Terminology

All the IKEv2 related terms used in this document are to be interpreted as defined in [RFC7296] and [RFC5739]. All the mobility related terms are to be interpreted as defined in [RFC5213] and [RFC5844]. Additionally, this document uses the following terms:

Proxy-Call Session Control Function (P-CSCF)

The P-CSCF is the entry point to the 3GPP IMS (IP Multimedia Subsystem) and serves as the SIP outbound proxy for the mobile node. The mobile node performs SIP registration to 3GPP IMS and initiates SIP sessions via a P-CSCF.

Evolved Packet Data Gateway (ePDG)

Its is a security gateway defined by the 3GPP system architecture. The protocol interfaces it supports include IKEv2 [RFC7296].

3. P_CSCF_IP4_ADDRESS Configuration Attribute

The P_CSCF_IP4_ADDRESS configuration attribute is formatted as follows:

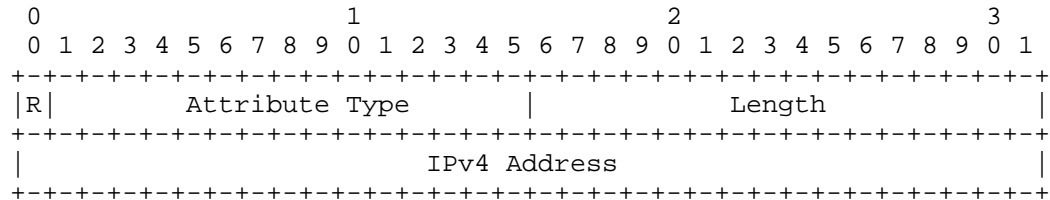


Figure 2: IPv4 Address of P-CSCF

Reserved (1 bit)

Refer to IKEv2 specification

Attribute Type (15 bits)

<IANA-1>

Length (2 octets)

Length of the IPv4 address field that follows. Possible values are (0) and (4). A value of (4) indicates the size of the 4-octet IPv4 address that follows. A value of (0) indicates that its a empty attribute with zero-length IPv4 address field, primarily used as a request indicator.

IPv4 Address (4 octets)

An IPv4 address of the P-CSCF server.

The P_CSCF_IP4_ADDRESS configuration attribute provides an IPv4 address of a P-CSCF server within the network. If an instance of an empty P_CSCF_IP4_ADDRESS attribute with zero-length IPv4 Address field is included by mobile node, the responder MAY respond with zero, one or more P_CSCF_IP4_ADDRESS attributes. If several P_CSCF_IP4_ADDRESS attributes are provided in one IKEv2 message, there is no implied order among the P_CSCF_IP4_ADDRESS attributes. However, a system architecture using this specification may be able to enforce some order at both the peers.

4. P_CSCF_IP6_ADDRESS Configuration Attribute

The P_CSCF_IP6_ADDRESS configuration attribute is formatted as follows:

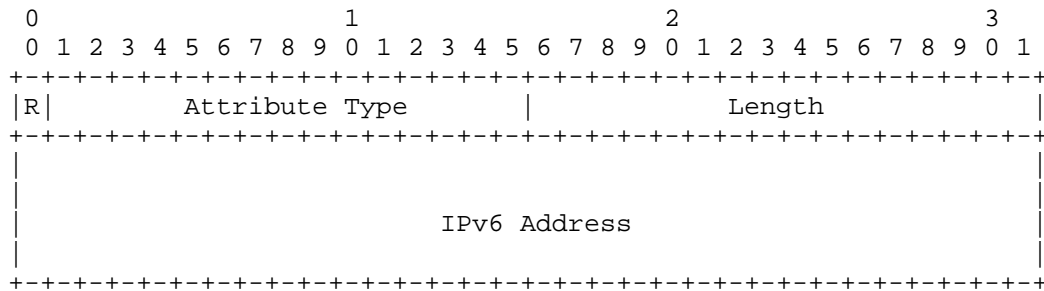


Figure 3: IPv6 Address of P-CSCF

Reserved (1 bit)

Refer to IKEv2 specification

Attribute Type (15 bits)

<IANA-1>

Length (2 octets)

Length of the IPv6 address field that follows. Possible values are (0) and (16). A value is (16) indicates the size of the 16-octet IPv6 address that follows. A value of (0) indicates that its a empty attribute with zero-length IPv6 address field, primarily used as a request indicator.

IPv6 Address (16 octets)

An IPv6 address of the P-CSCF server.

The P_CSCF_IP6_ADDRESS configuration attribute provides an IPv6 address of a P-CSCF server within the network. If an instance of an empty P_CSCF_IP6_ADDRESS attribute with zero-length IPv6 Address field is included by mobile node, the responder MAY respond with zero, one or more P_CSCF_IP6_ADDRESS attributes. If several P_CSCF_IP6_ADDRESS attributes are provided in one IKEv2 message, there is no implied order among the P_CSCF_IP6_ADDRESS attributes. However, a system architecture using this specification may be able to enforce some order at both the peers.

5. Example Scenario

The mobile node MAY request the IP address of an P-CSCF server as shown below.

```

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(),
      P_CSCF_IP4_ADDRESS() }, 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),
        P_CSCF_IP4_ADDRESS(192.0.2.1),
        P_CSCF_IP4_ADDRESS(192.0.2.4),
        INTERNAL_IP4_DNS(198.51.100.33) },
    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 4: P-CSCF Attribute Exchange

6. IANA Considerations

This document requires the following two IANA actions.

- o Action-1: This specification defines a new IKEv2 attribute for carrying the IPv4 address of P-CSCF server. This attribute is defined in Section 3. The Type value for this Attribute needs to be assigned from the IKEv2 Configuration Payload Attribute Types namespace defined in [RFC7296].
- o Action-2: This specification defines a new IKEv2 attribute for carrying the IPv6 address of P-CSCF server. This attribute is defined in Section 4. The Type value for this Attribute needs to be assigned from the IKEv2 Configuration Payload Attribute Types namespace defined in [RFC7296].

7. Security Considerations

This document is an extension to IKEv2 [RFC7296] and therefore it inherits all the security properties of IKEv2.

The two new IKEv2 attributes defined in this specification are for carrying the IPv4 and IPv6 address of the P-CSCF server. These attributes can be exchanged by IKE peers as part of the configuration payload and the currently defined IKEv2 security framework provides the needed integrity and privacy protection for these attributes. Therefore this specification does not introduce any new security vulnerabilities.

8. Acknowledgements

The Authors would like to specially thank Tero Kivinen for the detailed reviews. Authors would also like to thank Vojislav Vucetic, Heather Sze, Sebastian Speicher, Maulik Vaidya, Ivo Sedlacek, Pierrick Siete and Hui Deng for all the discussions related to this topic.

9. References

9.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.
- [RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, October 2014.

9.2. Informative References

- [RFC5213] Gundavelli, S., Leung, K., Devarapalli, V., Chowdhury, K., and B. Patil, "Proxy Mobile IPv6", RFC 5213, August 2008.
- [RFC5739] Eronen, P., Laganier, J., and C. Madson, "IPv6 Configuration in Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5739, February 2010.
- [RFC5844] Wakikawa, R. and S. Gundavelli, "IPv4 Support for Proxy Mobile IPv6", RFC 5844, May 2010.

- [TS23228] 3GPP, "Service requirements for the Internet Protocol (IP) multimedia core network subsystem (IMS); Stage 1", 2014.
- [TS23402] 3GPP, "Architecture enhancements for non-3GPP accesses", 2014.
- [TS24229] 3GPP, "IP multimedia call control protocol based on Session Initiation Protocol (SIP) and Session Description Protocol (SDP); Stage 3", 2014.

Authors' Addresses

Aeneas Noble
Cisco
30 International Pl
TEWKSBURY, MASSACHUSETTS 95134
USA

Email: noblea@cisco.com

Sri Gundavelli
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA

Email: sgundave@cisco.com

Jouni Korhonen
Broadcom Corporation
Porkkalankatu 24
Helsinki FIN-00180
Finland

Email: jouni.nospam@gmail.com

Florin Baboescu
Broadcom Corporation
100 Mathilda Place
Sunnyvale, CA 94086
USA

Email: baboescu@broadcom.com>

Brian Weis
Cisco
170 West Tasman Drive
San Jose, CA 95134
USA

Email: bew@cisco.com

IPsecME Working Group
Internet-Draft
Intended status: Informational
Expires: January 17, 2014

V. Manral
HP
S. Hanna
Juniper
July 16, 2013

Auto Discovery VPN Problem Statement and Requirements
draft-ietf-ipsecme-ad-vpn-problem-09

Abstract

This document describes the problem of enabling a large number of systems to communicate directly using IPsec to protect the traffic between them. It then expands on the requirements, for such a solution.

Manual configuration of all possible tunnels is too cumbersome in many such cases. In other cases the IP address of endpoints change or the endpoints may be behind NAT gateways, making static configuration impossible. The Auto Discovery VPN solution will address these requirements.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 17, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of

publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Terminology	3
1.2. Conventions Used in This Document	4
2. Use Cases	4
2.1. Endpoint-to-Endpoint VPN Use Case	4
2.2. Gateway-to-Gateway VPN Use Case	5
2.3. Endpoint-to-Gateway VPN Use Case	5
3. Inadequacy of Existing Solutions	6
3.1. Exhaustive Configuration	6
3.2. Star Topology	6
3.3. Proprietary Approaches	7
4. Requirements	7
4.1. Gateway and Endpoint Requirements	7
5. Security Considerations	10
6. IANA Considerations	11
7. Acknowledgements	11
8. Normative References	11
Authors' Addresses	11

1. Introduction

IPsec [RFC4301] is used in several different cases, including tunnel-mode site-to-site VPNs and Remote Access VPNs. Both tunneling modes for IPsec gateways and host-to-host transport mode are supported in this document.

The subject of this document is the problem presented by large scale deployments of IPsec and the requirements on a solution to address the problem. These may be a large collection of VPN gateways connecting various sites, a large number of remote endpoints connecting to a number of gateways or to each other, or a mix of the two. The gateways and endpoints may belong to a single administrative domain or several domains with a trust relationship.

Section 4.4 of RFC 4301 describes the major IPsec databases needed for IPsec processing. It requires an extensive configuration for each tunnel, so manually configuring a system of many gateways and endpoints becomes infeasible and inflexible.

The difficulty is that a lot of configuration mentioned in RFC 4301 is required to set up a Security Association. IKE implementations need to know the identity and credentials of all possible peer systems, as well as the addresses of hosts and/or networks behind them. A simplified mechanism for dynamically establishing point-to-point tunnels is needed. Section 2 contains several use cases that motivate this effort.

1.1. Terminology

ADVPN - Auto Discovery Virtual Private Network (ADVPN) is VPN solution that enables a large number of systems to communicate directly, with minimal configuration and operator intervention using IPsec to protect communication between them.

Endpoint - A device that implements IPsec for its own traffic but does not act as a gateway.

Gateway - A network device that implements IPsec to protect traffic flowing through the device.

Point-to-Point - Communication between two parties without active participation (e.g. encryption or decryption) by any other parties.

Hub - The central point in a star topology/ dynamic full mesh topology, or one of the central points in the full mesh style VPN, i.e. gateway where multiple other hubs or spokes connect to. The hubs usually forward traffic coming from encrypted links to other encrypted links, i.e. there are no devices connected to it in clear.

Spoke - The endpoint in a star topology/ dynamic full mesh topology, or gateway which forwards traffic from multiple cleartext devices to other hubs or spokes, and some of those other devices are connected to it in clear (i.e. it encrypts data coming from cleartext devices and forwards it to the ADVPN).

ADVPN Peer - any member of an ADVPN including gateways, endpoints, hub or spoke.

Star topology - This is the topology where there is direct connectivity only between the hub and spoke and communication between the 2 spokes happens through the hub.

Allied and Federated Environments - Environments where we have multiple different organizations that have close association and need to connect to each other.

Full Mesh topology - This is the topology where there is a direct connectivity between every Spoke to every other Spoke directly, without the traffic between the spokes having to be redirected through an intermediate hub device.

Dynamic Full Mesh topology - This is the topology where direct connections exist in a hub and spoke manner, but dynamic connections are created/ removed between the spokes on a need basis.

Security Association (SA) - Defined in [RFC4301].

1.2. Conventions 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 [RFC2119].

2. Use Cases

This section presents the key use cases for large-scale point-to-point VPN.

In all of these use cases, the participants (endpoints and gateways) may be from a single organization (administrative domain) or from multiple organizations with an established trust relationship. When multiple organizations are involved, products from multiple vendors are employed so open standards are needed to provide interoperability. Establishing communications between participants with no established trust relationship is out of scope for this effort.

2.1. Endpoint-to-Endpoint VPN Use Case

Two endpoints wish to communicate securely via a point-to-point Security Association (SA).

The need for secure endpoint-to-endpoint communications is often driven by a need to employ high-bandwidth, low -latency local connectivity instead of using slow, expensive links to remote gateways. For example, two users in close proximity may wish to place a direct, secure video or voice call without needing to send the call through remote gateways, which would add latency to the call, consume precious remote bandwidth, and increase overall costs. Such a use case also enables connectivity when both users are behind NAT gateways. Such a use case ought to allow for seamless connectivity even as endpoints roam, even if they are moving out from behind a NAT gateway, from behind one NAT gateway to behind another, or from a standalone position to behind a NAT gateway.

In a star topology, when two endpoints communicate they need a mechanism for authentication, such that they do not expose themselves to impersonation by the other spoke endpoint.

2.2. Gateway-to-Gateway VPN Use Case

A typical Enterprise traffic model follows a star topology, with the gateways connecting to each other using IPsec tunnels.

However for voice and other rich media traffic that requires a lot of bandwidth or is performance sensitive, the traffic tromboning (taking a suboptimal path) to the hub can create traffic bottlenecks on the hub and can lead to an increase in cost. A fully meshed solution would make best use of the available network capacity and performance but the deployment of a fully meshed solution involves considerable configuration, especially when a large number of nodes are involved. It is for this purpose spoke-to-spoke tunnels are dynamically created and torn-down. For the reasons of cost and manual error reduction, it is desired that there be minimal configuration on each gateway.

The solution ought to work in cases where the endpoints are in different administrative domains, albeit, ones that have an existing trust relationship (for example two organisations who are collaborating on a project, they may wish to join their networks, whilst retaining independent control over configuration). It is highly desirable that the solution works for the star, full mesh as well as dynamic full mesh topology.

The solution ought to also address the case where gateways use dynamic IP addresses.

Additionally, the routing implications of gateway-to-gateway communication need to be addressed. In the simple case, selectors provide sufficient information for a gateway to forward traffic appropriately. In other cases, additional tunneling (e.g., Generic Routing Encapsulation - GRE) and routing (e.g., Open Shortest Path First - OSPF) protocols are run over IPsec tunnels, and the configuration impact on those protocols needs to be considered. There is also the case when Layer-3 Virtual Private Networks (L3VPNs) operate over IPsec Tunnels.

When two gateways communicate, they need to use a mechanism for authentication, such that they do not expose themselves to the risk of impersonation by the other entities.

2.3. Endpoint-to-Gateway VPN Use Case

A mobile endpoint ought to be able to use the most efficient gateway as it roams in the internet.

A mobile user roaming on the Internet may connect to a gateway, which because of roaming is no longer the most efficient gateway to use (reasons could be cost/ efficiency/ latency or some other factor). The mobile user ought to be able to discover and then connect to the current most efficient gateway in a seamless way without having to bring down the connection.

3. Inadequacy of Existing Solutions

Several solutions exist for the problems described above. However, none of these solutions is adequate, as described here.

3.1. Exhaustive Configuration

One simple solution is to configure all gateways and endpoints in advance with all the information needed to determine which gateway or endpoint is optimal and to establish an SA with that gateway or endpoint. However, this solution does not scale in a large network with hundreds of thousands of gateways and endpoints, especially when multiple administrative domains are involved and things are rapidly changing (e.g. mobile endpoints). Such a solution is also limited by the smallest endpoint/ gateway, as the same exhaustive configuration is to be applied on all endpoints/ gateways. A more dynamic, secure and scalable system for establishing SAs between gateways is needed.

3.2. Star Topology

The most common way to address a part of this this problem today is to use what has been termed a "star topology". In this case one or a few gateways are defined as "hub gateways", while the rest of the systems (whether endpoints or gateways) are defined as "spokes". The spokes never connect to other spokes. They only open tunnels with the hub gateways. Also for a large number of gateways in one administrative domain, one gateway may be defined as the hub, and the rest of the gateways and remote access clients connect only to that gateway.

This solution however is complicated by the case when the spokes use dynamic IP addresses and DNS with dynamic updates needs to be used. It is also desired that there is minimal to no configuration on the hub as the number of spokes increases and new spokes are added and deleted randomly.

Another problem with the star topology is that it creates a high load on the hub gateways as well as on the connection between the spokes

and the hub. This load is both in processing power and in network bandwidth. A single packet in the hub-and-spoke scenario can be encrypted and decrypted multiple times. It would be much preferable if these gateways and clients could initiate tunnels between them, bypassing the hub gateways. Additionally, the path bandwidth to these hub gateways may be lower than that of the path between the spokes. For example, two remote access users may be in the same building with high-speed wifi (for example, at an IETF meeting). Channeling their conversation through the hub gateways of their respective employers seems extremely wasteful, as well as having lower bandwidth.

The challenge is to build large scale, IPsec-protected networks that can dynamically change with minimal administrative overhead.

3.3. Proprietary Approaches

Several vendors offer proprietary solutions to these problems. However, these solutions offer no interoperability between equipment from one vendor and another. This means that they are generally restricted to use within one organization, and it is harder to move off such solutions as the features are not standardized. Besides multiple organizations cannot be expected to all choose the same equipment vendor.

4. Requirements

This section defines the requirements, on which the solution will be based.

4.1. Gateway and Endpoint Requirements

1. For any network topology (star, full mesh and dynamic full mesh), when a new gateway or endpoint is added, removed, or changed, configuration changes are minimized as follows. Adding or removing a spoke in the topology **MUST NOT** require configuration changes to the hubs other than where the spoke was connected to and **SHOULD NOT** require configuration changes to the hub the spoke was connected to. The changes also **MUST NOT** require configuration changes in other spokes.

Specifically, when evaluating potential proposals, we will compare them by looking at how many endpoints or gateways must be reconfigured when a new gateway or endpoint is added, removed, or changed and how substantial this reconfiguration is, besides the amount of static configuration required.

This requirement is driven by use cases 2.1 and 2.2 and by the scaling limitations pointed out in section 3.1.

2. ADVPN peers MUST allow IPsec Tunnels to be setup with other members of the ADVPN without any configuration changes, even when peer addresses get updated every time the device comes up. This implies that SPD entries or other configuration based on peer IP address will need to be automatically updated, avoided, or handled in some manner to avoid a need to manually update policy whenever an address changes.

3. In many cases additional tunneling protocols (e.g. GRE) or Routing protocols (e.g. OSPF) are run over the IPsec tunnels. Gateways MUST allow for the operation of tunneling and Routing protocols operating over spoke-to-spoke IPsec Tunnels with minimal or no, configuration impact. The ADVPN solution SHOULD NOT increase the amount of information required to configure protocols running over IPsec tunnels.

4. In the full mesh and dynamic full mesh topology, Spokes MUST allow for direct communication with other spoke gateways and endpoints. In the star topology mode, direct communication between spokes MUST be disallowed.

This requirement is driven by use cases 2.1 and 2.2 and by the limitations of a star topology pointed out in section 3.2.

5. Any of the ADVPN Peers MUST NOT have a way to get the long term authentication credentials for any other ADVPN Peers. The compromise of an Endpoint MUST NOT affect the security of communications between other ADVPN Peers. The compromise of a Gateway SHOULD NOT affect the security of the communications between ADVPN Peers not associated with that Gateway.

This requirement is driven by use case 2.1. ADVPN Peers (especially Spokes) become compromised fairly often. The compromise of one ADVPN Peer SHOULD NOT affect the security of other unrelated ADVPN Peers.

6. Gateways SHOULD allow for seamless handoff of sessions in case endpoints are roaming, even if they cross policy boundaries. This would mean the data traffic is minimally affected even as the handoff happens. External factors like firewall, NAT boxes that will be part of the overall solution when DVPN is deployed will not be considered part of this solution.

Such endpoint roaming may affect not only the endpoint-to-endpoint SA but also the relationship between the endpoints and gateways (such as when an endpoint roams to a new network that is handled by a different gateway).

This requirement is driven by use case 2.1. Today's endpoints are mobile and transition often between different networks (from 4G to WiFi and among various WiFi networks).

7. Gateways SHOULD allow for easy handoff of a session to another gateway, to optimize latency, bandwidth, load balancing, availability, or other factors, based on policy.

This ability to migrate traffic from one gateway to another applies regardless of whether the gateways in question are hubs or spokes. It even applies in the case where a gateway (hub or spoke) moves in the network, as may happen with a vehicle-based network.

This requirement is driven by use case 2.3.

8. Gateways and endpoints MUST have the capability to participate in an ADVPN even when they are located behind NAT boxes. However, in some cases they may be deployed in such a way that they will not be fully reachable behind a NAT box. It is especially difficult to handle cases where the Hub is behind a NAT box. Where the two endpoints are both behind separate NATs, communication between these spokes SHOULD be supported using workarounds such as port forwarding by the NAT or detecting when two spokes are behind uncooperative NATs and using a hub in that case.

This requirement is driven by use cases 2.1 and 2.2. Endpoints are often behind NATs and gateways sometimes are. IPsec SHOULD continue to work seamlessly regardless, using ADVPN techniques whenever possible and providing graceful fallback to hub and spoke techniques as needed.

9. Changes such as establishing a new IPsec SA SHOULD be reportable and manageable. However, creating a MIB or other management technique is not within scope for this effort.

This requirement is driven by manageability concerns for all the use cases, especially use case 2.2. As IPsec networks become more dynamic, management tools become more essential.

10. To support allied and federated environments, endpoints and gateways from different organizations SHOULD be able to connect to each other.

This requirement is driven by demand for all the use cases in federated and allied environments.

11. The administrator of the ADVPN SHOULD allow for the configuration of a Star, Full mesh or a partial full mesh topology, based on which tunnels are allowed to be setup.

This requirement is driven by demand for all the use cases in federated and allied environments.

12. The ADVPN solution SHOULD be able to scale for multicast traffic.

This requirement is driven by the use case 2.2, where the amount of rich media multicast traffic is increasing.

13. The ADVPN solution SHOULD allow for easy monitoring, logging and reporting of the dynamic changes, to help for trouble shooting such environments.

This requirement is driven by demand for all the use cases in federated and allied environments.

14. There is also the case when L3VPNs operate over IPsec Tunnels, for example Provider Edge (PE) based VPN's. An ADVPN MUST support L3VPN as an application protected by the IPsec Tunnels.

This requirement is driven by demand for all the use cases in federated and allied environments.

15. There ADVPN solution SHOULD allow the enforcement of per peer QoS in both the Star as well as the Full Mesh topology.

This requirement is driven by demand for all the use cases in federated and allied environments.

16. There ADVPN solution SHOULD take care of not letting the Hub to be a single point of failure.

This requirement is driven by demand for all the use cases in federated and allied environments.

5. Security Considerations

This is a Problem statement and requirement document for the ADVPN solution, and in itself does not introduce any new security concerns. The solution to the problems presented in this draft may involve dynamic updates to databases defined by RFC 4301, such as the

Security Policy Database (SPD) or the Peer Authorization Database (PAD).

RFC 4301 is silent about the way these databases are populated, and it is implied that these databases are static and pre-configured by a human. Allowing dynamic updates to these databases must be thought out carefully, because it allows the protocol to alter the security policy that the IPsec endpoints implement.

One obvious attack to watch out for is stealing traffic to a particular site. The IP address for `www.example.com` is `192.0.2.10`. If we add an entry to an IPsec endpoint's SPD that says that traffic to `192.0.2.10` is protected through peer Gw-Mallory, then this allows Gw-Mallory to either pretend to be `www.example.com` or to proxy and read all traffic to that site. Updates to this database requires a clear trust model.

Hubs can be a single point of failure which can cause loss of connectivity of the entire system, that can be a big security issue. Any ADVPN solution design should take care of these concerns.

6. IANA Considerations

No actions are required from IANA for this informational document.

7. Acknowledgements

Many people have contributed to the development of this problem statement and many more will probably do so before we are done with it. While we cannot thank all contributors, some have played an especially prominent role. Yoav Nir, Yaron Sheffer, Jorge Coronel Mendoza, Chris Ulliot, and John Veizades wrote the document upon which this draft was based. Geoffrey Huang, Toby Mao, Suresh Melam, Praveen Sathyanarayan, Andreas Steffen, Brian Weis, and Lou Berger provided essential input.

8. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

Authors' Addresses

Vishwas Manral
Hewlett-Packard Co.
19111 Pruneridge Ave.
Cupertino, CA 95113
USA

Email: vishwas.manral@hp.com

Steve Hanna
Juniper Networks, Inc.
1194 N. Mathilda Ave.
Sunnyvale, CA 94089
USA

Email: shanna@juniper.net

ipsecme
Internet-Draft
Updates: 5996 (if approved)
Intended status: Standards Track
Expires: December 6, 2013

Y. Sheffer
Porticor
S. Fluhrer
Cisco
June 4, 2013

Additional Diffie-Hellman Tests for IKEv2
draft-ietf-ipsecme-dh-checks-05

Abstract

This document adds a small number of mandatory tests required for the secure operation of IKEv2 with elliptic curve groups. No change is required to IKE implementations that use modular exponential groups, other than a few rarely used so-called DSA groups. This document updates the IKEv2 protocol, RFC 5996.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 6, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as

described in the Simplified BSD License.

Table of Contents

1.	Introduction	3
1.1.	Conventions used in this document	3
2.	Group Membership Tests	3
2.1.	Sophie Germain Prime MODP Groups	3
2.2.	MODP Groups with Small Subgroups	4
2.3.	Elliptic Curve Groups	4
2.4.	Transition	5
2.5.	Protocol Behavior	5
3.	Side-Channel Attacks	6
4.	Security Considerations	6
4.1.	DH Key Reuse and Multiple Peers	7
4.2.	DH Key Reuse: Variants	7
4.3.	Groups not covered by this RFC	7
4.4.	Behavior Upon Test Failure	8
5.	IANA Considerations	8
6.	Acknowledgements	9
7.	References	9
7.1.	Normative References	9
7.2.	Informative References	9
Appendix A.	Appendix: Change Log	10
A.1.	-05	10
A.2.	-04	10
A.3.	-03	10
A.4.	-02	10
A.5.	-01	10
A.6.	-00	11
	Authors' Addresses	11

1. Introduction

IKEv2 [RFC5996] consists of the establishment of a shared secret using the Diffie-Hellman (DH) protocol, followed by authentication of the two peers. Existing implementations typically use modular exponential (MODP) DH groups, such as those defined in [RFC3526].

IKEv2 does not require that any tests be performed by a peer receiving a public Diffie-Hellman key from the other peer. This is fine for the common case of MODP groups. For other DH groups, when peers reuse DH values across multiple IKE sessions, the lack of tests by the recipient results in a potential vulnerability (see Section 4.1 for more details). In particular, this is true for Elliptic Curve (EC) groups whose use is becoming ever more popular. This document defines such tests for several types of DH groups.

In addition, this document describes another potential attack related to reuse of DH keys: a timing attack. This additional material is taken from [RFC2412].

This document updates [RFC5996] by adding security requirements that apply to many of the protocol's implementations.

1.1. Conventions 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 [RFC2119].

2. Group Membership Tests

This section describes the tests that need to be performed by IKE peers receiving a Key Exchange (KE) payload. The tests are RECOMMENDED for all implementations, but only REQUIRED for those that reuse DH private keys (as defined in [RFC5996], Sec. 2.12). The tests apply to the recipient of a KE payload, and describe how it should check the received payload. They are listed here according to the DH group being used.

2.1. Sophie Germain Prime MODP Groups

These are currently the most commonly used groups; all these groups have the property that $(p-1)/2$ is also prime; this section applies to any such MODP group. Each recipient MUST verify that the peer's public value r is in the legal range ($1 < r < p-1$). According to [Menezes], Sec 2.2, even with this check there remains the possibility of leaking a single bit of the secret exponent when DH

keys are reused; this amount of leakage is insignificant.

See Section 5 for the specific groups covered by this section.

2.2. MODP Groups with Small Subgroups

[RFC5114] defines modular exponential groups with small subgroups; these are modular exponential groups with comparatively small subgroups, and all have $(p-1)/2$ composite. Sec. 2.1 of [Menezes] describes some informational leakage from a small subgroup attack on these groups, if the DH private value is reused.

This leakage can be prevented if the recipient performs a test on the peer's public value, however this test is expensive (approximately as expensive as what reusing DH private values saves). In addition, the NIST standard [NIST-800-56A] requires that test (see section 5.6.2.4), hence anyone needing to conform to that standard will need to implement the test anyway.

Because of the above, the IKE implementation MUST choose between one of the following two options:

- o It MUST check both that the peer's public value is in range ($1 < r < p-1$) and that $r^q = 1 \bmod p$ (where q is the size of the subgroup, as listed in the RFC). DH private values MAY then be reused. This option is appropriate if conformance to [NIST-800-56A] is required.
- o It MUST NOT reuse DH private values (that is, the DH private value for each DH exchange MUST be generated from a fresh output of a cryptographically secure random number generator), and it MUST check that the peer's public value is in range ($1 < r < p-1$). This option is more appropriate if conformance to [NIST-800-56A] is not required.

See Section 5 for the specific groups covered by this section.

2.3. Elliptic Curve Groups

IKEv2 can be used with elliptic curve groups defined over a field $GF(p)$ [RFC5903] [RFC5114]. According to [Menezes], Sec. 2.3, there is some informational leakage possible. A receiving peer MUST check that its peer's public value is valid; that is, the x and y parameters from the peer's public value satisfy the curve equation, $y^2 = x^3 + ax + b \bmod p$ (where for groups 19, 20, 21, $a = -3 \bmod p$), and all other values of a , b and p for the group are listed in the RFC).

We note that an additional check to ensure that the public value is

not the point at infinity is not needed, because IKE (in Sec. 7 of [RFC5903]) does not allow for encoding this value.

See Section 5 for the specific groups covered by this section.

2.4. Transition

Existing implementations of IKEv2 with ECDH groups may be modified to include the tests described in the current document, even if they do not reuse DH keys. The tests can be considered as sanity checks, and will prevent the code having to handle inputs that it may not have been designed to handle.

ECDH implementations that do reuse DH keys MUST be enhanced to include the above tests.

2.5. Protocol Behavior

The recipient of a DH public key that fails one of the above tests must assume that the sender is either truly malicious or else it has a bug in its implementation. The behavior defined below attempts to balance resistance to attackers that are trying to disrupt the IKE exchange, against the need to help a badly implemented peer by providing useful error indications.

If this error happens during the IKE_SA_INIT exchange, then the recipient MUST drop the message that contains an invalid KE payload, and MUST NOT use that message when creating the IKE SA.

If the implementation implements the DoS-resistant behavior proposed in Sec. 2.4 of [RFC5996], it may simply ignore the erroneous request or response message, and continue waiting for a later message containing a legitimate KE payload.

If DoS-resistant behavior is not implemented, and the invalid KE payload was in the IKE_SA_INIT request, the implementation MAY send an INVALID_SYNTAX error notification back, and remove the in-progress IKE SA; if the invalid KE payload was in the IKE_SA_INIT response, then the implementation MAY simply delete the half created IKE SA, and re-initiate the exchange.

If the invalid KE payload is received during the CREATE_CHILD_SA exchange (or any other exchange after the IKE SA has been established) and the invalid KE payload is in the request message, the Responder MUST reply with an INVALID_SYNTAX error notification and drop the IKE SA. If the invalid KE payload is in a response, the Initiator getting this reply MUST immediately delete the IKE SA by sending an IKE SA Delete notification as a new exchange. In this

case the sender evidently has an implementation bug, and dropping the IKE SA makes it easier to detect.

3. Side-Channel Attacks

In addition to the small-subgroup attack, there is also a potential timing attack on IKE peers when they are reusing Diffie-Hellman secret values. This is a side-channel attack, which means that it may or may not be a vulnerability in certain cases, depending on implementation details and the threat model.

The remainder of this section is quoted from [RFC2412], Sec. 5, with a few minor clarifications. This attack still applies to IKEv2 implementations, and both to MODP groups and ECDH groups. We also note that more efficient countermeasures are available for EC groups represented in projective form, but these are outside the scope of the current document.

Timing attacks that are capable of recovering the exponent value used in Diffie-Hellman calculations have been described by Paul Kocher [Kocher]. In order to nullify the attack, implementors must take pains to obscure the sequence of operations involved in carrying out modular exponentiations.

One potential method to foil these timing attacks is to use a "blinding factor". In this method, a group element, r , is chosen at random, and its multiplicative inverse modulo p is computed, which we'll call r_{inv} . r_{inv} can be computed by the Extended Euclidean Method, using r and p as inputs. When an exponent x is chosen, the value r_{inv}^x is also calculated. Then, when calculating $(g^y)^x$, the implementation will calculate this sequence:

$$\begin{aligned} A &= r * g^y \\ B &= A^x = (r * g^y)^x = (r^x)(g^{xy}) \\ C &= B * r_{\text{inv}}^x = (r^x)(r^{-1*x})(g^{xy}) = g^{xy} \end{aligned}$$

The blinding factor is only necessary if the exponent x is used more than 100 times (estimate by Richard Schroepel).

4. Security Considerations

This entire document is concerned with the IKEv2 security protocol and the need to harden it in some cases.

4.1. DH Key Reuse and Multiple Peers

This section describes one variant of the attack prevented by the tests defined above.

Suppose that IKE peer Alice maintains IKE security associations with peers Bob and Eve. Alice uses the same secret ECDH key for both SAs, which is allowed with some restrictions. If Alice does not implement these tests, Eve will be able to send a malformed public key, which would allow her to efficiently determine Alice's private key (as described in Sec. 2 of [Menezes]). Since the key is shared, Eve will be able to obtain Alice's shared IKE SA key with Bob.

4.2. DH Key Reuse: Variants

Private DH keys can be reused in different ways, with subtly different security implications. For example:

1. DH keys are reused for multiple connections (IKE SAs) to the same peer, and for connections to different peers.
2. DH keys are reused for multiple connections to the same peer (e.g. when the peer is identified by its IP address) but not for different peers.
3. DH keys are reused only when they had not been used to complete an exchange, e.g. when the peer replies with an INVALID_KEY_PAYLOAD notification.

Both the small subgroup attack and the timing attack described in this document apply at least to options #1 and #2.

4.3. Groups not covered by this RFC

There are a number of group types that are not specifically addressed by this RFC. A document that defines such a group MUST describe the tests required by that group.

One specific type of group would be an even-characteristic elliptic curve group. Now, these curves have cofactors greater than 1; this leads to a possibility of some information leakage. There are several ways to address this information leakage, such as performing a test analogous to the test in section 2.2, or adjusting the ECDH operation to avoid this leakage (such as "ECC CDH", where the shared secret really is $hxyG$). Because the appropriate test depends on how the group is defined, we cannot document it in advance.

4.4. Behavior Upon Test Failure

The behavior recommended in Section 2.5 is in line with generic error treatment during the IKE_SA_INIT exchange, Sec. 2.21.1 of [RFC5996]. The sender is not required to send back an error notification, and the recipient cannot depend on this notification because it is unauthenticated, and may in fact have been sent by an attacker trying to DoS the connection. Thus, the notification is only useful to debug implementation errors.

On the other hand, the error notification is secure in the sense that no secret information is leaked. All IKEv2 Diffie-Hellman groups are publicly known, and none of the tests defined here depend on any private key. In fact the tests can all be performed by an eavesdropper.

The situation when the failure occurs in the Create Child SA exchange is different, since everything is protected by an IKE SA. The peers are authenticated, and error notifications can be relied on. See Sec. 2.21.3 of [RFC5996] for more details on error handling in this case.

5. IANA Considerations

This document requests that IANA should add a column named "Recipient Tests" to the IKEv2 DH Group Transform IDs Registry [IANA-DH-Registry].

This column should initially be populated as per the following table.

Number	Recipient Tests
1, 2, 5, 14, 15, 16, 17, 18	[current], Sec. 2.1
22, 23, 24	[current], Sec. 2.2
19, 20, 21, 25, 26, 27, 28, 29, 30	[current], Sec. 2.3

Note to RFC Editor: please replace [current] by the RFC number assigned to this document.

Groups 27-30 have been recently defined in [I-D.merkle-ikev2-ke-brainpool].

Future documents that define new DH groups for IKEv2 are REQUIRED to provide this information for each new group, possibly by referring to the current document.

6. Acknowledgements

We would like to thank Dan Harkins who initially raised this issue on the ipsec mailing list. Thanks to Tero Kivinen and Rene Struik for their useful comments. Much of the text in Section 3 is taken from [RFC2412] and we would like to thank its author, Hilarie Orman.

The document was prepared using the lyx2rfc tool, created by Nico Williams.

7. References

7.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

7.2. Informative References

- [RFC2412] Orman, H., "The OAKLEY Key Determination Protocol", RFC 2412, November 1998.
- [RFC3526] Kivinen, T. and M. Kojo, "More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)", RFC 3526, May 2003.
- [RFC5114] Lepinski, M. and S. Kent, "Additional Diffie-Hellman Groups for Use with IETF Standards", RFC 5114, January 2008.
- [RFC5903] Fu, D. and J. Solinas, "Elliptic Curve Groups modulo a Prime (ECP Groups) for IKE and IKEv2", RFC 5903, June 2010.
- [I-D.merkle-ikev2-ke-brainpool]
Merkle, J. and M. Lochter, "Using the ECC Brainpool Curves for IKEv2 Key Exchange",
draft-merkle-ikev2-ke-brainpool-06 (work in progress),
April 2013.
- [NIST-800-56A]
National Institute of Standards and Technology (NIST),
"Recommendation for Pair-Wise Key Establishment Schemes

Using Discrete Logarithm Cryptography (Revised)", NIST PUB 800-56A, March 2007.

- [Kocher] Kocher, P., "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems", December 1996, <<http://www.cryptography.com/timingattack/paper.html>>.
- [Menezes] Menezes, A. and B. Ustaoglu, "On Reusing Ephemeral Keys In Diffie-Hellman Key Agreement Protocols", December 2008, <<http://www.cacr.math.uwaterloo.ca/techreports/2008/cacr2008-24.pdf>>.
- [IANA-DH-Registry]
IANA, "Internet Key Exchange Version 2 (IKEv2) Parameters, Transform Type 4 - Diffie-Hellman Group Transform IDs", Jan. 2005, <<http://www.iana.org/assignments/ikev2-parameters/ikev2-parameters.xml#ikev2-parameters-8>>.

Appendix A. Appendix: Change Log

Note to RFC Editor: please remove this section before publication.

A.1. -05

- o Resolved IESG members' comments.

A.2. -04

- o Implemented Sean's AD review, and removed the inapplicable requirement on the point at infinity.

A.3. -03

- o Added the Brainpool curves to the IANA registration table.

A.4. -02

- o Based on Tero's review: Improved the protocol behavior, and mentioned that these checks apply to Create Child SA. Added a discussion of DH timing attacks, stolen from RFC 2412.

A.5. -01

- o Corrected an author's name that was misspelled.
- o Added recipient behavior if a test fails, and the related security considerations.

A.6. -00

- o First WG document.
- o Clarified IANA actions.
- o Discussion of potential future groups not covered here.
- o Clarification re: practicality of recipient tests for DSA groups.

Authors' Addresses

Yaron Sheffer
Porticor
10 Yirmiyahu St.
Ramat HaSharon 47298
Israel

Email: yaronf.ietf@gmail.com

Scott Fluhrer
Cisco Systems
1414 Massachusetts Ave.
Boxborough, MA 01719
USA

Email: sfluhrer@cisco.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: June 7, 2013

Y. Nir
Check Point
December 4, 2012

A TCP transport for the Internet Key Exchange
draft-ietf-ipsecme-ike-tcp-01

Abstract

This document describes using TCP for IKE messages. This facilitates the transport of large messages over paths where fragments are either dropped, or where packet loss makes the use of large UDP packets unreliable.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on June 7, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

1. Introduction

The Internet Key Exchange version 2 (IKEv2), specified in [RFC5996] uses UDP to transport the exchange messages. Some of those messages may be fairly large. Specifically, the messages of the IKE_AUTH exchange can become quite large, as they may contain a chain of certificates, an "Auth" payload (that may contain a public key signature), CRLs, and some configuration information that is carried in the CFG payload.

When such UDP packets exceed the path MTU, they get fragmented. This increases the probability of packets being dropped. The retransmission mechanisms in IKE (as described in section 2.1 of RFC 5996) takes care of that as long as packet loss is at a reasonable level. More recently we have seen a number of service providers dropping fragmented packets. Firewalls and NAT devices need to keep state for each packet where some (but not all) of the fragments have passed through. This creates a burden in terms of memory, especially for high capacity devices such as Carrier-Grade NAT (CGN) or high capacity firewalls.

The BEHAVE working group has an Internet Draft describing required behavior of CGNs ([I-D.ietf-behave-lsn-requirements]). It requires CGNs to comply with [RFC4787], which in section 11 requires NAT devices to support fragments. However, some people deploying IKE have found that some ISPs have begun to drop fragments in preparation for deploying CGNs. While we all hope for a future where all devices comply with the emerging standards, or even a future where CGNs are not required, we have to make IKE work today.

The solution described in this document is to transport the IKE messages over a TCP ([RFC0793]) connection rather than over UDP. IKE packets describe their own length, so they are well-suited for transport over a stream-based connection such as TCP. The Initiator opens a TCP connection to the Responder's port 500, sends the requests and receives the responses, and then closes the connection. TCP can handle arbitrary-length messages, works well with any sized data, and is well supported by all ISP infrastructure.

1.1. Non-Goals of this Specification

Firewall traversal is not a goal of this specification. If a firewall has a policy to block IKE and/or IPsec, hiding the IKE exchange in TCP is not expected to help. Some implementations hide both IKE and IPsec in a TCP connection, usually pretending to be HTTPS by using port 443. This has a significant impact on bandwidth and gateway capacity, and even this is defeated by better firewalls. SSL VPNs tunnel IP packets over TLS, but the latest firewalls are

also TLS proxies, and are able to defeat this as well.

This document is not part of that arms race. It is only meant to allow IKE to work When faced with broken infrastructure that drops large IP packets.

1.2. Conventions 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 [RFC2119].

2. The Protocol

2.1. Initiator

An Initiator MAY try IKE using TCP for any request. It opens a TCP connection from an arbitrary port to port 500 of the Responder. When the three-way handshake completes, the Initiator MUST send the request. If the Initiator knows that this request is the last request needed at this time, it MAY half-close the TCP connection, or it MAY wait until the last response has been received. When all responses have been received, the Initiator MUST close the connection. If the peer has closed the connection before all requests have been transmitted or responded to, the Initiator SHOULD either open a new TCP connection or transmit them over UDP again.

An initiator MUST accept responses sent over IKE within the same connection, but MUST also accept responses over other transports, if the request had been sent over them as well.

An initiator that is configured to respond to IKE over TCP on some port, and is not prevented from receiving TCP connections by network address translation (see Section 3.2), MUST send an IKE_TCP_SUPPORTED notification (Section 2.5) in the Initial request.

Note that stateless cookies may be dependent on some of the parameters of the connection, so retransmitting the IKE_INITIAL request with a stateless cookie over a different transport may cause the cookie to be invalid. For this reason, retransmissions with a cookie SHOULD be sent over the same transport.

2.2. Responder

A Responder MAY accept TCP connections to port 500, and if it does, it MUST accept IKE requests over this connection. Responses to requests received over this connection MUST also go over this

connection. If the connection has closed before the Responder has had a chance to respond, it MUST NOT respond over UDP, but MUST instead wait for a retransmission over UDP or over another TCP connection.

The responder MUST accept different requests on different transports. Specifically, the Responder MUST NOT rely on subsequent requests coming over the same transport. For example, it is entirely acceptable to have the IKE_INITIAL exchange come over UDP port 500, while the IKE_AUTH request comes over TCP, and some following requests might come over UDP port 4500 (because NAT has been detected).

A responder that is configured to support IKE over TCP and receives an IKEv2 Initial request over any other transport MUST send an IKE_TCP_SUPPORTED notification (Section 2.5) in the Initial response. the responder MAY send this notification even if the Initial request was received over TCP.

If the responder has some requests of its own to send, it MUST NOT use a connection that has been opened by a peer. Instead, it MUST either use UDP or else open a new TCP connection to the original Initiator's TCP port, specified in the IKE_TCP_SUPPORTED notification in the Initial request. If the Initial request did not include this notification, the original Responder MUST NOT initiate IKE over TCP to the original Initiator.

The normal flow of things is that the Initiator opens a connection and closes its side first. The responder closes after sending the last response where the initiator has already half-closed the connection. If, however, a significant amount of time has passed, and neither new requests arrive nor the connection is closed by the initiator, the Responder MAY close or even reset the connection.

This specification makes no recommendation as to how long such a timeout should be, but a few seconds should be enough.

The stateless cookie mechanism in IKEv2 only assures that the initiator is able to respond to the address and port of the request. TCP already provides this with the three-way handshake. If the IKE_INITIAL exchange is transmitted over TCP, the stateless cookie mechanism SHOULD NOT be used.

2.3. Transmitter

The transmitter, whether an initiator transmitting a request or a responder transmitting a response MUST NOT retransmit over the same connection. TCP takes care of that. It SHOULD send the IKE header

and the IKE payloads with a single command or in rapid succession, because the receiver might block on reading from the socket.

2.4. Receiver

The IKE header is copied from RFC 5996 below for reference:

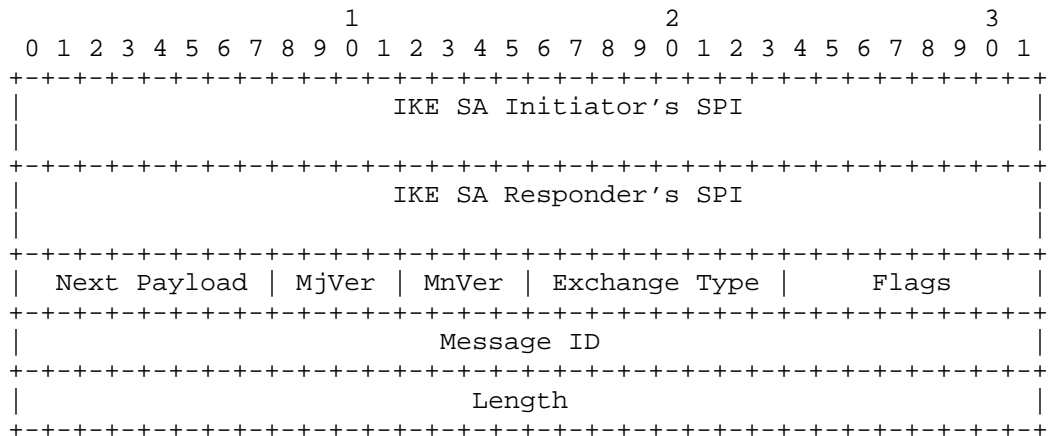


Figure 1: IKE Header Format

The receiver MUST first read in the 28 bytes that make up the IKE header. The Responder then subtracts 28 from the length field, and reads the resulting number of bytes. The combined message, comprised on 28 header bytes and whatever number of payload bytes is processed the same way as regular UDP messages. That includes retransmission detection, with one slight difference: if a retransmitted request is detected, the response is retransmitted as well, but using the current TCP connection rather than whatever other transport had been used for the original transmission of the request.

2.5. IKE_TCP_SUPPORTED Notification

This notification is sent by a responder over non-TCP transports to inform the initiator that this specification is supported and configured.

The Notify payload is formatted as follows:


```

                                1                2                3
      0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-----+-----+-----+-----+-----+-----+-----+-----+
! Next Payload !C!  RESERVED   !               Payload Length   !
+-----+-----+-----+-----+-----+-----+-----+-----+
! Protocol ID  !   SPI Size    !IKE_TCP_SUPPORTED Message Type !
+-----+-----+-----+-----+-----+-----+-----+-----+
|               TCP Port               |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

- o Protocol ID (1 octet) MUST be 0.
- o SPI Size (1 octet) MUST be zero, in conformance with section 3.10 of RFC 5996.
- o IKE_TCP_SUPPORTED Notify Message Type (2 octets) - MUST be xxxxxx, the value assigned for IKE_TCP_SUPPORTED. TBA by IANA.
- o TCP port (2 octets) - The TCP port to which the recipient should open TCP connections. This is not necessarily the same port that the IKE gateway is listening to. See Section 3.2. If the sender is not subject to network address translation, the port SHOULD be 500.

3. Operational Considerations

Most IKE messages are relatively short. All but the IKE_AUTH exchange in IKEv2 are comprised of short messages that fit in a single packet on most networks. The Informational exchange could be an exception, as it may contain arbitrary-length CFG payloads, but in practice this is not done. It is only the IKE_AUTH exchange that has long messages. UDP has advantages in lower latency and lower resource consumption, so it makes sense to use UDP whenever TCP is not required.

The requirements in Section 2.2 were written so that different requests may be sent over different transports. The initiator can choose the transport on a per-request basis. So one obvious policy would be to do everything over UDP except the specific requests that tend to become too big. This way the first messages use UDP, and the Initiator can set up the TCP connection at the same time, eliminating the latency penalty of using TCP. This may not always be the most efficient policy, though. It means that the first messages sent over TCP are relatively large ones, and TCP slow start may cause an extra roundtrip, because the message has exceeded the transmission window. An initiator using this policy MUST NOT go to TCP if the responder has not indicated support by sending the IKE_TCP_SUPPORTED notification (Section 2.5) in the Initial response.

An alternative method, that is probably easier for the Initiator to

implement, is to do an entire "mission" using the same transport. So if TCP is needed for long messages and an IKE SA has not yet been created, the Initiator will open a TCP connection, and perform all 2-4 requests needed to set up a child SA over the same connection.

Yet another policy would be to begin by using UDP, and at the same time set up the TCP connection. If at any point the TCP handshake completes, the next requests go over that connection. This method can be used to auto-discover support of TCP on the responder. This is easier for the user than configuring which peers support TCP, but has the potential of wasting resources, as TCP connections may finish the three-way handshake just when IKE over UDP has finished. The requirements from the responder ensure that all these policies will work.

3.1. Liveness Check

The TCP connections described in this document are short-lived. We do not expect them to stay for the lifetime of the SA, but to get torn down by either side within seconds of the SA being set up. Because of this, they are not well-suited for the transport of short requests such as those for liveness check.

Although liveness checks MAY be sent over TCP, this is not recommended.

On the other hand, see Section 3.2 for when liveness check should be used.

3.2. Network Address Translation

If the IKE gateway is subject to network address translation (NAT), TCP ports may be translated, so that one port on the NAT device gets translated to some other port on the gateway. In this case, the gateway MUST advertise the NAT device port in the IKE_TCP_SUPPORTED notification.

In some cases, the NAT or some other box prevents incoming TCP connections to the IKE peer behind it. In these cases, the IKE peer MUST NOT advertise support using the IKE_TCP_SUPPORTED notification.

When IKE peers detect the presence of a NAT device during the IKE exchange, they typically switch to working over UDP port 4500. Sending the IKE_AUTH messages over this UDP port creates a port mapping entry on the NAT device, and this mapping can then be used for bidirectional traffic between the peers. When using IKE over TCP, this mapping is not created, so traffic can only flow from the initiator to the responder. To make a bidirectional mapping, it is

RECOMMENDED that when NAT is detected, initiators initiate a liveness check using UDP 4500 to the responders immediately following the successful IKE_AUTH exchange.

4. Security Considerations

Most of the security considerations for IKE over TCP are the same as those for UDP as in RFC 5996.

For the Responder, listening to TCP port 500 involves all the risks of maintaining any TCP server. Precautions against DoS attacks, such as SYN cookies are RECOMMENDED. see [RFC4987] for details.

5. IANA Considerations

IANA is requested to assign a notify message type from the status types range (16418-40959) of the "IKEv2 Notify Message Types" registry with name "IKE_TCP_SUPPORTED"

No IANA action is required for the TCP port, as TCP port 500 is already allocated to "ISAKMP".

6. References

6.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

6.2. Informative References

- [I-D.ietf-behave-lsn-requirements]
Perreault, S., Yamagata, I., Miyakawa, S., Nakagawa, A., and H. Ashida, "Common requirements for Carrier Grade NATs (CGNs)", draft-ietf-behave-lsn-requirements-09 (work in progress), August 2012.
- [RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.
- [RFC4787] Audet, F. and C. Jennings, "Network Address Translation

(NAT) Behavioral Requirements for Unicast UDP", BCP 127,
RFC 4787, January 2007.

[RFC4987] Eddy, W., "TCP SYN Flooding Attacks and Common
Mitigations", RFC 4987, August 2007.

Author's Address

Yoav Nir
Check Point Software Technologies Ltd.
5 Hasolelim st.
Tel Aviv 67897
Israel

Email: ynir@checkpoint.com

Network Working Group
Internet-Draft
Updates: 7296 (if approved)
Intended status: Standards Track
Expires: April 18, 2016

T. Kivinen
INSIDE Secure
P. Wouters
Red Hat
H. Tschofenig
October 16, 2015

Generic Raw Public Key Support for IKEv2
draft-kivinen-ipsecme-oob-pubkey-14.txt

Abstract

The Internet Key Exchange Version 2 (IKEv2) protocol did have support for raw public keys, but it only supported RSA Raw public keys. In constrained environments it is useful to make use of other types of public keys, such as those based on Elliptic Curve Cryptography. This document updates RFC 7296, adding support for other types of raw public keys to IKEv2.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 18, 2016.

Copyright Notice

Copyright (c) 2015 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must

include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. Certificate Encoding Payload	3
4. Security Considerations	4
5. IANA Considerations	5
6. Acknowledgements	5
7. References	5
7.1. Normative References	5
7.2. Informative References	6
Appendix A. Examples	7
A.1. ECDSA Example	7
A.2. RSA Example	8
Authors' Addresses	10

1. Introduction

This document replaces an algorithm-specific version of raw public keys of Internet Key Exchange Version 2 (IKEv2) [RFC7296] with a generic version of raw public keys that is algorithm agnostic.

In [RFC5996] IKEv2 had support for PKCS #1 encoded RSA keys, i.e., a DER-encoded RSAPublicKey structure (see [RSA] and [RFC3447]). Other raw public key types are, however, not supported. In [RFC7296] this feature was removed, and this document adds support for raw public keys back to IKEv2 in a more generic way.

DNSSEC allows public keys to be associated with domain names for usage with security protocols like IKEv2 and Transport Layer Security (TLS) [RFC5246] but it relies on extensions in those protocols to be specified.

The Raw Public Keys in Transport Layer Security specification ([RFC7250]) adds generic support for raw public keys to TLS by re-using the SubjectPublicKeyInfo format from the X.509 Public Key Infrastructure Certificate profile [RFC5280].

This document is similar to the Raw Public Keys in Transport Layer Security specification and applies the concept to IKEv2 to support all public key formats defined by PKIX. This approach also allows future public key extensions to be supported without the need to introduce further enhancements to IKEv2.

To support new types of public keys in IKEv2 the following changes are needed:

- o A new Certificate Encoding format needs to be defined for carrying the SubjectPublicKeyInfo structure. Section 3 specifies this new encoding format.
- o A new Certificate Encoding type needs to be allocated from the IANA registry. Section 5 contains this request to IANA.

The base IKEv2 specification includes support for RSA and DSA signatures, but the Signature Authentication in IKEv2 [RFC7427] extended IKEv2 so that signature methods over any key type can be used. Implementations using raw public keys SHOULD use the Digital Signature method described in the RFC7427.

When using raw public keys, the authenticated identity is not usually the identity from the ID payload, but instead the public key itself is used as identity for the other end. This means that ID payload contents might not be useful for authentication purposes. It might still be used for policy decisions, for example to simplify the policy lookup etc. Alternatively, the ID_NULL type [RFC7619] can be used to indicate that the ID payload is not relevant to this authentication.

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 [RFC2119].

3. Certificate Encoding Payload

Section 3.6 of RFC 7296 defines the Certificate payload format as shown in Figure 1.

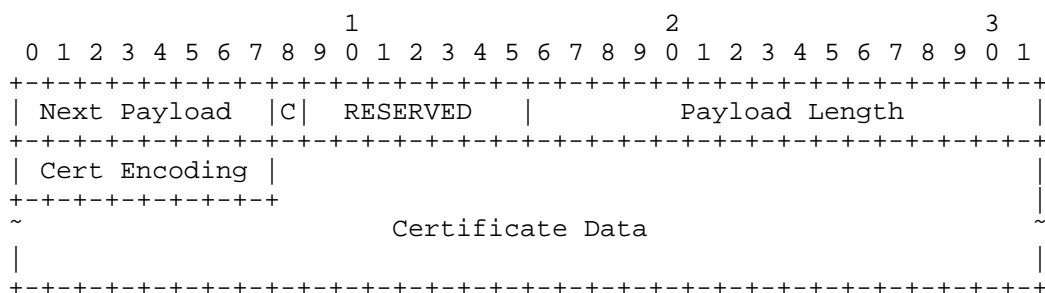


Figure 1: Certificate Payload Format.

To support raw public keys, the field values are as follows:

- o Certificate Encoding (1 octet) - This field indicates the type of certificate or certificate-related information contained in the Certificate Data field.

Certificate Encoding	Value
-----	-----
Raw Public Key	TBD

- o Certificate Data (variable length) - Actual encoding of the certificate data.

In order to provide a simple and standard way to indicate the key type when the encoding type is 'Raw Public Key', the SubjectPublicKeyInfo structure of the PKIX certificate is used. This is a very simple encoding, as most of the ASN.1 part can be included literally, and recognized by block comparison. See [RFC7250] Appendix A for a detailed breakdown. In addition, Appendix A has several examples.

In addition to the Certificate payload, the Cert Encoding for Raw Public Key can be used in the Certificate Request payload. In that case the Certification Authority field MUST be empty if the "Raw Public Key" certificate encoding is used.

For RSA keys, the implementations MUST follow the public key processing rules of section 1.2 of the Additional Algorithms and Identifiers for RSA Cryptography for PKIX ([RFC4055]) even when the SubjectPublicKeyInfo is not part of a certificate, but rather sent as a Certificate Data field. This means that RSASSA-PSS and RSASSA-PSS-params inside the SubjectPublicKeyInfo structure MUST be sent when applicable.

4. Security Considerations

An IKEv2 deployment using raw public keys needs to utilize an out-of-band public key validation procedure to be confident in the authenticity of the keys being used. One way to achieve this goal is to use a configuration mechanism for provisioning raw public keys into the IKEv2 software. "Smart object" deployments are likely to use such preconfigured public keys.

Another approach is to rely on secure DNS to associate public keys with domain names using the IPSECKEY DNS RRtype [RFC4025]. More information can be found in DNS-Based Authentication of Named Entities (DANE) [RFC6394].

This document does not change the security assumptions made by the IKEv2 specification since "Raw RSA Key" support was already available in IKEv2 in [RFC5996]. This document only generalizes raw public key support.

5. IANA Considerations

This document allocates a new value from the IKEv2 Certificate Encodings registry:

TBD Raw Public Key

6. Acknowledgements

This document reproduces some parts of the similar TLS document ([RFC7250]).

7. References

7.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<http://www.rfc-editor.org/info/rfc2119>>.
- [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <<http://www.rfc-editor.org/info/rfc5280>>.
- [RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <<http://www.rfc-editor.org/info/rfc7296>>.
- [RFC7427] Kivinen, T. and J. Snyder, "Signature Authentication in the Internet Key Exchange Version 2 (IKEv2)", RFC 7427, DOI 10.17487/RFC7427, January 2015, <<http://www.rfc-editor.org/info/rfc7427>>.
- [RFC7619] Smyslov, V. and P. Wouters, "The NULL Authentication Method in the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 7619, DOI 10.17487/RFC7619, August 2015, <<http://www.rfc-editor.org/info/rfc7619>>.

7.2. Informative References

- [RFC3447] Jonsson, J. and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", RFC 3447, DOI 10.17487/RFC3447, February 2003, <<http://www.rfc-editor.org/info/rfc3447>>.
- [RFC4025] Richardson, M., "A Method for Storing IPsec Keying Material in DNS", RFC 4025, DOI 10.17487/RFC4025, March 2005, <<http://www.rfc-editor.org/info/rfc4025>>.
- [RFC4055] Schaad, J., Kaliski, B., and R. Housley, "Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 4055, DOI 10.17487/RFC4055, June 2005, <<http://www.rfc-editor.org/info/rfc4055>>.
- [RFC4754] Fu, D. and J. Solinas, "IKE and IKEv2 Authentication Using the Elliptic Curve Digital Signature Algorithm (ECDSA)", RFC 4754, DOI 10.17487/RFC4754, January 2007, <<http://www.rfc-editor.org/info/rfc4754>>.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <<http://www.rfc-editor.org/info/rfc5246>>.
- [RFC5480] Turner, S., Brown, D., Yiu, K., Housley, R., and T. Polk, "Elliptic Curve Cryptography Subject Public Key Information", RFC 5480, DOI 10.17487/RFC5480, March 2009, <<http://www.rfc-editor.org/info/rfc5480>>.
- [RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, DOI 10.17487/RFC5996, September 2010, <<http://www.rfc-editor.org/info/rfc5996>>.
- [RFC6394] Barnes, R., "Use Cases and Requirements for DNS-Based Authentication of Named Entities (DANE)", RFC 6394, DOI 10.17487/RFC6394, October 2011, <<http://www.rfc-editor.org/info/rfc6394>>.
- [RFC7250] Wouters, P., Ed., Tschofenig, H., Ed., Gilmore, J., Weiler, S., and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", RFC 7250, DOI 10.17487/RFC7250, June 2014, <<http://www.rfc-editor.org/info/rfc7250>>.

[RSA] R. Rivest, , A. Shamir, , and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", February 1978.

Appendix A. Examples

This appendix provides examples of the actual payloads sent on the wire.

A.1. ECDSA Example

This first example uses the 256-bit ECDSA private/public key pair defined in section 8.1 of the IKEv2 ECDSA document [RFC4754].

The public key is as follows:

- o Algorithm : id-ecPublicKey (1.2.840.10045.2.1)
- o Fixed curve: secp256r1 (1.2.840.10045.3.1.7)
- o Public key x coordinate : cb28e099 9b9c7715 fd0a80d8 e47a7707 9716cbbf 917dd72e 97566eal c066957c
- o Public key y coordinate : 2b57c023 5fb74897 68d058ff 4911c20f dbe71e36 99d91339 afbb903e e17255dc

The SubjectPublicKeyInfo ASN.1 object is as follows:

```
0000 :        SEQUENCE
0002 :        SEQUENCE
0004 :        OBJECT IDENTIFIER   id-ecPublicKey (1.2.840.10045.2.1)
000d :        OBJECT IDENTIFIER   secp256r1 (1.2.840.10045.3.1.7)
0017 :        BIT STRING   (66 bytes)
00000000: 0004 cb28 e099 9b9c 7715 fd0a 80d8 e47a
00000010: 7707 9716 cbbf 917d d72e 9756 6eal c066
00000020: 957c 2b57 c023 5fb7 4897 68d0 58ff 4911
00000030: c20f dbe7 1e36 99d9 1339 afbb 903e e172
00000040: 55dc
```

The first byte (00) of the bit string indicates that there is no "number of unused bits", and the second byte (04) indicates uncompressed form ([RFC5480]). Those two octets are followed by the values of X and Y.

The final encoded SubjectPublicKeyInfo object is as follows:

```
00000000: 3059 3013 0607 2a86 48ce 3d02 0106 082a
00000010: 8648 ce3d 0301 0703 4200 04cb 28e0 999b
00000020: 9c77 15fd 0a80 d8e4 7a77 0797 16cb bf91
00000030: 7dd7 2e97 566e alc0 6695 7c2b 57c0 235f
00000040: b748 9768 d058 ff49 11c2 0fdb e71e 3699
00000050: d913 39af bb90 3ee1 7255 dc
```

This will result in the final IKEv2 Certificate Payload:

```
00000000: NN00 0060 XX30 5930 1306 072a 8648 ce3d
00000010: 0201 0608 2a86 48ce 3d03 0107 0342 0004
00000020: cb28 e099 9b9c 7715 fd0a 80d8 e47a 7707
00000030: 9716 cbbf 917d d72e 9756 6ea1 c066 957c
00000040: 2b57 c023 5fb7 4897 68d0 58ff 4911 c20f
00000050: dbe7 1e36 99d9 1339 afbb 903e e172 55dc
```

Where NN is the next payload type (i.e., the type of the payload that immediately follows this Certificate payload).

Note to the RFC editor / IANA, replace the XX above with the newly allocated Raw Public Key number (in hex notation), and remove this note.

A.2. RSA Example

This second example uses a random 1024-bit RSA key.

The public key is as follows:

- o Algorithm : rsaEncryption (1.2.840.113549.1.1.1)
- o Modulus n (1024 bits, decimal):


```
1323562071162740912417075551025599045700
3972512968992059076067098474693867078469
7654066339302927451756327389839253751712
9485277759962777278073526290329821841100
9721044682579432931952695408402169276996
5181887843758615443536914372816830537901
8976615344413864477626646564638249672329
04996914356093900776754835411
```
- o Modulus n (1024 bits, hexadecimal): bc7b4347 49c7b386 00bfa84b


```
44f88187 9a2dda08 d1f0145a f5806c2a ed6a6172 ff0dc3d4 cd601638
e8ca348e bdca5742 31cad9c7 12e209b1 fddba58a 8c62b369 038a3d1e
aa727c1f 39ae49ed 6ebc30f8 d9b52e23 385a4019 15858c59 be72f343
fbleb87b 16ffc5ab 0f8f8fe9 f7cb3e66 3d8fe9f9 ecfa1230 66f36835
8ceaefd3
```

- o Exponent e (17 bits, decimal): 65537
- o Exponent e (17 bits, hexadecimal): 10001

The SubjectPublicKeyInfo ASN.1 object is as follows:

```

0000 :      SEQUENCE
0003 :      SEQUENCE
0005 :      OBJECT IDENTIFIER rsaEncryption (1.2.840.113549.1.1.1)
0016 :      NULL
0018 :      BIT STRING  (141 bytes)
00000000: 0030 8189 0281 8100 bc7b 4347 49c7 b386
00000010: 00bf a84b 44f8 8187 9a2d da08 d1f0 145a
00000020: f580 6c2a ed6a 6172 ff0d c3d4 cd60 1638
00000030: e8ca 348e bdca 5742 31ca dc97 12e2 09b1
00000040: fddb a58a 8c62 b369 038a 3d1e aa72 7c1f
00000050: 39ae 49ed 6ebc 30f8 d9b5 2e23 385a 4019
00000060: 1585 8c59 be72 f343 fb1e b87b 16ff c5ab
00000070: 0f8f 8fe9 f7cb 3e66 3d8f e9f9 ecfa 1230
00000080: 66f3 6835 8cea efd3 0203 0100 01

```

The first byte (00) of the bit string indicates that there is no "number of unused bits". Inside that bit string there is an ASN.1 sequence having 2 integers. The second byte (30) indicates that this is beginning of the sequence, and the next byte (81) indicates the length does not fit in 7 bits, but requires one byte, so the length is in the next byte (89). Then starts the first integer with tag (02) and length (81 81). After that we have the modulus (prefixed with 0 so it will not be a negative number). After the modulus there follows the tag (02) and length (03) of the exponent, and the last 3 bytes are the exponent.

The final encoded SubjectPublicKeyInfo object is as follows:

```

00000000: 3081 9f30 0d06 092a 8648 86f7 0d01 0101
00000010: 0500 0381 8d00 3081 8902 8181 00bc 7b43
00000020: 4749 c7b3 8600 bfa8 4b44 f881 879a 2dda
00000030: 08d1 f014 5af5 806c 2aed 6a61 72ff 0dc3
00000040: d4cd 6016 38e8 ca34 8ebd ca57 4231 cadc
00000050: 9712 e209 b1fd dba5 8a8c 62b3 6903 8a3d
00000060: 1eaa 727c 1f39 ae49 ed6e bc30 f8d9 b52e
00000070: 2338 5a40 1915 858c 59be 72f3 43fb 1eb8
00000080: 7b16 ffc5 ab0f 8f8f e9f7 cb3e 663d 8fe9
00000090: f9ec fa12 3066 f368 358c eaef d302 0301
000000a0: 0001

```

This will result in the final IKEv2 Certificate Payload:

```
00000000: NN00 00a7 XX30 819f 300d 0609 2a86 4886
00000010: f70d 0101 0105 0003 818d 0030 8189 0281
00000020: 8100 bc7b 4347 49c7 b386 00bf a84b 44f8
00000030: 8187 9a2d da08 d1f0 145a f580 6c2a ed6a
00000040: 6172 ff0d c3d4 cd60 1638 e8ca 348e bdca
00000050: 5742 31ca dc97 12e2 09b1 fddb a58a 8c62
00000060: b369 038a 3d1e aa72 7c1f 39ae 49ed 6ebc
00000070: 30f8 d9b5 2e23 385a 4019 1585 8c59 be72
00000080: f343 fb1e b87b 16ff c5ab 0f8f 8fe9 f7cb
00000090: 3e66 3d8f e9f9 ecfa 1230 66f3 6835 8cea
000000a0: efd3 0203 0100 01
```

Where NN is the next payload type (i.e., the type of the payload that immediately follows this Certificate payload).

Note to the RFC editor / IANA, replace the XX above with the newly allocated Raw Public Key number, and remove this note.

Authors' Addresses

Tero Kivinen
INSIDE Secure
Eerikinkatu 28
HELSINKI FI-00180
FI

Email: kivinen@iki.fi

Paul Wouters
Red Hat

Email: pwouters@redhat.com

Hannes Tschofenig

Email: Hannes.Tschofenig@gmx.net
URI: <http://www.tschofenig.priv.at>

IP Security Maintenance and Extensions
(ipsecme)
Internet-Draft
Updates: RFC 5996 (if approved)
Intended status: Standards Track
Expires: January 22, 2015

T. Kivinen
INSIDE Secure
J. Snyder
Opus One
July 21, 2014

Signature Authentication in IKEv2
draft-kivinen-ipsecme-signature-auth-07.txt

Abstract

The Internet Key Exchange Version 2 (IKEv2) protocol has limited support for the Elliptic Curve Digital Signature Algorithm (ECDSA). The current version only includes support for three Elliptic Curve groups, and there is a fixed hash algorithm tied to each group. This document generalizes IKEv2 signature support to allow any signature method supported by the PKIX and also adds signature hash algorithm negotiation. This is a generic mechanism, and is not limited to ECDSA, but can also be used with other signature algorithms.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on January 22, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	4
3. Authentication Payload	4
4. Hash Algorithm Notification	7
5. Selecting the Public Key Algorithm	8
6. Security Considerations	8
7. IANA Considerations	9
8. Acknowledgements	10
9. References	10
9.1. Normative References	10
9.2. Informative References	11
Appendix A. Commonly used ASN.1 objects	12
A.1. PKCS#1 1.5 RSA Encryption	12
A.1.1. sha1WithRSAEncryption	12
A.1.2. sha256WithRSAEncryption	13
A.1.3. sha384WithRSAEncryption	13
A.1.4. sha512WithRSAEncryption	13
A.2. DSA	13
A.2.1. dsa-with-sha1	13
A.2.2. dsa-with-sha256	14
A.3. ECDSA	14
A.3.1. ecdsa-with-sha1	14
A.3.2. ecdsa-with-sha256	14
A.3.3. ecdsa-with-sha384	15
A.3.4. ecdsa-with-sha512	15
A.4. RSASSA-PSS	15
A.4.1. RSASSA-PSS with empty parameters	15
A.4.2. RSASSA-PSS with default parameters	16
A.4.3. RSASSA-PSS with SHA-256	16
Appendix B. IKEv2 Payload Example	17
B.1. sha1WithRSAEncryption	17
Authors' Addresses	18

1. Introduction

This document adds a new IKEv2 ([RFC5996]) authentication method to support signature methods in a more general way. The current signature-based authentication methods in IKEv2 are per-algorithm, i.e. there is one for RSA digital signatures, one for DSS digital signatures (using SHA-1) and three for different ECDSA curves, each tied to exactly one hash algorithm. This design is cumbersome when more signature algorithms, hash algorithms and elliptic curves need to be supported:

- o In IKEv2, authentication using RSA digital signatures calls for padding based on RSASSA-PKCS1-v1_5, although the newer RSASSA_PSS padding method is now recommended. (See section 5 of "Additional Algorithms and Identifiers for RSA Cryptography for use in PKIX Profile" [RFC4055]).
- o With ECDSA and DSS there is no way to extract the hash algorithm from the signature. Thus, for each new hash function to be supported with ECDSA or DSA, new authentication methods would be needed. Support for new hash functions is particularly needed for DSS because the current restriction to SHA-1 limits its security, meaning there is no point of using long keys with SHA-1.
- o The tying of ECDSA authentication methods to particular elliptic curve groups requires definition of additional methods for each new group. The combination of new ECDSA groups and hash functions will cause the number of required authentication methods to become unmanageable. Furthermore, the restriction of ECDSA authentication to a specific group is inconsistent with the approach taken with DSS.

With the selection of SHA-3, it might be possible that a signature method can be used with either SHA-3 or SHA-2. This means that a new mechanism for negotiating the hash algorithm for a signature algorithm is needed.

This document specifies two things:

1. A new authentication method which includes enough information inside the Authentication payload data so that the signature hash algorithm can be extracted (see Section 3).
2. A method to indicate supported signature hash algorithms (see Section 4). This allows the peer to know which hash algorithms are supported by the other end and use one of them (provided one is allowed by policy). There is no requirement to actually negotiate one common hash algorithm, as different hash algorithms can be used in different directions if needed.

The new digital signature method is flexible enough to include all

current signature methods (RSA, DSA, ECDSA, RSASSA-PSS, etc.), and add new methods (ECGDSA, ElGamal, etc.) in the future. To support this flexibility, the signature algorithm is specified in the same way that PKIX ([RFC5280]) specifies the signature of the Digital Certificate, by placing a simple ASN.1 object before the actual signature data. This ASN.1 object contains an OID specifying the algorithm and associated parameters. When an IKEv2 implementation supports a fixed set of signature methods with commonly used parameters, it is acceptable for the implementation to treat the ASN.1 object as a binary blob which can be compared against the fixed set of known values. IKEv2 implementations can also parse the ASN.1 and extract the signature algorithm and associated parameters.

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 [RFC2119].

3. Authentication Payload

This document specifies a new "Digital Signature" authentication method. This method can be used with any type of signature. As the authentication methods are not negotiated in IKEv2, the peer is only allowed to use this authentication method if the Notify payload of type SIGNATURE_HASH_ALGORITHMS has been sent and received by each peer.

In this authentication method, the Authentication Data field inside the Authentication Payload does not just include the signature value, as do other existing IKEv2 Authentication Payloads. Instead, the signature value is prefixed with an ASN.1 object indicating the algorithm used to generate the signature. The ASN.1 object contains the algorithm identification OID, which identifies both the signature algorithm and the hash used when calculating the signature. In addition to the OID, the ASN.1 object can contain optional parameters which might be needed for algorithms such as RSASSA-PSS (Section 8.1 of [RFC3447]).

To make implementations easier, the ASN.1 object is prefixed by the 8-bit length field. This length field allows simple implementations to know the length of the ASN.1 object without the need to parse it, so they can use it as a binary blob to be compared against known signature algorithm ASN.1 objects. Thus, simple implementations may not need to be able to parse or generate ASN.1 objects. See Appendix A for commonly used ASN.1 objects.

The ASN.1 used here is the same ASN.1 used in the AlgorithmIdentifier of PKIX (Section 4.1.1.2 of [RFC5280]), encoded using distinguished encoding rules (DER) [CCITT.X690.2002]. The algorithm OID inside the ASN.1 specifies the signature algorithm and the hash function, both of which are needed for signature verification.

Currently, only the RSASSA-PSS signature algorithm uses the optional parameters. For other signature algorithms, the parameters are either NULL or missing. Note, that for some algorithms there are two possible ASN.1 encodings, one with optional parameters included but set to NULL and the other where the optional parameters are omitted. These dual encodings exist because of the way those algorithms are specified. When encoding the ASN.1, implementations SHOULD use the preferred format called for by the algorithm specification. If the algorithm specification says "preferredPresent" then the parameters object needs to be present, although it will be NULL if no parameters are specified. If the algorithm specification says "preferredAbsent", then the entire optional parameters object is missing.

The Authentication payload is defined in IKEv2 as follows:

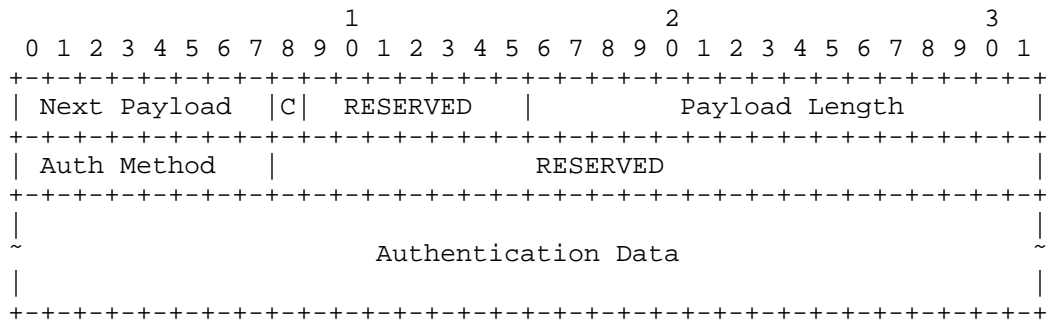


Figure 1: Authentication Payload Format.

- o Auth Method (1 octet) - Specifies the method of authentication used.

Mechanism	Value
Digital Signature	<TBD>

Computed as specified in Section 2.15 of RFC5996 using a private key associated with the public key sent in the Certificate payload, and using one of the hash algorithms sent by the other end in the Notify payload of type SIGNATURE_HASH_ALGORITHMS. If both ends send and receive SIGNATURE_HASH_ALGORITHMS Notify payloads and signature authentication is to be used, then the authentication method specified in this Authentication payload MUST be used. The format of the Authentication Data field is different from other Authentication methods and is specified below.

- o Authentication Data (variable length) - see Section 2.15 of RFC5996. For "Digital Signature" format, the Authentication data is formatted as follows:

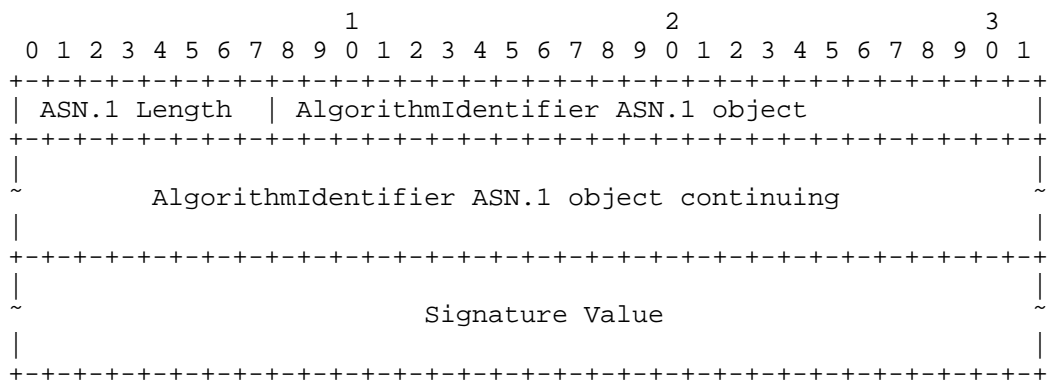


Figure 2: Authentication Data Format.

- * ASN.1 Length (1 octet) - This field contains the length of the ASN.1 encoded AlgorithmIdentifier object.
- * Algorithm Identifier (variable length) - This field contains the AlgorithmIdentifier ASN.1 object.
- * Signature Value (variable length) - This field contains the actual signature value.

There is no padding between ASN.1 object and signature value. For hash truncation, the method specified in ANSI X9.62:2005 ([X9.62]) MUST be used.

4. Hash Algorithm Notification

The supported hash algorithms that can be used for the signature algorithms are indicated with a Notify payload of type SIGNATURE_HASH_ALGORITHMS sent inside the IKE_SA_INIT exchange.

This notification also implicitly indicates support of the new "Digital Signature" algorithm method, as well as the list of hash functions supported by the sending peer.

Both ends send their list of supported hash algorithms. When calculating the digital signature, a peer MUST pick one algorithm sent by the other peer. Note that different algorithms can be used in different directions. The algorithm OID indicating the selected hash algorithm (and signature algorithm) used when calculating the signature is sent inside the Authentication Data field of the Authentication payload (with Auth Method of "Digital Signature" as defined above).

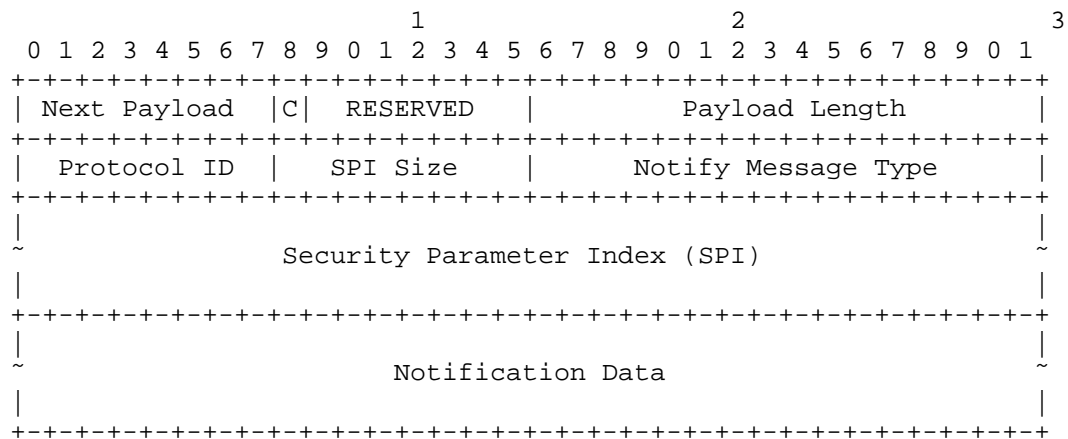


Figure 3: Notify Payload Format.

The Notify payload format is defined in RFC5996 section 3.10. When a Notify payload of type SIGNATURE_HASH_ALGORITHMS is sent, the Protocol ID field is set to 0, the SPI Size is set to 0, and the Notify Message Type is set to <TBD from status types>.

The Notification Data field contains the list of 16-bit hash algorithm identifiers from the Hash Algorithm Identifiers for the IKEv2 IANA registry. There is no padding between the hash algorithm identifiers.

5. Selecting the Public Key Algorithm

This specification does not provide a way for the peers to indicate the public / private key pair types they have. This raises the question of how the responder selects a public / private key pair type that the initiator supports. This information can be found by several methods.

One method to signal the key the initiator wants the responder to use is to indicate that in the IDr payload of the IKE_AUTH request sent by the initiator. In this case, the initiator indicates that it wants the responder to use a particular public / private key pair by sending an IDr payload which indicates that information. In this case, the responder has different identities configured, with each of those identities associated to a public / private key or key type.

Another method to ascertain the key the initiator wants the responder to use is through a Certificate Request payload sent by the initiator. For example, the initiator could indicate in the Certificate Request payload that it trusts a CA signed by an ECDSA key. This indication implies that the initiator can process ECDSA signatures, which means that the responder can safely use ECDSA keys when authenticating.

A third method is for the responder to check the key type used by the initiator, and use same key type that the initiator used. This method does not work if the initiator is using shared secret or EAP authentication (i.e., is not using public keys). If the initiator is using public key authentication, this method is the best way for the responder to ascertain the type of key the initiator supports.

If the initiator uses a public key type that the responder does not support, the responder replies with a Notify message with error type AUTHENTICATION_FAILED. If the initiator has multiple different keys, it may try a different key (and perhaps a different key type) until it finds a key that the other end accepts. The initiator can also use the Certificate Request payload sent by the responder to help decide which public key should be tried. In normal cases, when the initiator has multiple public keys, out-of-band configuration is used to select a public key for each connection.

6. Security Considerations

The "Recommendations for Key Management" ([NIST800-57]) table 2 combined with table 3 gives recommendations for how to select suitable hash functions for the signature.

This new digital signature method does not tie the Elliptic Curve to a specific hash function, which was done in the old IKEv2 ECDSA methods. This means it is possible to mix different security levels. For example, it is possible to use 512-bit Elliptic Curve with SHA1. This means that the security of the authentication method is the security of the weakest component (signature algorithm, hash algorithm, or curve). This complicates the security analysis of the system.

IKEv2 peers have a series of policy databases (see [RFC4301] section 4.4 "Major IPsec Databases") that define which security algorithms and methods should be used during establishment of security associations. To help end-users select the desired security levels for communications protected by IPsec, implementers may wish to provide a mechanism in the IKE policy databases to limit the mixing of security levels or to restrict combinations of protocols.

Security downgrade attacks, where more secure methods are deleted or modified from a payload by a Man-in-the-Middle to force lower levels of security, are not a significant concern in IKEv2 Authentication Payloads as discussed in this RFC. This is because a modified AUTH payload will be detected when the peer computes a signature over the IKE messages.

One specific class of downgrade attacks requires selection of catastrophically weak ciphers. In this type of attack, the Man-in-the-Middle attacker is able to "break" the cryptography in real time. This type of downgrade attack should be blocked by policy regarding cipher algorithm selection, as discussed above.

The hash algorithm registry does not include MD5 as a supported hash algorithm, as it is not considered safe enough for signature use ([WY05]).

The current IKEv2 protocol uses RSASSA-PKCS1-v1_5, which has known security vulnerabilities ([KA08], [ME01]) and does not allow using newer padding methods such as RSASSA-PSS. The new method described in this RFC allows using other padding methods.

The current IKEv2 protocol only allows use of normal DSA with SHA-1, which means the security of the authentication is limited to the security of SHA-1. This new method allows using longer keys and longer hashes with DSA.

7. IANA Considerations

This document creates a new IANA registry for IKEv2 Hash Algorithms.

Changes and additions to this registry are by expert review.

The initial values of this registry are:

Hash Algorithm	Value
-----	-----
RESERVED	0
SHA1	1
SHA2-256	2
SHA2-384	3
SHA2-512	4

MD5 is not included in the hash algorithm list as it is not considered safe enough for signature hash uses.

Values 5-1023 are reserved to IANA. Values 1024-65535 are for private use among mutually consenting parties.

This specification also adds one new "IKEv2 Notify Message Types - Status Types" value for SIGNATURE_HASH_ALGORITHMS, and adds one new "IKEv2 Authentication Method" value for "Digital Signature".

8. Acknowledgements

Most of this work was based on the work done in the IPsecME design team for the ECDSA. The design team members were: Dan Harkins, Johannes Merkle, Tero Kivinen, David McGrew, and Yoav Nir.

9. References

9.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.
- [RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

9.2. Informative References

- [CCITT.X690.2002]
International Telephone and Telegraph Consultative Committee, "ASN.1 encoding rules: Specification of basic encoding Rules (BER), Canonical encoding rules (CER) and Distinguished encoding rules (DER)", CCITT Recommendation X.690, July 2002.
- [KA08] Kuehn, U., Pyshkin, A., Tews, E., and R. Weinmann, "Variants of Bleichenbacher's Low-Exponent Attack on PKCS#1 RSA Signatures", Proc. Sicherheit 2008 pp.97-109.
- [ME01] Menezes, A., "Evaluation of Security Level of Cryptography: RSA-OAEP, RSA-PSS, RSA Signature", December 2001.
- [NIST800-57]
Barker, E., Barker, W., Burr, W., Polk, W., and M. Smid, "Recommendations for Key Management", NIST SP 800-57, March 2007.
- [RFC3279] Bassham, L., Polk, W., and R. Housley, "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3279, April 2002.
- [RFC3447] Jonsson, J. and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", RFC 3447, February 2003.
- [RFC4055] Schaad, J., Kaliski, B., and R. Housley, "Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 4055, June 2005.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.
- [RFC5480] Turner, S., Brown, D., Yiu, K., Housley, R., and T. Polk, "Elliptic Curve Cryptography Subject Public Key Information", RFC 5480, March 2009.
- [RFC5758] Dang, Q., Santesson, S., Moriarty, K., Brown, D., and T. Polk, "Internet X.509 Public Key Infrastructure: Additional Algorithms and Identifiers for DSA and ECDSA", RFC 5758, January 2010.

- [RFC5912] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)", RFC 5912, June 2010.
- [WY05] Wang, X. and H. Yu, "How to break MD5 and other hash functions", Proceedings of EuroCrypt 2005, Lecture Notes in Computer Science Vol. 3494, 2005.
- [X9.62] American National Standards Institute, "Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)", ANSI X9.62, November 2005.

Appendix A. Commonly used ASN.1 objects

This section lists commonly used ASN.1 objects in binary form. This section is not normative, and these values should only be used as examples. If the ASN.1 object listed in Appendix A and the ASN.1 object specified by the algorithm differ, then the algorithm specification must be used. These values are taken from "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)" ([RFC5912]).

A.1. PKCS#1 1.5 RSA Encryption

The algorithm identifiers here include several different ASN.1 objects with different hash algorithms. This document only includes the commonly used ones, i.e. the ones using SHA-1 or SHA-2 as hash function. Some other algorithms (such as MD2 and MD5) are not safe enough to be used as signature hash algorithms, and are omitted. The IANA registry does not have code points for these other algorithms with RSA Encryption. Note that there are no optional parameters in any of these algorithm identifiers, but all included here need NULL optional parameters present in the ASN.1.

See "Algorithms and Identifiers for PKIX Profile" ([RFC3279]) and "Additional Algorithms and Identifiers for RSA Cryptography for use in PKIX Profile" ([RFC4055]) for more information.

A.1.1. sha1WithRSAEncryption

```
sha1WithRSAEncryption OBJECT IDENTIFIER ::= { iso(1) member-body(2)
us(840) rsadsi(113549) pkcs(1) pkcs-1(1) 5 }
```

Parameters are required, and they must be NULL.

Name = sha1WithRSAEncryption, oid = 1.2.840.113549.1.1.5
Length = 15
0000: 300d 0609 2a86 4886 f70d 0101 0505 00

A.1.2. sha256WithRSAEncryption

sha256WithRSAEncryption OBJECT IDENTIFIER ::= { pkcs-1 11 }

Parameters are required, and they must be NULL.

Name = sha256WithRSAEncryption, oid = 1.2.840.113549.1.1.11
Length = 15
0000: 300d 0609 2a86 4886 f70d 0101 0b05 00

A.1.3. sha384WithRSAEncryption

sha384WithRSAEncryption OBJECT IDENTIFIER ::= { pkcs-1 12 }

Parameters are required, and they must be NULL.

Name = sha384WithRSAEncryption, oid = 1.2.840.113549.1.1.12
Length = 15
0000: 300d 0609 2a86 4886 f70d 0101 0c05 00

A.1.4. sha512WithRSAEncryption

sha512WithRSAEncryption OBJECT IDENTIFIER ::= { pkcs-1 13 }

Parameters are required, and they must be NULL.

Name = sha512WithRSAEncryption, oid = 1.2.840.113549.1.1.13
Length = 15
0000: 300d 0609 2a86 4886 f70d 0101 0d05 00

A.2. DSA

With DSA algorithms, optional parameters are always omitted. Only algorithm combinations for DSA listed in the IANA registry are included.

See "Algorithms and Identifiers for PKIX Profile" ([RFC3279]) and "PKIX Additional Algorithms and Identifiers for DSA and ECDSA" ([RFC5758]) for more information.

A.2.1. dsa-with-sha1

dsa-with-sha1 OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840) x9-57(10040) x9algorithm(4) 3 }

Parameters are absent.

Name = dsa-with-sha1, oid = 1.2.840.10040.4.3
Length = 11
0000: 3009 0607 2a86 48ce 3804 03

A.2.2. dsa-with-sha256

dsa-with-sha256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2)
country(16) us(840) organization(1) gov(101) csor(3) algorithms(4)
id-dsa-with-sha2(3) 2 }

Parameters are absent.

Name = dsa-with-sha256, oid = 2.16.840.1.101.3.4.3.2
Length = 13
0000: 300b 0609 6086 4801 6503 0403 02

A.3. ECDSA

With ECDSA algorithms, the optional parameters are always omitted.
Only algorithm combinations for ECDSA listed in the IANA registry are
included.

See "Elliptic Curve Cryptography Subject Public Key Information"
([RFC5480]), "Algorithms and Identifiers for PKIX Profile"
([RFC3279]) and "PKIX Additional Algorithms and Identifiers for DSA
and ECDSA" ([RFC5758] for more information.

A.3.1. ecdsa-with-sha1

ecdsa-with-SHA1 OBJECT IDENTIFIER ::= { iso(1) member-body(2) us(840)
ansi-X9-62(10045) signatures(4) 1 }

Parameters are absent.

Name = ecdsa-with-sha1, oid = 1.2.840.10045.4.1
Length = 11
0000: 3009 0607 2a86 48ce 3d04 01

A.3.2. ecdsa-with-sha256

ecdsa-with-SHA256 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
us(840) ansi-X9-62(10045) signatures(4) ecdsa-with-SHA2(3) 2 }

Parameters are absent.

```
Name = ecdsa-with-sha256, oid = 1.2.840.10045.4.3.2
Length = 12
0000: 300a 0608 2a86 48ce 3d04 0302
```

A.3.3. ecdsa-with-sha384

```
ecdsa-with-SHA384 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
us(840) ansi-X9-62(10045) signatures(4) ecdsa-with-SHA2(3) 3 }
```

Parameters are absent.

```
Name = ecdsa-with-sha384, oid = 1.2.840.10045.4.3.3
Length = 12
0000: 300a 0608 2a86 48ce 3d04 0303
```

A.3.4. ecdsa-with-sha512

```
ecdsa-with-SHA512 OBJECT IDENTIFIER ::= { iso(1) member-body(2)
us(840) ansi-X9-62(10045) signatures(4) ecdsa-with-SHA2(3) 4 }
```

Parameters are absent.

```
Name = ecdsa-with-sha512, oid = 1.2.840.10045.4.3.4
Length = 12
0000: 300a 0608 2a86 48ce 3d04 0304
```

A.4. RSASSA-PSS

With RSASSA-PSS, the algorithm object identifier must always be id-RSASSA-PSS, and the hash function and padding parameters are conveyed in the parameters (which are not optional in this case). See [RFC4055] for more information.

A.4.1. RSASSA-PSS with empty parameters

```
id-RSASSA-PSS OBJECT IDENTIFIER ::= { pkcs-1 10 }
```

Parameters are empty, but the ASN.1 part of the sequence must be present. This means default parameters are used.

```
0000 : SEQUENCE
0002 :   OBJECT IDENTIFIER  RSASSA-PSS (1.2.840.113549.1.1.10)
000d :   SEQUENCE
```

```
Length = 15
0000: 300d 0609 2a86 4886 f70d 0101 0a30 00
```

A.4.2. RSASSA-PSS with default parameters

id-RSASSA-PSS OBJECT IDENTIFIER ::= { pkcs-1 10 }

Here the parameters are present, and contain the default parameters, i.e. hashAlgorithm of SHA-1, maskGenAlgorithm of mgf1SHA1, saltLength of 20, trailerField of 1.

```
0000 : SEQUENCE
0002 :   OBJECT IDENTIFIER  RSASSA-PSS (1.2.840.113549.1.1.10)
000d :   SEQUENCE
000f :     CONTEXT 0
0011 :       SEQUENCE
0013 :         OBJECT IDENTIFIER  id-sha1 (1.3.14.3.2.26)
001a :         NULL
001c :       CONTEXT 1
001e :       SEQUENCE
0020 :         OBJECT IDENTIFIER  1.2.840.113549.1.1.8
002b :         SEQUENCE
002d :         OBJECT IDENTIFIER  id-sha1 (1.3.14.3.2.26)
0034 :         NULL
0036 :       CONTEXT 2
0038 :         INTEGER  0x14 (5 bits)
003b :       CONTEXT 3
003d :         INTEGER  0x1 (1 bits)
```

Name = RSASSA-PSS with default parameters,
oid = 1.2.840.113549.1.1.10

Length = 64

```
0000: 303e 0609 2a86 4886 f70d 0101 0a30 31a0
0010: 0b30 0906 052b 0e03 021a 0500 a118 3016
0020: 0609 2a86 4886 f70d 0101 0830 0906 052b
0030: 0e03 021a 0500 a203 0201 14a3 0302 0101
```

A.4.3. RSASSA-PSS with SHA-256

id-RSASSA-PSS OBJECT IDENTIFIER ::= { pkcs-1 10 }

Here the parameters are present, and contain hashAlgorithm of SHA-256, maskGenAlgorithm of SHA-256, saltLength of 32, trailerField of 1.

```

0000 : SEQUENCE
0002 :   OBJECT IDENTIFIER  RSASSA-PSS (1.2.840.113549.1.1.10)
000d :   SEQUENCE
000f :     CONTEXT 0
0011 :     SEQUENCE
0013 :       OBJECT IDENTIFIER  id-sha256 (2.16.840.1.101.3.4.2.1)
001e :       NULL
0020 :     CONTEXT 1
0022 :     SEQUENCE
0024 :       OBJECT IDENTIFIER  1.2.840.113549.1.1.8
002f :       SEQUENCE
0031 :         OBJECT IDENTIFIER id-sha256 (2.16.840.1.101.3.4.2.1)
003c :         NULL
003e :     CONTEXT 2
0040 :       INTEGER    0x20 (6 bits)
0043 :     CONTEXT 3
0045 :       INTEGER    0x1 (1 bits)

```

Name = RSASSA-PSS with sha-256, oid = 1.2.840.113549.1.1.10

Length = 72

```

0000: 3046 0609 2a86 4886 f70d 0101 0a30 39a0
0010: 0f30 0d06 0960 8648 0165 0304 0201 0500
0020: a11c 301a 0609 2a86 4886 f70d 0101 0830
0030: 0d06 0960 8648 0165 0304 0201 0500 a203
0040: 0201 20a3 0302 0101

```

Appendix B. IKEv2 Payload Example

B.1. sha1WithRSAEncryption

The IKEv2 AUTH payload would start like this:

```

00000000: NN00 00LL XX00 0000 0f30 0d06 092a 8648
00000010: 86f7 0d01 0105 0500 ....

```

Where the NN will be the next payload type (i.e. the value depends on the next payload after this Authentication payload), the LL will be the length of this payload, and after the sha1WithRSAEncryption ASN.1 block (15 bytes) there will be the actual signature, which is omitted here.

Note to the RFC editor / IANA, replace the XX above with the newly allocated authentication method type for Digital Signature, and remove this note.

Authors' Addresses

Tero Kivinen
INSIDE Secure
Eerikinkatu 28
HELSINKI FI-00180
FI

Email: kivinen@iki.fi

Joel Snyder
Opus One
1404 East Lind Road
Tucson, AZ 85719

Phone: +1 520 324 0494
Email: jms@opus1.com

IPSECME
Internet-Draft
Intended status: Standards Track
Expires: August 19, 2013

D. Migault (Ed)
Francetelecom - Orange
February 15, 2013

IKEv2 Alternate Outer IP Address Extension
draft-mglt-ipsecme-alternate-outer-address-00.txt

Abstract

Current IKEv2 protocol has been designed to establish VPNs with the same outer IP addresses as those used for the IKEv2 channel. This describes the alternate outer IP address extension, and IKEv2 extension that enables the VPN End User to negotiate a VPN on different interfaces as those used for the IKEv2 channel.

Thus, this extension makes possible a VPN End User with multiple interfaces to set an IPsec tunnel on each interface with a Security Gateway by using a single IKEv2 channel instead of using an IKEv2 channel per interface. Similarly, for distributed Security Gateways, it also makes possible to split the IKEv2 and IPsec traffic on different interfaces.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 19, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents

(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Requirements notation	3
2. Introduction	3
3. Terminology	3
4. Alternate outer address scenarios	4
4.1. VPN End User with Multiple Interfaces	4
4.2. Security Gateway with Multiple Interfaces	6
4.3. Distributed Security Gateways	7
5. Protocol Overview	7
5.1. Alternate outer IP addresses Transform	8
5.2. Initiator: Sending OADD Transforms in Proposals	10
5.3. Responder: Receiving OADD Transforms in Proposals	11
5.4. Incompatible Proposal with OADD Transforms	11
5.5. Supporting alternate outer IP address exchange	11
5.6. Basic Exchange	12
6. Payload Formats	13
6.1. Outer IP address Transform OADD	14
6.2. IP Attribute with IP addresses	15
6.3. IP Attribute indicating ANY_IP	15
6.4. Alternate Outer IP Address Notify Payload	16
7. NAT considerations	16
7.1. Prohibiting NAT	18
7.2. NAT detection	19
7.3. The VPN End User does not know the NATted IP addresses	19
7.4. The VPN End User does know the NATted IP addresses	20
8. IANA Considerations	20
9. Security Considerations	21
10. Acknowledgment	21
11. References	21
11.1. Normative References	21
11.2. Informational References	21
Appendix A. Document Change Log	22
Author's Address	22

1. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Introduction

When a VPN End User establishes a VPN with a Security Gateway, it starts by establishing an authenticated channel for IKEv2. Then the VPN Security Associations [RFC4301] are negotiated via the IKEv2 [RFC5996] channel. Once the peers agree on the Security Associations, the VPN can be used.

Currently, IKEv2 does not negotiate the outer IP addresses of the VPN. The security Association set these VPN outer IP addresses as the IP addresses used by the IKEv2 channel.

These implicit values are perfect for VPN End Users with a single interface. This was the case for a long time, making them unnecessary to be negotiated. However, today's VPN End Users and Security Gateways have multiple interfaces. Relying on the default value of the VPN outer IP addresses makes it hard, - or at least in a non optimal way - to take advantage of multiple interfaces. This document specifies how alternate outer IP addresses can be negotiated during the Security Association negotiation. This involves new signaling, thus the document also specifies how the VPN End User and the Security Gateway can optionally inform each other they support the alternate outer IP address extension.

The remaining of this document is as follows. Section 3 defines the terminology used in this document. Section 4 provides scenarios that motivate this alternate outer IP address extension. Section 5 describes the new protocol, as well as the new involved entities and Section 6 describes the payload format defined for the protocol. In this document, we assumed that no NAT are between the VPN End User and the Security Gateway, however, Section 7 provides some considerations when NAT is used.

The alternate outer IP address extension provides VPN End Users and Security Gateway a way to take advantage of multiple interfaces for a VPN service.

3. Terminology

This section defines terms and acronyms used in this document.

- VPN End User: designates the End User that initiates the VPN with a Security Gateway. This End User may be mobile and moves its VPN from on Security Gateway to the other.
- Security Gateway: designates a point of attachment for the VPN service. In this document, the VPN service is provided by multiple Security Gateways. Each Security Gateway may be considered as a specific hardware.
- Security Association (SA): The Security Association is defined in [RFC4301].

4. Alternate outer address scenarios

This section provides scenarios where a VPN End User and a Security Gateways share more than one VPN. For each scenario, the document describes the alternatives that currently exist, their limitations and the motivations for the alternate outer IP address extension. The scenarios herein are a subset of the scenarios described in [I-D.mglt-mif-security-requirements].

4.1. VPN End User with Multiple Interfaces

More and more terminals have multiple interfaces, and a VPN End User may take advantage of these multiple interfaces by setting multiple tunnels with its Security Gateways as represented in figure 1. A typical example would be a VPN End User attached to its Radio Access Network via Interface_0 and attached to a WLAN access point via Interface_1. The VPN End User may use one or the other interface according to the Quality of Service or the fees associated to each network. In figure 1. the VPN End User has established two distinct VPNs, one on each of its interfaces. Both VPNs are attached to the same Security Gateway interface. A packet can be sent or received from either one or the other VPN.

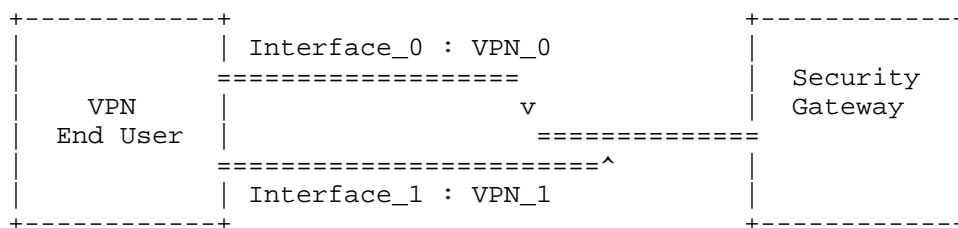


Figure 1: VPN End User with Multiple Interfaces

SAs negotiated for the VPN_0 and VPN_1 have the same network

configuration except that the outer interface of VPN_0 on the End User side is Interface_0 whereas VPN_1 has Interface_1. More specifically, these SAs have the same Selectors.

[RFC4301] section 4.1 states that parallel SAs are compliant with the IPsec architecture, and that traffic may be sent to one or the other VPN, for example, according to the Differentiated Services Code Point (DSCP). DSCP is called a "classifier" which differs from the Selector. How the End User chooses which interface to use is beyond the scope of this document.

As mentioned in [RFC5996] the VPN uses the IP addresses of the IKEv2 channel as outer IP addresses. One way to establish these two VPNs is to create an IKEv2 channel for each interface. This results in unnecessary IKE negotiations with multiple authentications $\text{Nbr}(\text{EU_interfaces}) \times \text{Nbr}(\text{SG_interface}) \times \text{Nbr}(\text{Flows})$. This number rapidly grows with the number of involved interfaces both on the Security Gateway and on the End User.

[RFC6027] section 3.8 mentions that peers using different IP addresses for the VPN and the IKEv2 channel SHOULD be modified unless they may drop the packets. The alternate outer IP address described in this document is described so that any VPN End User can interact with any Security Gateway.

[I-D.arora-ipsecme-ikev2-alt-tunnel-addresses] addresses this issue. The End User VPN indicates during the SA negotiation the outer IP address it wants, and in return the Security Gateway indicates the outer IP address of the Security Gateway. Motivations for [I-D.arora-ipsecme-ikev2-alt-tunnel-addresses] is a cluster of Security Gateways that splits the IKEv2 traffic and the VPN traffic, so that the VPN traffic avoids overloading some equipments like firewalls or load balancers for example.

[I-D.arora-ipsecme-ikev2-alt-tunnel-addresses] would also address the case of figure 1 because the the path used by the VPN is defined by the interface used by the VPN End User VPN. This results from the fact that the Security Gateway has only one interface. However, [I-D.arora-ipsecme-ikev2-alt-tunnel-addresses] would need slight modifications in order to address the more general case where VPN End User and the Security Gateways have multiple interfaces. In that case, a path would be defined not by a single interface (as in figure 1), but by a pair of interface.

In addition to path negotiation, [I-D.arora-ipsecme-ikev2-alt-tunnel-addresses] uses a Notify Payload that is not bound to a SA Proposal, thus making multiple SA Proposals with different outer IP address difficult. Again this case is very

specific to multiple interfaces. Even though the protocol described in this document address these limitations, it remains very closed to [I-D.arora-ipsecme-ikev2-alt-tunnel-addresses].

4.2. Security Gateway with Multiple Interfaces

In the scenario presented in figure 2, the VPN End User has two interfaces and the VPN End User has a single interface. Like the VPN End User with multiple interfaces presented in Section 4.1, we suppose that the VPNs are established by the VPN End User with the Security Gateway. Unlike the scenarios of Section 4.1, motivations for choosing VPN_0 or VPN_1 are not associated to the interface used by the VPN End User, but the path taken by the packets. As a result, the VPN End User cares about both source and destination outer IP addresses that defines the path.

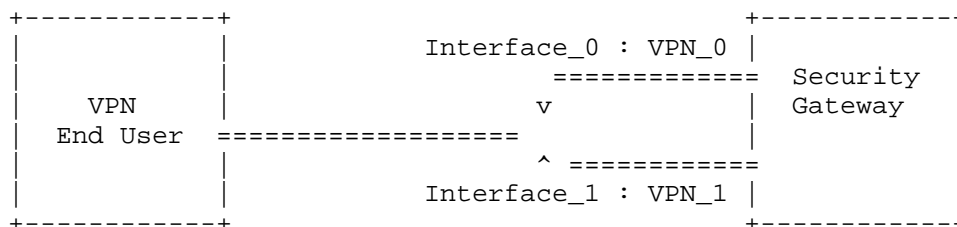


Figure 2: Security Gateway with Multiple Interfaces

Comments of Section 4.1 also applies to this scenario, but this scenario stresses that the choice of the VPN outer IP addresses SHOULD result from a negotiation between the two peers, and both outer IP addresses SHOULD be negotiated.

Note that the scenario described in figure 2, considers that all interfaces are used to setup all different VPNs. As described in Section 4.1, if VPN End Users and Security Gateways have both multiple interfaces, setting up all possible tunnels may be unnecessarily heavy. As a result, the VPN End User SHOULD be able to negotiate both outer IP addresses of its VPN.

Note that if the VPN End User negotiates the outer IP address used by the Security Gateway, the VPN End User may know in advance what interfaces are available. It is beyond the scope of this document to define how the VPN End User may know this information. MOBIKE [RFC4555] defines the ADDITIONAL_IP*_ADDRESSES Notify Payload, and [I-D.mglt-ipsecme-security-gateway-discovery] defines how these pieces of information may be provided by other Security Gateways.

initiator and responder, as the peer initiating the negotiation.

Note that these negotiations makes possible that any peer can negotiate one, or both outer IP address, that is to say, the outer IP address source and destination.

Section 5.1 briefly reminds how the Security Association' parameters are negotiated with IKEv2, and then proposes the new involved payloads to negotiate the outer IP addresses. Basically a new Alternate Outer Address Transform (OADD) and a new IP Attribute are defined. Section 5.2 and Section 5.3 and Section 5.4 are focused on the exchanged when both peers support the alternate outer IP address extension. Section 5.2 describes how the initiator builds a SA Proposal and Section 5.3 defines how the responder handles it. Section 5.4 defines the case where the Proposal MUST be discarded. Although not mandatory, there MAY be an advantage that peers are informed whether the alternate outer IP address is supported or not before sending Proposals. Section 5.5 presents how peers can inform each other the support this extension. At last, Section 5.6 illustrates the different exchanged described in the document.

5.1. Alternate outer IP addresses Transform

This section does not intend to explain how SAs are negotiated, and the reader is expected to refer to [RFC5996] section 3.3. This section briefly sums up the different type of payload involved in order to clarify our purpose. Figure 4 is copied from [RFC5996] to illustrate concepts involved in the Security Association negotiation.

```

SA Payload
|
+---- Proposal #1 ( Proto ID = ESP(3), SPI size = 4,
|                  7 transforms,          SPI = 0x052357bb )
|
|   +--- Transform ENCR ( Name = ENCR_AES_CBC )
|   |   +--- Attribute ( Key Length = 128 )
|   |
|   +--- Transform ENCR ( Name = ENCR_AES_CBC )
|   |   +--- Attribute ( Key Length = 192 )
|   |
|   +--- Transform ENCR ( Name = ENCR_AES_CBC )
|   |   +--- Attribute ( Key Length = 256 )
|   |
|   +--- Transform INTEG ( Name = AUTH_HMAC_SHA1_96 )
|   +--- Transform INTEG ( Name = AUTH_AES_XCBC_96 )
|   +--- Transform ESN ( Name = ESNs )
|   +--- Transform ESN ( Name = No ESNs )
|
+---- Proposal #2 ( Proto ID = ESP(3), SPI size = 4,
|                  4 transforms,          SPI = 0x35ald6f2 )
|
|   +--- Transform ENCR ( Name = AES-GCM with a 8 octet ICV )
|   |   +--- Attribute ( Key Length = 128 )
|   |
|   +--- Transform ENCR ( Name = AES-GCM with a 8 octet ICV )
|   |   +--- Attribute ( Key Length = 256 )
|   |
|   +--- Transform ESN ( Name = ESNs )
|   +--- Transform ESN ( Name = No ESNs )

```

Figure 4: Security Association Payload Structure

A Security Association is defined by various parameters such as Encryption (ENCR) or Integrity (INTEG), Pseudorandom Function (PRF), Diffie-Hellman group (D-H) or Extended Sequence Numbers (ESN). These parameters are defined through Transforms and each parameter is a Transform Type.

A Security Association is negotiated through the SA Payload which contains one or more Proposals Payloads. Each Proposal contains one or multiple acceptable "values" for each Transformed Type. These "values" can be seen as an OR. The Proposal is accepted if for each Transform Type one of the proposed "value" is accepted by the responder. If the responder cannot choose an acceptable "value" for each Transform Type, the proposition is rejected. A "value" is composed of a Transform ID, like the name of the encryption algorithm, and eventually one or more Attributes, like the key length

for example.

In our case, we consider a new Transform Type OADD. This Transform Type has two Transform ID (INIT or RESP) that designates the initiator outer IP address (INIT) or the responder outer IP address (RESP). The Attributes associated to each Transform ID is the IP Attribute that can be an IPv4 address, an IPv6 address or a specific value.

5.2. Initiator: Sending OADD Transforms in Proposals

In Section 5.2 and Section 5.3 we suppose that both the initiator and the responder support the alternate outer IP address extension, that no USE_TRANSPORT_MODE Notify Payload is sent in conjunction of the SA Payload, and that the Proposal Payload as defined in [RFC5996] Section 3.3.1 has its Protocol ID set to AH or ESP. Other cases are discussed in Section 5.4

If the initiator wants to propose the Security Gateway to choose among a set of the initiator's interfaces IP_init_0, ..., IP_init_k for the VPN outer IP address, it MUST include k+1 Transforms with Transform Type OADD and Transform ID set to INIT. The Transform is associated to the Attribute of Type IP. Transform Attributes are defined in [RFC5996] 3.3.5.

Similarly, if the initiator wants to select on the Security Gateway one interface among a set of interface IP_resp_0, ..., IP_resp_l, it MUST include l+1 OADD Transform with Transform ID set to RESP, and an Attribute of Type IP.

If the initiator does not know the interface that the responder may choose, it may indicate the responder to define the most appropriated interface with a OADD Transform with Transform ID set to RESP and an Attribute of Type with the specific value ANY_IP.

If no OADD Transform with Transform ID set to INIT (Respectively RESP) are provided in the Proposal, the default value for the outer IP address is the one used by the IKEv2 channel. More specifically, if the initiator considers the interface used for the IKEv2 channel as an alternative to other IP addresses, a OADD Transform with this IP address MUST explicitly be in the Proposal.

Note that a Proposal does not need to have both OADD Transform with Transform ID INIT and RESP. The initiator can choose to have only OADD Transforms with Transform ID INIT (respectively RESP).

5.3. Responder: Receiving OADD Transforms in Proposals

As mentioned in Section 5.2, we suppose the responder supports the alternate outer IP address extension. If a Proposal contains one or multiple OADD with a Transform ID set to INIT (respectively RESP), the responder choose one of these. If selected OADD Transform (INIT or RESP) with an IP Attribute, the responder returns the Transform without modification. Otherwise, if selected OADD Transform is with an ANY_IP Attribute, the responder returns a IP Attribute with the correct value.

If the responder has no OADD Transform with Transform ID INIT (respectively RESP), then by default the outer IP address of the VPN is equal to the IP address used by the IKEv2 channel.

5.4. Incompatible Proposal with OADD Transforms

The alternate outer IP address extension only makes sense for the IPsec tunnel mode. The SA Payload with Proposals that contains one or more OADD Transforms MUST NOT be used with a USE_TRANSPORT_MODE Notify Payload. Responder MUST reject these Proposals.

Similarly, Proposals with a Protocol other than AH or ESP, (that is to say IKE), MUST NOT be used with OADD Transforms. Responder MUST reject these Proposals.

As mentioned in [RFC5996] Section 3.3.6, a responder that does not support the alternate outer IP address extension MUST reject any Proposal that contains a Transform with a Transform Type OADD. If the responder rejects all Proposals, it MUST send a NO_PROPOSAL_CHOSEN Notify Payload.

5.5. Supporting alternate outer IP address exchange

This section describes an informational exchange where each peer informs the other that it supports the alternate outer IP address extension. This exchange is not mandatory, but is recommended as it MAY ease to format the Proposals for the Security Association negotiation.

In fact the negotiation of the alternate outer IP address is included in SA negotiation. As described in Section 5.1, this introduces new Transform Type and new Attributes. [RFC5996] Section 3.3.6 mentions that a peer does not understand the new Transform Type or the new Attributes, it MUST reject the Proposal. As a result, if the initiator does not know if the responder supports the alternate outer IP address extension, it SHOULD include proposals without the associated Transform Type and Attributes to avoid that all Proposals

are rejected by the responder and receives a NO_PROPOSAL_CHOSEN Notify Payload.

To limit the number of proposals to be sent by the initiator during the SA negotiation, we define the supporting alternate outer IP address exchange where the initiator can advertise it supports the alternate outer IP address extension by sending a `ALTERNATE_OUTER_IP_ADDRESS_SUPPORTED` Notify Payload. When a node receives this Notify Payload and support the alternate outer IP address extension, it **MUST** send back the same Notify Payload.

5.6. Basic Exchange

Figure 5 provides a basic exchange. The initiator and the responder agree on supporting the alternate outer IP address extension. This exchange is optional but recommended. In Figure 5 this exchange occurs during the `IKE_INIT` exchange, but it **MAY** occur anytime.

The SA negotiation consists in sending multiple Proposals. In figure 5, the OADD Transform specify the initiator and responder's IP address. The responder choose one of the proposed transformed.

```

Initiator                                Responder
-----
HDR, SAi1, KEi, Ni    -->
N(ALTERNATE_OUTER_IP_ADDRESS_SUPPORTED)
N(NAT_DETECTION_SOURCE_IP),
N(NAT_DETECTION_DESTINATION_IP)
    <-- HDR, SAR1, KEr, Nr, [CERTREQ]
        N(ALTERNATE_OUTER_IP_ADDRESS_SUPPORTED)
        N(NAT_DETECTION_SOURCE_IP),
        N(NAT_DETECTION_DESTINATION_IP)

==== From this exchange:
    - the Initiator and the Responder support the alternate
      outer IP address extension
    - no NAT has been detected      =====

HDR, SK {IDi, [CERT,] [CERTREQ,]
  [IDr,] AUTH,  TSi, TSr
  SAI2( Proposal(ENCR, INTEG, ESN,    < proposes IP1, IP2 for
        OADD(INIT, IP1),              the init., ANY IP for
        OADD(INIT, IP2),              the resp.
        OADD(RES, ANY_IP))
    Proposal(ENCR, INTEG, ESN))) < proposes to use IKEv2 IP
  }                                for the VPN outer IP
    -->

    <-- HDR, SK {IDr, [CERT,] AUTH, TSi, TSr,
        SAR2(Proposal(ENCR, INTEG, ESN,
            OADD(INIT, IP1),
            OADD(RES, IPr)))
        }

```

Figure 5: Basic Exchange for VPN alternate outer IP addresses negotiation

6. Payload Formats

As mentioned in Section 5 this document introduces a new Transform of Transform Type OADD. The associated Transform ID are INIT for the initiator outer IP address and RESP for the responder's IP address. These Transforms are associated a Attributes that are either carrying an IP address (IPv4 or IPv6) or associated to a specific value like ANY_IP.

This document also introduces the ALTERNATE_OUTER_IP_ADDRESS_SUPPORTED Notify Payload, so peers can

inform the other they support the alternate outer IP address extension.

This section describes the format of all new payload introduced for the outer IP address extension.

6.1. Outer IP address Transform OADD

This section specifies the Transform structure as defined in [RFC5996] Section 3.3.2.

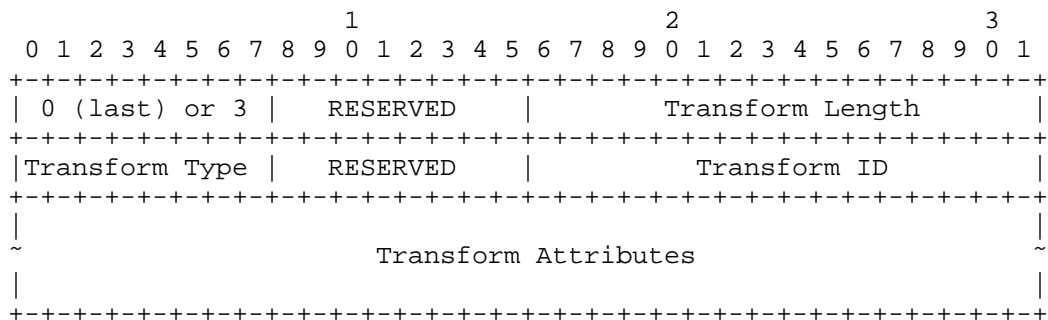


Figure 6: OADD Transform Substructure

- 0 (last) or 3 (more) (1 octet): Specifies whether this is the last Transform Substructure in the Proposal.
- RESERVED (1 octet): MUST be sent as zero; MUST be ignored on receipt.
- Transform Length (2 octets): The length (in octets) of the Transform Substructure including Header and Attributes.
- Transform Type (2 octets): The type of transform being specified in this transform. Set to OADD in this document.
- Transform ID (2 octets): The specific instance of the Transform Type being proposed. Set to INIT or RESP in this document.
- Transform Attributes (variable length): The IP Attribute in this document.

6.2. IP Attribute with IP addresses

This section specifies the Attribute structure as defined in [RFC5996] Section 3.3.5.

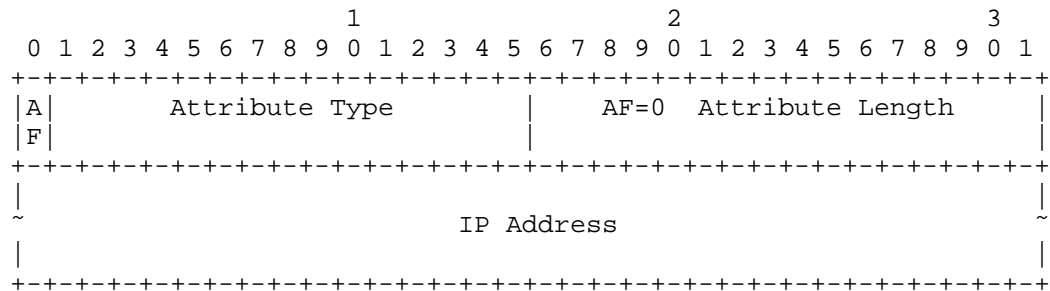


Figure 7: IP Attribute with IP address

- Attribute Format (AF) (1 bit): Set to 0, indicating a TLV format.
- Attribute Type (15 bits): Set to IP in this document.
- Attribute Length (16 bits): The length is either 8 to designate the length of an IPv4 or 20 to designate the length of an IPv6 address. The length includes the headers of 4 octets.

6.3. IP Attribute indicating ANY_IP

This section specifies the Attribute structure as defined in [RFC5996] Section 3.3.5.

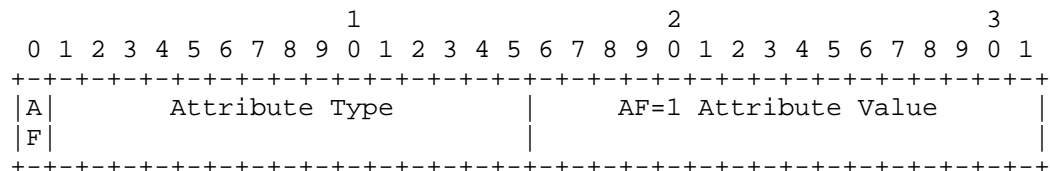


Figure 8: IP Attribute set to ANY_IP

- Attribute Format (AF) (1 bit): Set to 1, Attribute Value.
- Attribute Value (15 bits): Set to ANY_IP in this document.

6.4. Alternate Outer IP Address Notify Payload

This section presents the Notify Payload as defined in [RFC5996] Section 3.10.

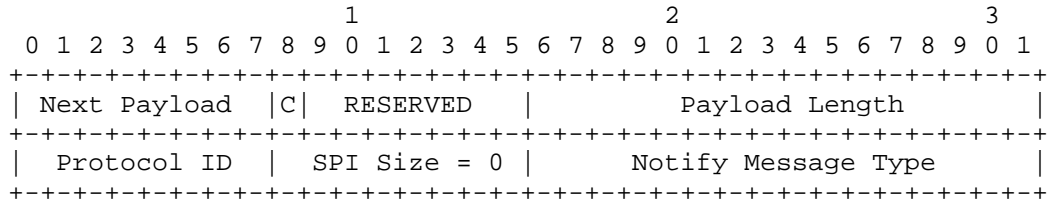


Figure 9: Alternate Outer IP Address Notify Payload

- Next Payload (1 octet): Identifier for the payload type of the next payload in the message. If the current payload is the last in the message, then this field will be 0.
- Critical (1 bit): MUST be set to zero for payload types defined in this document.
- RESERVED (7 bits): MUST be sent as zero; MUST be ignored on receipt.
- Payload Length (2 octets, unsigned integer): Length in octets of the current payload, including the generic payload header. Set to 16 in this document.
- Protocol ID (1 octet): This field MUST be sent as zero and MUST be ignored on receipt.
- Notify Message Type (2 octets): Set to ALTERNATE_OUTER_IP_ADDRESS_SUPPORTED in this document.

7. NAT considerations

In the document we assumed that there were no NAT between the VPN End User and the Security Gateway. This means that the VPN End User and the Security Gateway know 1) the interface they are receiving data on is the interface used as a destination by the other peer and 2) the interface set as destination is the interface used by the other peer to receive the data. As a result, if the VPN End User (respectively the Security Gateway) is behind a NAT, the VPN End User may be seen by the Security Gateway (respectively the VPN End User) with another IP address unknown to the VPN End User (respectively the Security Gateway).

NATs impact the alternate outer IP address extensions in two ways:

- IPsec configuration: The alternate outer IP addresses the two peers are negotiating may not be the ones in the Security Associations. More specifically, suppose the VPN End User and the Security Gateway depicted in figure 10 have negotiated the alternate outer IP addresses `src_0`, `dst_1`. `src_0` is NATted with `NAT_0`, and may be unreachable, the outer IP address in the Security Gateway SA should be `src_nat_0` instead.
- NAT traversal: NATs may make an IP address behind it reachable only if this IP address has initiated a connection. More specifically, suppose the VPN End User and the Security Gateway depicted in figure 10 have established an IKEv2 channel between `src_0` and `dst_1` and are MOBIKE enabled. Suppose the VPN End User sends the Security Gateway an `ADDITIONAL_IP*_ADDRESS` with `src_1` or eventually with `src_nat_1`. Unless `NAT_1` has been configured to forward the traffic from the Security Gateway to the VPN End User, this traffic will most likely be discarded by `NAT_1`. Similarly, if the Security Gateway moves the VPN from `dst_0` to `dst_1`, the VPN may be broken. Note that we use MOBIKE to illustrate the problems of reachability through NATs, but these operations are discussed more in depth in [RFC4555].

This section does not intend to discuss all NATs configuration as described in [RFC5389]. Instead the only NAT scenario we consider is a single NAT and the VPN End User behind that NAT initiates the alternate outer IP address exchange. The architecture this section considers is depicted in figure 10. Furthermore, this section does not consider the NAT traversal aspect. We assume that the VPN End User is NAT aware and perform the necessary actions to make/configure the NATs so that they do not block the traffic.

Section 7.1 defines how the End User MAY prohibit the alternate outer IP address extension if a NAT is detected. Then, in Section 7.2 how the VPN End User can detect the presence of NAT. Section 7.3 discusses the case where the VPN End User does not know the values of the NATted IP addresses and Section 7.4 discusses the case where the VPN End User knows all NATted IP addresses values.

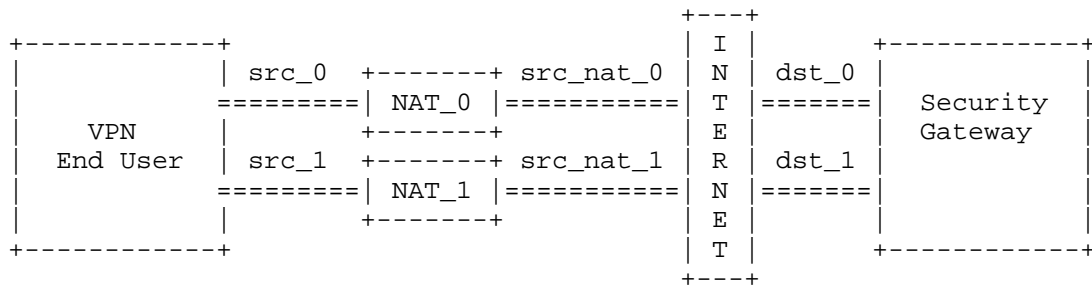


Figure 10: VPN End User behind a NAT scenario

7.1. Prohibiting NAT

This section considers that the VPN End User does not want to use the alternate outer IP address extension if a NAT is detected. This section differs from the NAT detection because it both detects the existence of a NAT and provide an indication that some supported functionalities like MOBIKE SHOULD NOT be used if a NAT is detected.

The NO_NATS_ALLOWED Notify Payload is defined in [RFC4555]. If the VPN End User supports MOBIKE, it MAY send a NO_NATS_ALLOWED Notify Payload with the original IP addresses and ports. When the Notify Payload is received by the Security Gateway, it checks the IP addresses values in the IP header and in the Payload, in case of mismatch, a UNEXPECTED_NAT_DETECTED Notify Payload is returned.

In our case, the NO_NATS_ALLOWED MAY be used by the VPN End User if both the VPN End User and the Security Gateway support MOBIKE. When the Security Gateway receives the NO_NATS_ALLOWED Notify Payload, it MUST NOT use MOBIKE and SHOULD NOT use the alternate outer IP address extension.

There are corner cases that are not considered by this policy. First, a VPN End User or a Security Gateway that do not support MOBIKE cannot use the NO_NATS_ALLOWED Notify Payload. However, it seems hardly possible that peers supporting the alternate outer IP address extension support MOBIKE. Second, a VPN End User using the NO_NATS_ALLOWED applies the same policy for MOBIKE and the alternate outer address extension. Here again, it seems unlikely that NAT policies differ. Furthermore, the NO_NATS_ALLOWED exchange only prevent the Security Gateway to initiate a MOBIKE or alternate outer IP address negotiation. The VPN End User can still use one or the other extension. From our experience, this constraint seems acceptable.

7.2. NAT detection

This section details how NAT can be detailed with IKEv2 extensions. We do not consider here other mechanisms like ICE described in [RFC5768] or STUN [RFC5389].

The VPN End User can detect the NAT by using the NAT_DETECTION_SOURCE_IP and NAT_DETECTION_DESTINATION_IP Notify Payload as described in [RFC5996]. These Notify Payloads carry the SHA-1 of the source (respectively the destination) IP address. At the reception, the Security Gateway can compare their content with the SHA-1 of the IP addresses in the IP header. A mismatch between the two values indicates the presence of a NAT, but do not provide the value of the original IP address. Usually, this exchange is performed during the IKE_INIT exchange to decide whether or not IKEv2 should proceed to UDP encapsulation.

Note that with the NAT detection exchange, the NAT is detected on the IKEv2 channel. If the IKEv2 channel is using src_0, the NAT detection exchange will detect NAT_0. To detect NAT_1 using IKEv2, the VPN End User SHOULD move the IKEv2 channel on src_1 with MOBIKE for example. Since the UPDATE_SA Notify Payload is initiated by the VPN End User, NAT_1 is expected to accept the traffic from the Security Gateway. Note also that the NAT detection exchange does not provide the value of the src_nat* IP addresses.

7.3. The VPN End User does not know the NATted IP addresses

This section analyses how the alternate outer IP address extension can be used when the VPN End User does not know the values of the NATted IP addresses, i.e. src_nat_0 and src_nat_1.

In that case, the VPN End User MAY only select the destination outer IP address corresponding to the Security Gateway IP addresses. How the VPN End User gets these IP addresses is out of scope of the document, however, if the VPN End User and the Security Gateway support MOBIKE, the MOBIKE ADDITIONAL_IP*_ADDRESS Notify Payload MAY be used for that purpose. It is recommended that the VPN End User does not provide the outer source IP, in which case, the one from the IKEv2 channel will be considered by default. More specifically, the VPN End User cannot provide the Security Gateway its alternate IP addresses.

The VPN End User MAY use the ANY_IP IP Attribute for the source outer IP address. This would enable the Security Gateway to select an alternate IP address that differs from the one used by the IKEv2 channel. In order to select the IP addresses associated to the VPN End User, the Security Gateway has to be aware of the NATted IP

addresses depicted as src_nat_0 and src_nat_1. One possibility is that the Security Gateway log the IP addresses used by the VPN End User when it moves from src_0 to src_1. This also means that the VPN is being negotiated with a CREATE_CHILD_SA exchange after the initial IKE_INIT exchange.

7.4. The VPN End User does know the NATted IP addresses

In this section the VPN End User knows the NATted IP addresses src_nat_0 and src_nat_1. How the End User get these values is out of scope of the document. This case should be considered only if the VPN End User exactly know what it is doing.

In this case, the VPN End User can proceed as if no NAT exist. The VPN End User considers in the alternate outer IP address negotiation that its IP addresses are the NATted IP addresses that is src_nat_0 and src_nat_1. On the other hand, the VPN End User MUST configure properly its SAs with src_0 if src_nat_0 is selected or with src_1 if src_nat_1 is selected.

The VPN End User is also responsible to make the NAT Traversal possible.

8. IANA Considerations

The new fields and number are the following:

IKEv2 Notify Message Types - Status Types

ALTERNATE_OUTER_IP_ADDRESS_SUPPORTED TBD

Transform Attribute Types

OADD TBD

Transform Type OADD IDs

INIT TBD
RESP TBD

Attribute Type

IP TBD

IP Attribute Type Values

ANY_IP

TBD

9. Security Considerations

The exchange described in this document is protected by the IKEv2 channel.

10. Acknowledgment

The author would like to thank Yoav Nir for its helpful comments.

11. References

11.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.
- [RFC4555] Eronen, P., "IKEv2 Mobility and Multihoming Protocol (MOBIKE)", RFC 4555, June 2006.
- [RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.
- [RFC5768] Rosenberg, J., "Indicating Support for Interactive Connectivity Establishment (ICE) in the Session Initiation Protocol (SIP)", RFC 5768, April 2010.
- [RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.
- [RFC6027] Nir, Y., "IPsec Cluster Problem Statement", RFC 6027, October 2010.

11.2. Informational References

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

[I-D.mglt-ipsecme-security-gateway-discovery]
Migault, D. and K. Pentikousis, "IKEv2 Security Gateway
Discovery",
draft-mglt-ipsecme-security-gateway-discovery-00 (work in
progress), February 2013.

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

Appendix A. Document Change Log

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

-00: First version published.

Author's Address

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

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

IPSECME
Internet-Draft
Intended status: Standards Track
Expires: August 18, 2013

D. Migault (Ed)
Francetelecom - Orange
K. Pentikousis
Huawei Technologies
February 14, 2013

IKEv2 Security Gateway Discovery
draft-mglt-ipsecme-security-gateway-discovery-00.txt

Abstract

Modern Virtual Private Network (VPN) services are typically deployed using several security gateways and are frequently accessed over a wireless network. There are several reasons for such a deployment ranging from enhancing system resilience to improving performance.

For example, in order to handle traffic efficiently and reduce the burden in the core network, the VPN service may be implemented in a distributed manner using multiple Security Gateways. A mobile VPN End User is attached to one of them using a WLAN interface and over time is likely to change its Security Gateway of attachment. In this case, in order to optimize the overall user Quality of Experience (QoE), a VPN End User should select the next most appropriate Security Gateway based on the characteristics of the available Security Gateways. This draft specifies how a VPN End User can securely collect information about Security Gateways in its network neighborhood in order to optimize its VPN experience.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on August 18, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Requirements notation	4
2. Introduction	4
3. Terminology	4
4. Motivation	5
4.1. Multiple Interfaces	5
4.2. Closest Next Neighbor	6
4.3. Intra-Security Gateway Services	7
4.4. Why We Cannot Rely On DNS Only	7
5. Security Gateway Discovery Protocol	8
5.1. Sending a NEIGHBOR_INFORMATION Query	8
5.2. Receiving NEIGHBOR_INFORMATION	9
5.2.1. NEIGHBOR_INFORMATION Query Processing	10
5.2.2. NEIGHBOR_INFORMATION Response Processing	10
5.2.3. Informative NEIGHBOR_INFORMATION	11
6. Notify Payload Format	11
6.1. NEIGHBOR_INFORMATION Notify Payload	11
6.2. Initiator Options: O-REQUEST	12
6.3. Responder Options	13
6.3.1. Neighbor: NEIGHBOR	13
6.3.2. Interface Option: O_INTERFACE	13
6.3.3. Geo-localization Option: O_GEOLOC	14
6.3.4. Intra-Security Gateway Bandwidth Option: O_ISG-BW	14
6.3.5. Intra-Security Gateway Mobility Support Option: O_ISG-MOB	15
6.4. General Options	15
6.4.1. Padding Payload: PADDING	16
6.4.2. Maximum Neighbors Payload: MAX-NEIGHBOR	16
7. IANA Considerations	17
8. Security Considerations	17
9. Acknowledgments	17
10. References	18
10.1. Normative References	18
10.2. Informative References	18
Appendix A. Document Change Log	18
Authors' Addresses	18

1. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

2. Introduction

When a Virtual Private Network (VPN) client establishes a VPN connection with a distributed VPN infrastructure, care should be taken to choose the most appropriate Security Gateway. DNS may be considered as a selection mechanism to determine the first point of attachment to the distributed VPN infrastructure. However, as we explain later in this document, the information provided by DNS is limited and insufficient for this purpose. In effect, the VPN End User cannot rely on this information to optimize its point of attachment. Moreover, for the case of mobile nodes, such information cannot help in the case of multiple interface communication nor properly handle VPN mobility from one Security Gateway to another. This document addresses this problem by describing how a VPN End User can request from its Security Gateway information about other neighbor Security Gateways. Equipped with this knowledge the VPN End User can select the most appropriate Security Gateway.

The remainder of this document is organized as follows. Section 3 defines the terms and acronyms used in this document. Section 4 introduces scenarios that relate to Security Gateway selection. For each scenario, specific criteria are used by the VPN End User to select the most appropriate Security Gateway. Section 5 and Section 6 specify the Security Gateway Discovery Protocol introduced in this document, including defining the packet exchanges and the corresponding involved payloads, respectively.

3. Terminology

This section defines the terms and acronyms used in this document.

- VPN End User (EU): designates the entity that initiates a VPN connection with a Security Gateway. A VPN End User may be mobile and, as per this document, can change its VPN connection from one Security Gateway to another.
- Security Gateway: designates the network point of attachment for the VPN service. In this document, the VPN service can be provided by multiple Security Gateways. Each Security Gateway may be considered as a specific logical or physical network

entity.

- VPN service: designates the service provided to the End User.
From the end-user point of view, in colloquial terms, this is what typical users consider as "establishing a VPN connection".

Throughout the document we assume that the user is not interested and, therefore, is not informed about which Security Gateway is chosen. We consider that mobility, both in terms of network point of attachment and the Security Gateway used for the VPN service, is handled inherently by the network and the user is not concerned about the actual operational details.

4. Motivation

This section motivates the technical solution advocated in this document by presenting three scenarios where the selection of the Security Gateway can significantly improve the Quality of Experience (QoE) of a VPN End User. For each scenario, we describe the information that the VPN End User needs in order to select the appropriate Security Gateway.

4.1. Multiple Interfaces

Multiple interfaces on the VPN End User or on the Security Gateway make possible path selection. If the VPN End User is able to perform path selection, it is likely to choose a Security Gateway that has multiple interfaces. Between multiple Security Gateways with multiple interfaces it may choose the one whose interfaces are attached to its preferred networks. This Security Gateway selection is particularly important since VPN End User can hardly split their VPN on two distinct Security Gateways.

Distributed VPN infrastructures are composed of multiple, independent Security Gateways. Currently, IPsec [RFC4301] does not have the mechanisms that enable "moving" a VPN connection from one Security Gateway to another Security Gateway. In practice, this means that moving the endpoint of a VPN connection from one Security Gateway to another requires a renegotiation establishment of a new VPN. This may also include new authentication for the VPN End User, likely with the need for user input in the process. On the other hand, MOBIKE [RFC4555] enables moving a VPN connection from one interface to another as long as they are attached to the same Security Gateway. Thus, we have two ways with different impact on the corresponding end user Quality of Experience (QoE), to move a VPN connection from one interface to another depending on whether these interfaces belong to the same node or not. As a result, a client implementing the MOBIKE

extension can perform interface management, and opt to be attached to a Security Gateway with multiple interfaces.

Note that with IPsec [RFC4301], the signaling channel is defined by the IKE_SA while the user data is designated by the IPsec_SA. Unless specifically designed otherwise, these two channels are highly dependent on each other and MUST be hosted on the same host. More specifically, it is not possible for a VPN End User to have its IKE channel with one host and its IPsec_SA with a different, independent host.

Note also that MOBIKE enables a Security Gateway to inform a VPN End User about its available interfaces. However, these interfaces belongs to the Security Gateway the VPN End User is attached to, not another Security Gateway.

This document defines how a VPN End User can query a Security Gateway in a distributed VPN infrastructure whether other, neighboring Security Gateway have one or multiple interfaces. In this document we are concerned about the other Security Gateways so that the VPN End User can decide which Security Gateway it should be attached to next.

4.2. Closest Next Neighbor

With a large distributed VPN infrastructure like those serving xDSL broadband networks, a mobile VPN End User needs to define which Security Gateway it will be attached to next. The current Security Gateway can assist a VPN End User to avoid spending effort on Security Gateway discovery by providing this localization information. This is beneficial both in terms of network bandwidth and system resources.

Localization may be based on geo-localization data. Nevertheless, in many cases, the optimal Security Gateway for each particular VPN End User may not be the one that is closer in geographical terms, but the one with the best inter-Security Gateway bandwidth. In fact, in recent distributed mobility architectures, DSL boxes in a typical urban environment exchange information using their WLAN interface to avoid congesting the core network.

We argue that if Security Gateways can exchange information they can improve VPN client mobility and reduce traffic overhead. Such information may include, for instance, VPN client authentication credentials, IPsec counters, or packet redirection. Using this information-exchange protocol, the VPN End User has, for example, the advantage of moving to the DSL box with the best inter-Security-Gateway bandwidth.

4.3. Intra-Security Gateway Services

Although currently IPsec does not enable a VPN client to move from one Security Gateway to another one, proprietary protocols that enable such mobility from one Security Gateway to another do exist. This may, for example, involve exchange of IPsec counters. This information may help the VPN End User to properly chose the next Security Gateway it will be attached to. Standardizing the way this information is exchanged can benefit end users and network operators alike.

4.4. Why We Cannot Rely On DNS Only

DNS binds a FQDN to one or multiple IP addresses. In that sense, one may consider that DNS could be leveraged upon to provide information sufficient to determine the neighboring Security Gateways. Unfortunately, this is not the case because FQDN is an abstraction, and in our case, the FQDN most probably designates the name of the VPN service as a whole. Thus, DNS is used to bind the VPN service with specific interfaces, without specifying which Security Gateway they belong to. Since this information is not available, the VPN End User cannot select a specific Security Gateway, as two issues arise as we explain next.

First, DNS can provide a list of multiple interfaces available for a given service (i.e. FQDN), which enables a client to choose the most appropriate interface at the moment in time that it initiates a VPN service. Once connected to one of the Security Gateways, MOBIKE makes possible to convey to the VPN End User the available interfaces on the Security Gateway that the client is attached to. In principle, the VPN End User could then use the list of interfaces provided by DNS, correlate it with that received via MOBIKE and come to some conclusion with respect to Security Gateway availability. Besides the fact that this method is inexact science at best, it does not add much value in large deployments. Since each Security Gateway may have multiple interfaces, it has no clue if the remaining interfaces belong to a single Security Gateway or to multiple Security Gateways. This information cannot be provided by DNS. This motivates us to provide this information at the service layer, that is to say, for the VPN service, via IKEv2.

Second, DNS usually does not provide the complete list of all Security Gateway interfaces, but often just a subset of those available by the VPN service. For largely distributed applications, DNS provides a subset of available interfaces that are "close" to the resolving server. The problem with this is that DNS can hardly provide the "closest" server to the VPN End User. Firstly, defining the closest interface of the DNS query emitter remains difficult.

Secondly, it is impossible to consider the various interfaces of the VPN End User. Thirdly, the DNS query is usually sent by a resolving server, not by the VPN End User. Because of this indeterminacy, DNS may be more concerned about avoiding the worst answer, rather than looking for the best option. Thus, it may look for answers with a large diversity instead of focusing their answers to a given location. Among the proposed interfaces, the VPN End User may chose the most convenient interface according to its policy or its interfaces.

Note that [I-D.vandergaast-edns-client-ip] makes possible to avoid considering the resolving server location instead of the VPN client.

5. Security Gateway Discovery Protocol

In this document we assume that the VPN End User is already attached to a Security Gateway. The goal of this exchange is that the VPN End User can obtain information about other Security Gateways which are designated as neighbors.

The proposed Security Gateway Discovery Protocol (SGDP) employs a query / response exchange mechanism. Usually, the exchange is initiated by the VPN End User and the responder is the Security Gateway that the VPN End User is connected to. However, the protocol does not exclude that either of the peers initiates the exchange.

5.1. Sending a NEIGHBOR_INFORMATION Query

The initiator builds the NEIGHBOR_INFORMATION Notify Payload (described in Section 6.1) by setting the Question bit to 1 and providing the necessary Options. Notify Payloads have a Critical bit set.

The Option request Option (described in Section 6.2) makes possible to list the queried information about each neighboring Security Gateway. In this document, the Options that can be queried are:

- Interface Option: lists the interfaces associated to the neighboring Security Gateway.
- Geo-localization Option: provides geographic coordinates of the neighboring Security Gateway.
- Intra-Security Gateway Bandwidth Option: indicates how much bandwidth the current Security Gateway shares with the neighboring Security Gateway.

- Intra-Security Gateway Mobility Support Option: indicates if the current Security Gateway and the neighboring Security Gateway share a specific mobility protocol to ease moving the VPN connection from the current Security Gateway to the neighboring Security Gateway.

The Maximum Neighbor Option is intended to limit the size of the response and indicates how many neighboring Security Gateway SHOULD be considered. Finally, the Padding Payload format pads the overall Notify Payload to a length that is a multiple of 32 bits. Other Options may be added for future use.

5.2. Receiving NEIGHBOR_INFORMATION

A received NEIGHBOR_INFORMATION Notify Payload may be originating from a query by the initiator as described in Section 5.1. This case is detailed in Section 5.2.1, below. Alternatively, the incoming message may be a response to a query previously sent by the VPN connection peer, which is detailed in Section 5.2.2. The protocol also supports informative messages as detailed in Section 5.2.3. Finally, the received NEIGHBOR_INFORMATION Notify Payload may be an unwanted message.

Once a NEIGHBOR_INFORMATION Notify Payload is received, the responder checks whether the Critical bit is set to 1. If the Critical Bit is set and the Notify Payload is not supported by the responder then, following [RFC5996] section 2.5, setting the Critical bit to one forces the Responder to send back a UNSUPPORTED_CRITICAL_PAYLOAD Notify Payload if it does not understand the received Notify Payload.

If the Critical bit is set, and the receiver supports the NEIGHBOR_INFORMATION Notify Payload, the receiver checks the Question Bit. A set Question Bit means that the Notify Payload is a query as described in Section 5.1, and a response MUST be formed and sent back to the initiator. This is described in Section 5.2.1. If the Question Bit is not set, then the Notify Payload corresponds to a response. If no corresponding query has been sent previously an INVALID_SYNTAX MUST be sent back and the rest of the Notify Payload MUST be ignored. Conversely, if a query has been sent, the receiver will process the response as per Section 5.2.2.

If the Critical bit is not set and the Notify Payload is not supported by the receiver, the Notify Payload MUST be ignored. However, this case is expected to only occur for informative NEIGHBOR_INFORMATION Notify Payload as described in Section 5.2.3.

If the Critical Bit is not set and the receiver supports the NEIGHBOR_INFORMATION Notify Payload, then the receiver examines the

Question Bit. If it is set, the message MUST be ignored. This is to avoid ambiguity in cases where the initiator does not know if it receives no response because there is no information or because the Notify Payload is not supported by the responder. If the Question Bit is not set, the Notify Payload corresponds to an informative NEIGHBOR_INFORMATION Notify Payload. This case is detailed in Section 5.2.3.

5.2.1. NEIGHBOR_INFORMATION Query Processing

For this section we assume that the Critical Bit and the Question Bit are set, the Notify Payload is properly formed and the receiver understands the NEIGHBOR_INFORMATION Notify Payload.

The responder checks if a Maximum Neighbor Option is in the query. If not present, the responder is allowed to provide as much Neighbor Payload information as deemed best. If the option is present, then the responder SHOULD check its internal policy and determine how many Neighbor Payload can be provided in the response. If the limit set by the internal policy is lower than what is requested by the initiator in the Maximum Neighbor Option, the responder MUST indicate it by providing a Maximum Neighbor Option that corresponds to the actual number of Neighbor Payloads.

The responder checks if a Option request Option is in the query. If not, the responder MAY use its default policy about the default Options to be returned. It MAY also return a void response. In any other case, the responder lists the queried Options. For each Neighbor, if the responder has the queried information, it MUST indicate it in the Neighbor Payload.

The Padding Option is used to properly format the response, and the response is sent to the initiator.

5.2.2. NEIGHBOR_INFORMATION Response Processing

This section assumes that the Critical Bit is set and the Question Bit is not set, the Notify Payload is properly formed and the receiver understands the NEIGHBOR_INFORMATION Notify Payload.

If a Maximum Neighbor Option is present, this means that only a subset of the available information has been sent. If no Maximum Neighbor Option has been sent in the query, the number received indicates an internal policy of the responder. On the other hand, if a Maximum Neighbor Option has been sent in the query, a number equal to the one specified in the query is expected. Other values indicate an internal policy of the responder.

5.2.3. Informative NEIGHBOR_INFORMATION

The VPN connection peer may provide informative NEIGHBOR_INFORMATION without being queried. This is the case when the Critical Bit and the Question Bit are not set, the Notify Payload is properly formed and the receiver understands the NEIGHBOR_INFORMATION Notify Payload.

6. Notify Payload Format

This section introduces the Notify Payload for the Security Gateway Discovery Protocol.

6.1. NEIGHBOR_INFORMATION Notify Payload

Fig. 1 illustrates the NEIGHBOR_INFORMATION Notify Payload packet format.

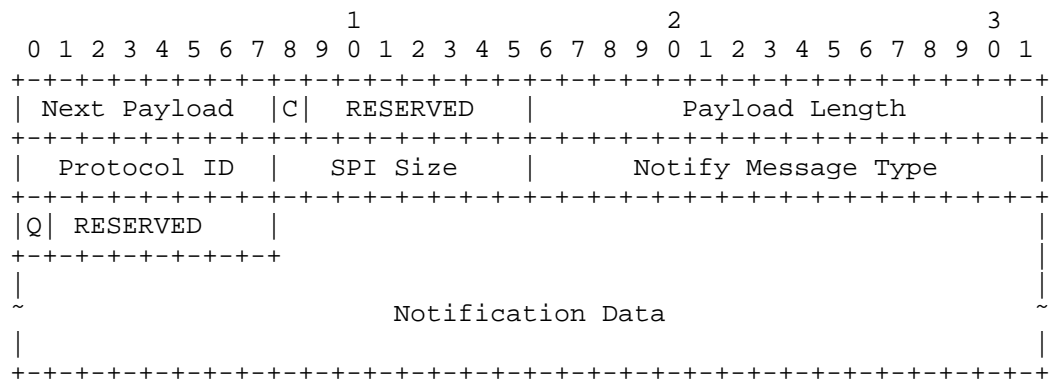


Figure 1: NEIGHBOR_INFORMATION Notify Payload

- Next Payload (1 octet): Indicates the type of payload that follows after the header.
- Critical Bit (1 bit): Indicates how the responder handles the Notify Payload. In this document the Critical Bit is not set only when an informative NEIGHBOR_INFORMATION is sent. Otherwise, the Critical bit is set to 1.
- RESERVED (7 bits): MUST be sen as zero; MUST be ignored on receipt.

- Payload Length (2 octet): Length in octets of the current payload, including the generic payload header.
- Protocol ID (1 octet): set to zero.
- SPI Size (1 octet): set to zero.
- Notify Message Type (2 octets): Specifies the type of notification message NEIGHBOR_INFORMATION_QUERY
- Question Bit (1 bit): set to one by the initiator and set to zero by the responder.
- RESERVED (7 bits): set to zero.
- Notification Data (variable length): When the Notify Payload is sent by the initiator, the Notification data is composed of Parameters.

6.2. Initiator Options: O-REQUEST

This section provides the parameters that comprise the Notification Data of the initiator.

The Option Request Payload defines the Options requested for each neighbor. In other words, it is expected in the response that each Neighbor Payload (NEIGHBOR) Section 6.3.1 is filled with these Options.

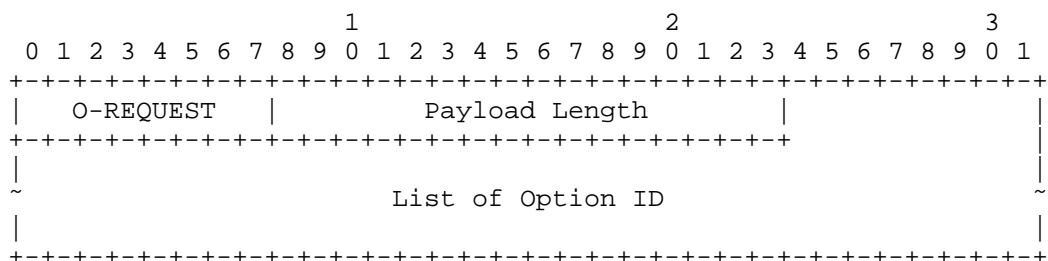


Figure 2: Option Request Option: O-REQUEST

- Option-ID (1 octet): O-REQUEST
- Payload Length (2 octet): Payload Length expressed in octet and includes the Option-ID and Payload Length fields' length. The Payload may not be a multiple of 32 bytes.

- List of Option ID (variable length): List of the Option that are expected for each NEIGHBOR Payload.

6.3. Responder Options

6.3.1. Neighbor: NEIGHBOR

The Neighbor Payload contains information about a neighbor Security Gateway. The number of Neighbor Payloads is defined by the Maximum Neighbors Payload, or if not specified by the responder. If the number of Neighbor Payloads is defined by the responder, the responder MUST add the Maximum Neighbors Payload.

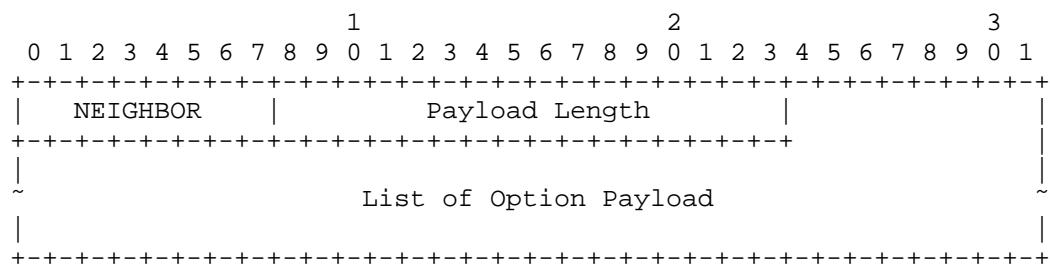


Figure 3: Neighbor: NEIGHBOR

- Option-ID (1 octet): NEIGHBOR
- Payload Length (2 octet): Payload Length expressed in octets, including the Option-ID and Payload Length fields' length. The Payload may not be a multiple of 32 bytes.
- List of Option Payload (variable length): List of the Option Payload requested by the initiator.

6.3.2. Interface Option: O_INTERFACE

The Interface Option provides the IP addresses of the Neighbor.

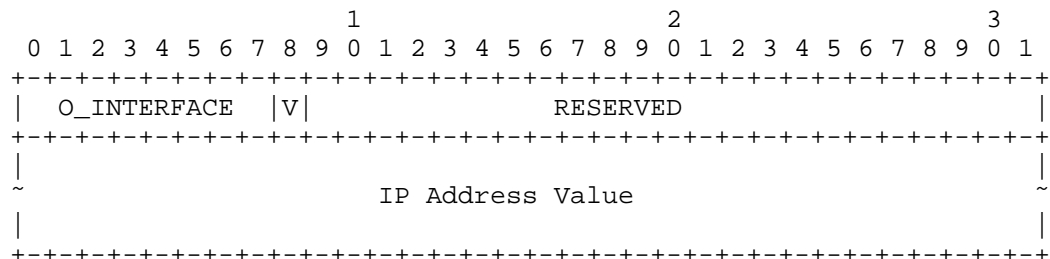


Figure 4: Interface Option: O_INTERFACE

- Option-ID (1 octet): O_INTERFACE
- Version Bit (1 bit): The Version Bit indicates if the IP address is an IPv4 or an IPv6 IP address. The Version Bit is set to 1 for an IPv4 address.
- RESERVED (23 bits): Set to Zero.
- IP Address Value (4 or 16 octets): The IP address value. An IPv4 address is 4 octet long and an IPv6 address is 16 octets long.

6.3.3. Geo-localization Option: O_GEOLOC

The Geo-localization Option provides Geographic coordinates of the Neighbor.

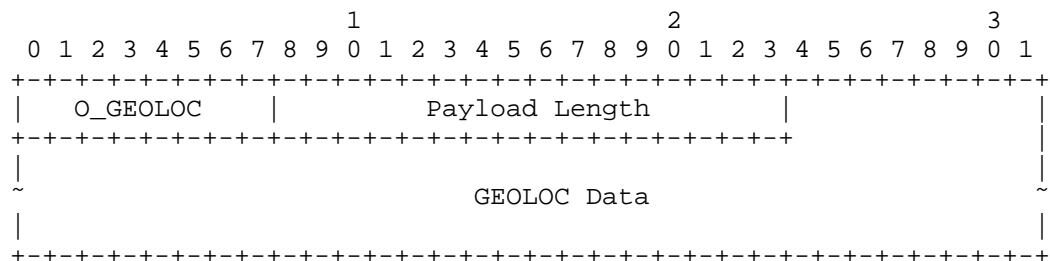


Figure 5: Geo-localization Option: O_GEOLOC

- Option-ID (1 octet): O_GEOLOC
- Payload Length (2 octet): Payload Length expressed in octets including the Option-ID and Payload Length fields' length. The Payload may not be a multiple of 32 bytes.
- GEOLOC Data (variable length): GEOLOC Data as defined in [RFC1876].

6.3.4. Intra-Security Gateway Bandwidth Option: O_ISG-BW

The Intra-Security Gateway Bandwidth Option characterizes the link between the responder and the Neighbor.

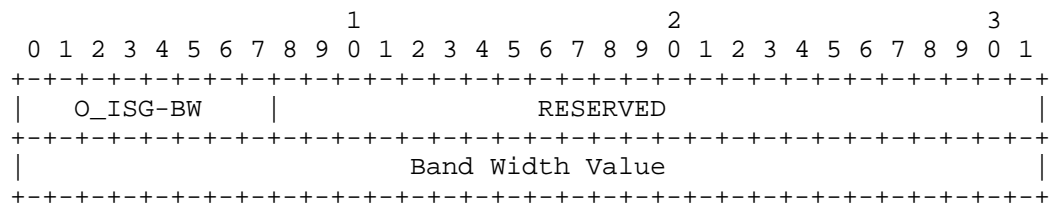


Figure 6: Intra-Security Gateway Bandwidth Option: O_ISG-BW

- Option-ID (1 octet): O_ISG-BW
- RESERVED (3 octets): Set to Zero.
- Band Width Value (4 octets): Specifies the bandwidth in octets per second.

6.3.5. Intra-Security Gateway Mobility Support Option: O_ISG-MOB

The Intra-Security Gateway Mobility Option defines if there are any mechanisms that support VPN mobility from the responder to the Neighbor.

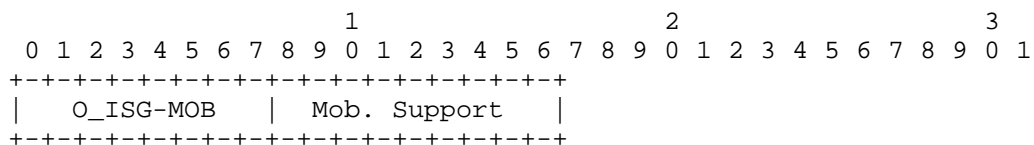


Figure 7: Intra-Security Gateway Mobility Support Option: O_ISG-MOB

- Option-ID (1 octet): O_ISG-MOB
- Mobility Support (1 octet): Specifies how VPN mobility is supported from the responder to the Neighbor.

Currently the following values are provided for Mobility Support:

- ```
- UNSUPPORTED_MOBILITY: 0
- IPSEC_CONTEXT_TRANSFERED: 1
```

## 6.4. General Options

This section describes two options that can be used by both the initiator and the responder.



#### 6.4.1. Padding Payload: PADDING

The Padding Payload is used to make the NEIGHBOR\_INFORMATION Notify Payload length a multiple of 32 bits.

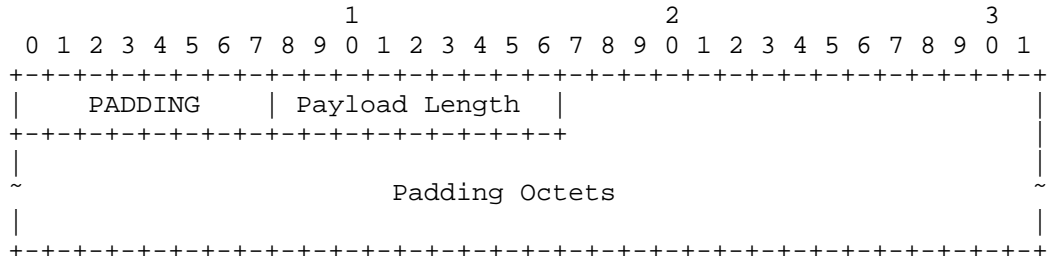


Figure 8: Padding Payload: PADDING

- Option-ID (1 octet): PADDING
- Payload Length (1 octet): Payload Length expressed in octet and includes the Option-ID and Payload Length fields' length. In case one need 2 octet padding, the Payload Length is set to 2. If there is only a need for a 1 octet padding, then 4 additional padding octets must be added and the Payload Length is set to 5.
- Padding Octets (variable length): These Octets are for padding and MUST NOT be interpreted.

#### 6.4.2. Maximum Neighbors Payload: MAX-NEIGHBOR

The Maximum Neighbors Payload sets the maximum number of Neighbor the VPN End User wants information about. This Option is of fixed size.

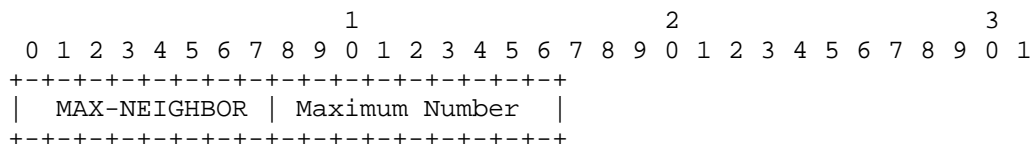


Figure 9: Maximum Neighbors Payload: MAX-NEIGHBOR

- Option-ID (1 octet): MAX-NEIGHBOR
- Maximum Number (1 octet): Specifies the maximum number of NEIGHBOR Payload the response carries.

## 7. IANA Considerations

The new fields and number are the following:

IKEv2 Notify Message Types - Status Types

-----  
NEIGHBOR\_INFORMATION

Security Gateway Discovery Attributes

-----  
O-REQUEST  
PADDING  
MAX-NEIGHBOR  
NEIGHBOR

Neighbor Options

-----  
O\_INTERFACE  
O\_GEOLOC  
O\_ISG-BW  
O\_ISG-MOB

O\_ISG-MOB Attributes

-----  
UNSUPPORTED\_MOBILITY  
IPSEC\_CONTEXT\_TRANSFERRED

## 8. Security Considerations

The exchange described in this document is protected by the IKEv2 channel. Then, the only concern may be the information that a Security Gateway provides to the VPN End User. We do not see how the provided information can be used against the Security Gateway. Furthermore, the VPN End User has already been authenticated by IKEv2 prior to being able to obtain such information.

## 9. Acknowledgments

TBD

## 10. References

## 10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.
- [RFC4555] Eronen, P., "IKEv2 Mobility and Multihoming Protocol (MOBIKE)", RFC 4555, June 2006.
- [RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

## 10.2. Informative References

- [I-D.vandergaast-edns-client-ip]  
Contavalli, C., Gaast, W., Leach, S., and D. Rodden,  
"Client IP information in DNS requests",  
draft-vandergaast-edns-client-ip-01 (work in progress),  
May 2010.
- [RFC1876] Davis, C., Vixie, P., Goodwin, T., and I. Dickinson, "A Means for Expressing Location Information in the Domain Name System", RFC 1876, January 1996.

## Appendix A. Document Change Log

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

-00: First version published.

## Authors' Addresses

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

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

Kostas Pentikousis  
Huawei Technologies  
Carnotstrasse 4  
10587 Berlin  
Germany

Email: k.pentikousis@huawei.com



IPSecME Working Group  
Internet-Draft  
Intended Status: Informational  
Expires: March 27, 2016

A. Yamaya  
Furukawa Network Solution  
T. Ohya  
NTT  
T. Yamagata  
KDDI  
S. Matsushima  
Softbank Telecom  
September 24, 2015

Simple VPN solution using Multi-point Security Association  
draft-yamaya-ipsecme-mps-a-06

Abstract

This document describes the over-lay network solution by utilizing dynamically established IPsec multi-point Security Association (SA) without individual connection.

Multi-point SA technology provides the simplified mechanism of the Auto Discovery and Configuration function. This is applicable for any IPsec tunnels such as IPv4 over IPv4, IPv4 over IPv6, IPv6 over IPv4 and IPv6 over IPv6.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 27, 2016.

#### Copyright and License Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

|                                                            |    |
|------------------------------------------------------------|----|
| 1. Introduction . . . . .                                  | 4  |
| 1.1. Terminology . . . . .                                 | 4  |
| 1.2. Conventions Used in This Document . . . . .           | 4  |
| 2. Motivation . . . . .                                    | 5  |
| 3. Procedure . . . . .                                     | 7  |
| 3.1. Sequence . . . . .                                    | 7  |
| 3.2. Extended format . . . . .                             | 8  |
| 3.2.1. Vendor ID . . . . .                                 | 8  |
| 3.2.2. MPSA_PUT . . . . .                                  | 8  |
| 3.3. Multi-point SA Management . . . . .                   | 14 |
| 3.3.1. Controller . . . . .                                | 14 |
| 3.3.2. CPE . . . . .                                       | 14 |
| 3.3.3. Rekeying . . . . .                                  | 15 |
| 3.4. Forwarding . . . . .                                  | 15 |
| 4. Peer discovery . . . . .                                | 16 |
| 4.1 example of MPSA with BGP for route based VPN . . . . . | 16 |
| 5. Security Considerations . . . . .                       | 16 |
| 5.1. Protected by MPSA . . . . .                           | 17 |
| 5.2 Security issues not to be solved by MPSA . . . . .     | 17 |
| 5.2.1 Attack from outside of the group . . . . .           | 17 |
| 5.2.2 Attack from inside of the group . . . . .            | 17 |
| 5.3 Forward secrecy and backward secrecy . . . . .         | 17 |
| 5. IANA Considerations . . . . .                           | 18 |
| 6. References . . . . .                                    | 18 |
| 6.1. Normative References . . . . .                        | 18 |
| 6.2. Informative References . . . . .                      | 18 |
| Authors' Addresses . . . . .                               | 18 |



## 1. Introduction

As described in the problem statement document [ad-vpn-problem], dynamic, secure and scalable system for establishing SAs is needed.

With multi-point SA, an endpoint automatically discovers other endpoint. In this draft, an endpoint means an inexpensive CPE, which can hardly establish large number of IPsec sessions simultaneously. The CPEs also share a multi-point SA within the same group, and there is no individual connection between them.

Scalability issue becomes serious in the service, such as triple play which requires large number of sessions at the same time. MPSA enables large scale simultaneous sessions even with inexpensive CPEs, and can avoid scalability issue.

The latency between CPEs can be minimized because of stateless shared multipoint SA, MPSA is suitable for video and voice services which is very sensitive to latency.

It can avoid the exhaustive configuration for CPEs and controllers. No reconfiguration is needed when a new CPE is added, removed, or changed. It can avoid high load on the controllers.

### 1.1. Terminology

Multi-point SA - This is similar to Dynamic Full Mesh topology described in [ad-vpn-problem]; direct connections exist in a hub and spoke manner, but only one SA for data transfer is shared with all CPEs.

### 1.2. Conventions 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 RFC 2119 [RFC2119].



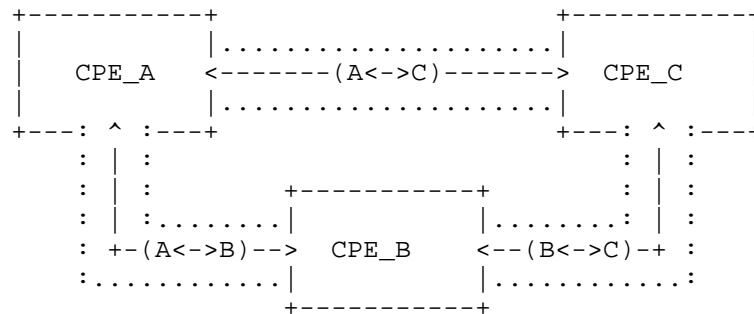


Figure 2

The solution in this document eliminates the problems listed above. Figure 3 shows topology of multi-point SA. Traffic between CPEs does not go through the controller, low latency, AAA of each CPE can be achieved, the number of IPsec connection is almost same as star topology, and no reconfiguration is needed for all the rest of CPEs even when a CPE is added, removed or changed. MPSA controller do not necessarily need to be router. It is possible to change MPSA controller for a software, because a communication load which spans IPsec Gateway by multi-point SA is not big.

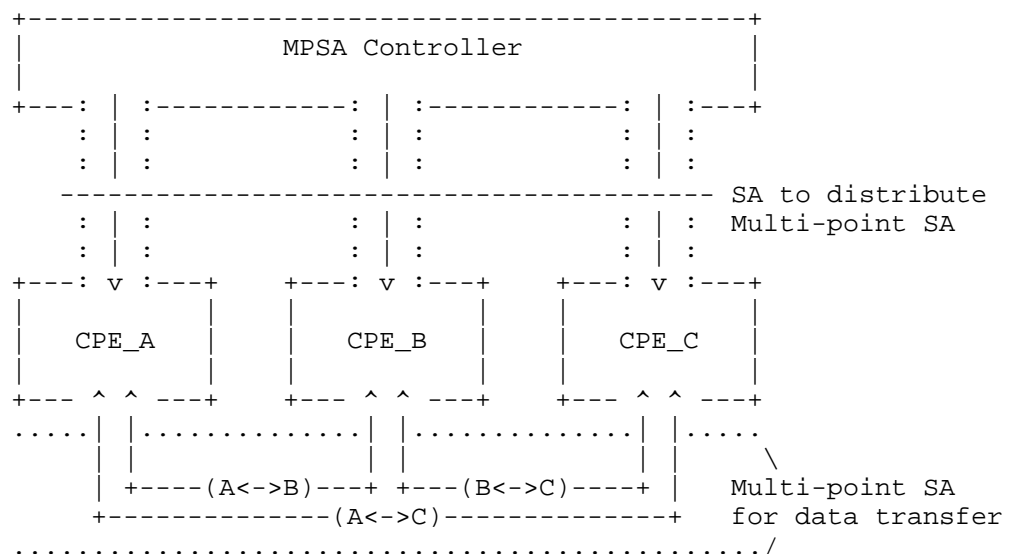


Figure 3

### 3. Procedure

#### 3.1. Sequence

The multi-point SA capability of the remote host is determined by an exchange of Vendor ID payloads. In the IKE\_SA\_INIT exchange, the Vendor ID payload for this specification is sent if the multi-point SA is used.

```

CPE Controller

HDR, SAi1, KEi,
 Ni, V(MPSA) -->
<-- HDR, SAR1, KEr,
 Nr, [CERTREQ,] V(MPSA)

```

MPSA: multi-point SA

The initial exchange (including IKE\_AUTH) is same as [IKEV2], other than Vendor ID payload included in IKE\_SA\_INIT.

After the initial exchange has finished successfully, a new INFORMATIONAL exchange is used to distribute multi-point SA to the CPE, with the Notify payload of MPSA\_PUT that includes cryptographic algorithm, nonce, keying material, SPI and so on. Keys for multi-point SA is generated according to the contents of the Notify payload by the CPE. The response of the Notify payload has empty Encrypted payload.

```

CPE Controller

HDR, SK {} -->
<-- HDR, SK {N(MPSA_PUT)}

```

### 3.2. Extended format

#### 3.2.1. Vendor ID

This document defines a new Vendor ID. The content of the payload is described below.

"multi-point SA"

#### 3.2.2. MPSA\_PUT

This document defines a new Notify Message Type MPSA\_PUT. The Notify Message Type of MPSA\_PUT is 40960. Notification Data of MPSA\_PUT has a Proposal-substructure-like format. It consists of Transform-substructure-like structures that have following data.

| Description                 | Trans. Type | Reference |
|-----------------------------|-------------|-----------|
| -----                       | -----       | -----     |
| Encryption Algorithm (ENCR) | 1           | RFC5996   |
| Pseudorandom Function (PRF) | 2           | RFC5996   |
| Integrity Algorithm (INTEG) | 3           | RFC5996   |
| Nonce (NONCE)               | 241         |           |
| SK_d (SKD)                  | 242         |           |
| Lifetime (LIFE)             | 243         |           |
| Rollover time 1 (ROLL1)     | 244         |           |
| Rollover time 2 (ROLL2)     | 245         |           |

- o Nonce - For Transform Type 241, the Transform ID is 1. The attribute contains actual nonce value with attribute type 16384. The size of the Nonce Data is between 16 and 256 octets.

| Name        | Number |
|-------------|--------|
| -----       | -----  |
| NONCE_NONCE | 1      |

| Attribute Type | Value | Attribute Format |
|----------------|-------|------------------|
| -----          | ----- | -----            |
| Nonce Value    | 16384 | TLV              |

- o SK\_d - For Transform Type 242, the Transform ID is 1. The attribute contains actual SK\_d value with attribute type 16385. The length of SK\_d Data is the preferred key length of the PRF.

| Name           | Number |                  |  |
|----------------|--------|------------------|--|
| -----          |        |                  |  |
| SKD_SK_D       | 1      |                  |  |
|                |        |                  |  |
| Attribute Type | Value  | Attribute Format |  |
| -----          |        |                  |  |
| SK_d Value     | 16385  | TLV              |  |

- o Lifetime - For For Transform Type 243, the Transform ID is 1. The attribute contains actual lifetime value with attribute type 16386. The length of Lifetime Value is 4 octets. Lifetime is stored in seconds as effective time of the multi-point SA.

| Name           | Number |                  |  |
|----------------|--------|------------------|--|
| -----          |        |                  |  |
| LIFE_LIFETIME  | 1      |                  |  |
|                |        |                  |  |
| Attribute Type | Value  | Attribute Format |  |
| -----          |        |                  |  |
| Lifetime Value | 16386  | TLV              |  |

- o Rollover time 1 - For Transform Type 244, the Transform ID is 1. The attribute contains actual rollover time 1 value with attribute type 16387. The length of Rollover time 1 Value is 4 octets. Rollover time 1 defines activation time delay for new outbound multi-point SA.

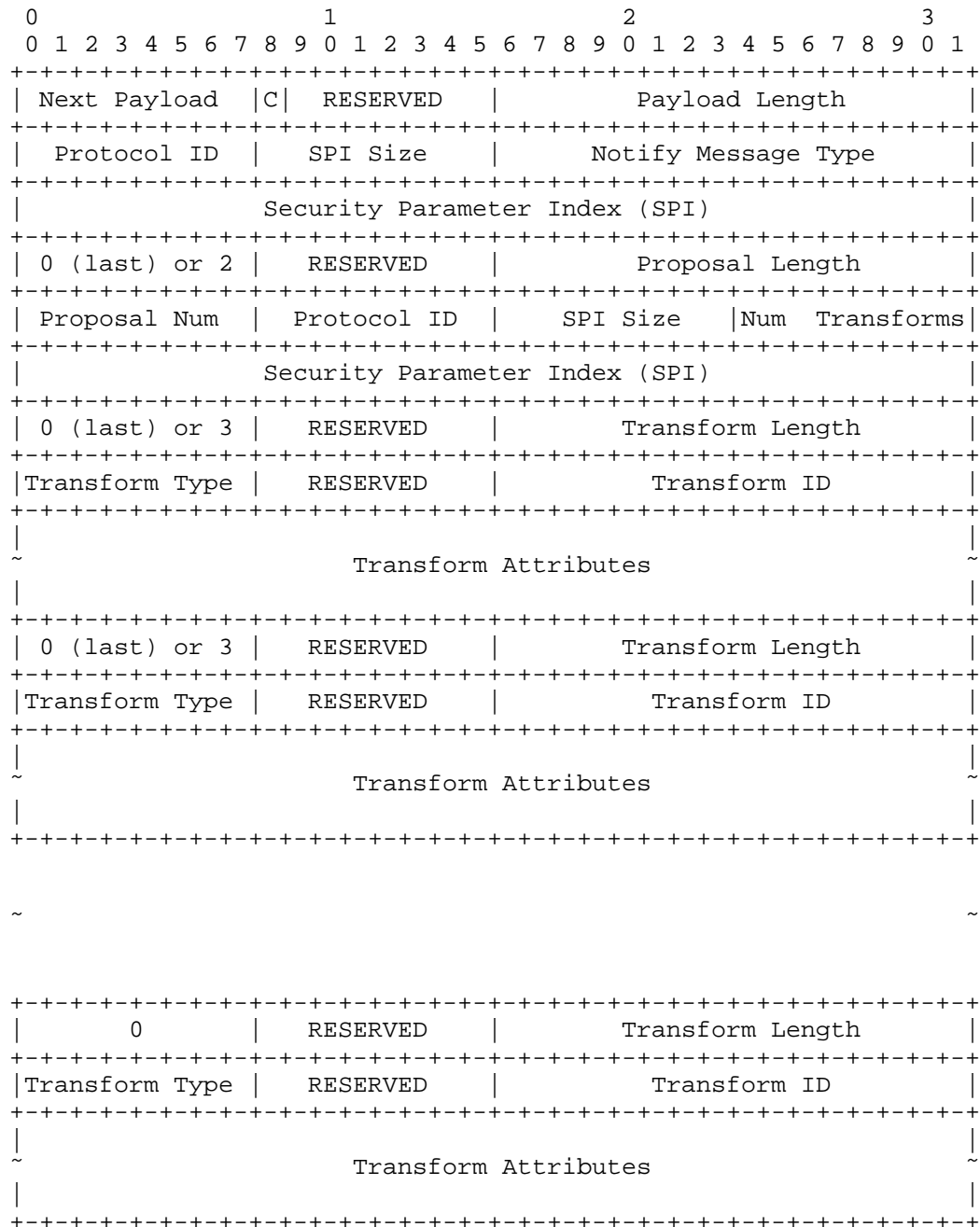
| Name            | Number |                  |  |
|-----------------|--------|------------------|--|
| -----           |        |                  |  |
| ROLL1_ROLLOVER1 | 1      |                  |  |
|                 |        |                  |  |
| Attribute Type  | Value  | Attribute Format |  |
| -----           |        |                  |  |
| Rollover1 Value | 16387  | TLV              |  |

- o Rollover time 2 - For Transform Type 245, the Transform ID is 1. The attribute contains actual rollover time 2 value with attribute type 16388. The length of Rollover time 2 Value is 4 octets. Rollover time 2 defines deactivation time delay for old inbound multi-point SA.

| Name            | Number |
|-----------------|--------|
| -----           |        |
| ROLL2_ROLLOVER2 | 1      |

| Attribute Type  | Value | Attribute Format |
|-----------------|-------|------------------|
| -----           |       |                  |
| Rollover2 Value | 16388 | TLV              |

Therefore, the format of the MPSA\_PUT of the Notify Message is described below.





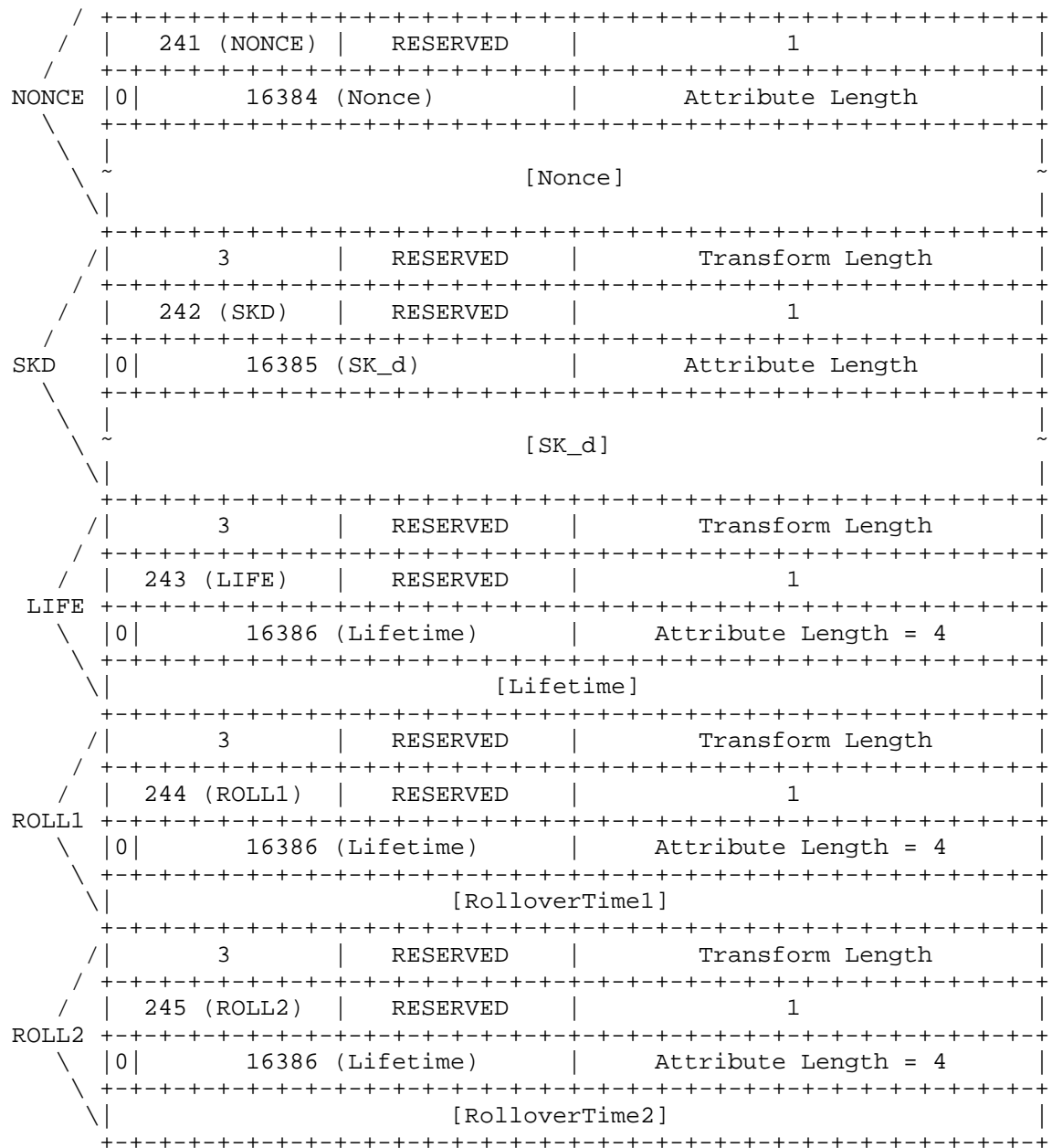
The following example shows a N(MPSA\_PUT) notification message. The SPIs in the Proposal-like and Tranform-like substructure are the same value. Following values are defined by the example.

```

Protocol: ESP
ENCR: AES-CBC (256bits)
PRF: SHA-1
INTEG: HMAC-SHA-1-96
NONCE: 241
SKD: 242
LIFE: 243
ROLL1: 244
ROLL2: 245

```

|        | 0                              | 1                   | 2                     | 3              |
|--------|--------------------------------|---------------------|-----------------------|----------------|
|        | 0 1 2 3 4 5 6 7 8 9            | 0 1 2 3 4 5 6 7 8 9 | 0 1 2 3 4 5 6 7 8 9   | 0 1            |
|        | +++++                          |                     |                       |                |
| /      | 0 (last)                       | C  RESERVED         | Payload Length        |                |
| /      | +++++                          |                     |                       |                |
| Notify | 3 (ESP)                        | SPI Size = 4        | MPSA_PUT              |                |
| \      | +++++                          |                     |                       |                |
| \      | Security Parameter Index (SPI) |                     |                       |                |
| /      | +++++                          |                     |                       |                |
| /      | 0 (last)                       | RESERVED            | Proposal Length       |                |
| Pro-   | +++++                          |                     |                       |                |
| pos-   | Prop Num = 1                   | 3 (ESP)             | SPI Size = 4          | Num Transforms |
| like   | +++++                          |                     |                       |                |
| \      | Security Parameter Index (SPI) |                     |                       |                |
| /      | +++++                          |                     |                       |                |
| /      | 3                              | RESERVED            | Transform Length      |                |
| /      | +++++                          |                     |                       |                |
| ENCR   | 1 (ENCR)                       | RESERVED            | 12 (ENCR_AES_CBC)     |                |
| \      | +++++                          |                     |                       |                |
| \      | 1                              | 14 (Key Length)     | 256                   |                |
| /      | +++++                          |                     |                       |                |
| /      | 3                              | RESERVED            | Transform Length      |                |
| PRF    | +++++                          |                     |                       |                |
| \      | 2 (PRF)                        | RESERVED            | 2 (PRF_HMAC_SHA1)     |                |
| /      | +++++                          |                     |                       |                |
| /      | 3                              | RESERVED            | Transform Length      |                |
| INTEG  | +++++                          |                     |                       |                |
| \      | 3 (INTEG)                      | RESERVED            | 2 (AUTH_HMAC_SHA1_96) |                |
| /      | +++++                          |                     |                       |                |
| /      | 3                              | RESERVED            | Transform Length      |                |



### 3.3. Multi-point SA Management

#### 3.3.1. Controller

Controller generates a multi-point SA for a group before connecting to any CPEs.

After the initial exchanges have finished, controller distributes the same multi-point SA information to CPEs within the group by sending N(MPSA\_PUT).

SPI and Nonce is generated similar way of [IKEv2]. SK\_d is generated from random numbers similar to Nonce.

The same SPI value is stored to Notify payload and Proposal-like substructure.

The multi-point SA will not be negotiated between controller and CPE, but will be notified from controller to CPE one way.

Controller initiates rekey before Lifetime expiration. As the Lifetime, controller notifies the effective time left of the multi-point SA.

#### 3.3.2. CPE

After the initial exchange has finished, CPE obtains multi-point SA information by receiving N(MPSA\_PUT) from controller. The keys for the multi-point SA are generated in the same procedure described in [IKEv2], except Ni | Nr is replaced by Nonce.

Therefore, KEYMAT is derived by PRF listed below.

$$\text{KEYMAT} = \text{prf}+(\text{SK\_d}, \text{Nonce})$$

The multi-point SA is protected in a cryptographic manner by ENCR and

INTEG which uses the generated keys.

The SPI value for the multi-point SA is the same of its in Notify message.

CPE uses the same multi-point SA as both inbound and outbound SAs.

CPE deletes both of inbound and outbound SA when Lifetime is expired.

Rollover time 1, 2 have no meaning when no old multi-point SA exists.

### 3.3.3. Rekeying

Rekeying should be finished before Lifetime expiration of current multi-point SA. Rekeying of multi-point SA will be performed as follows.

- Controller generates a new multi-point SA
- Controller distributes a new multi-point SA to all CPEs within the group
- CPE replaces the current multi-point SA to new one

CPE replaces multi-point SA using rollover method like [GDOI].

### 3.4. Forwarding

Each CPE sends and receives encapsulated packets using the multi-point SA.

The destination address of encapsulated packet will be determined with routing information, which can be achieved by static configuration or route exchange mechanism such as BGP on encapsulated environment described in [MESH].

It is applicable for any IPsec tunnels such as IPv4 over IPv4, IPv4

over IPv6, IPv6 over IPv4 and IPv6 over IPv6.

#### 4. Peer discovery

MPSA does not provide peer discovery function by itself. However, other mechanism, such as BGP, can be employed with MPSA for automatic peer discovery. One example is a use of BGP, described in [MESH], to learn peer information as next-hops.

##### 4.1 example of MPSA with BGP for route based VPN

Between controller and each peer, IKE\_SA and CHILD\_SA are established by IKEv2. On the IKE\_SA, an MPSA management message (MPSA\_PUT) is served from the controller to the peer.

On the CHILD\_SA, the controller and the peer establish a iBGP session to exchange route information (NLRIs). Controller can act as a BGP route reflector (RR), which can reflect NLRIs among all iBGP peers of the controller. In other words, the peer can learn all NLRIs advertised by all other peers.

According to [ENCAPS], each peer can advertise ESP peer address as well as conventional NLRIs, all of those can be reflected by RR on the controller.

At this point, each peer can have all other peer addresses as well as route information. The peer can decide a peer address by mean of recursive route lookup from the destination address of a packet to be forwarded. This decision can be made by the peer itself, without any additional communication with the controller.

Instead of [ENCAPS], each peer can also do it by [RNH]. Each peer learns all other peer addresses by BGP Remote-Next-Hop attributes and decides a peer address from a packet to be forwarded, as same as using [ENCAPS].

#### 5. Security Considerations

MPSA uses IKEv2 to protect MPSA management message, MPSA\_PUT. Thus, CPEs are authenticated by IKEv2. Using a shared SA for communication between CPEs, MPSA does not provide the following features.

- Data origin authentication
- Anti-replay protection

MPSA itself does not provide access control for user datagrams, but peer discovery may be able to provide access control as well as those

of route based VPN. For example, using BGP for peer discovery described in 4.1, access control could be provided by filtering exchanged routes at the controller. In this case, filtering by source address, protocol and ports can not be achieved. If you need it, you could do by other security policy rules as local setting at CPEs .

### 5.1. Protected by MPSA

- Authenticating CPEs and controller Authentication is provided by IKEv2 with pre-shared key or RSA signature. MPSA management messages are exchanged after IKEv2 negotiation.

- Confidentiality and integrity Packets are encapsulated by ESP, so that MPSA provides confidentiality and integrity against outside of the group, but does not them against members of the group

### 5.2 Security issues not to be solved by MPSA

#### 5.2.1 Attack from outside of the group

- Anti-replay protection

MPSA does not provide anti-replay protection, because sequence number synchronization between peers needs additional mechanism. Using a closed network as a transport might be effective to mitigate this kind of attacks.

- Leaking a IKE\_SA key

If an attacker could sniff packets on a IKE\_SA, and key of the SA were leaked, the attacker may get a key of MPSA by decoding a sniffed MPSA\_PUT message.

#### 5.2.2 Attack from inside of the group

If there is a malicious CPE or a CPE is hijacked by an attacker, MPSA can be attacked in the following way because MPSA, including cryptographic key, is shared by all CPEs.

- An attacker can impersonate another CPE. A closed network that prohibits source address spoofing could mitigate the impersonating.

- An attacker can decode packets between the other CPEs if the attacker could sniff packets.

### 5.3 Forward secrecy and backward secrecy

MPSA MAY be rekeyed when a CPE is removed from the group, for the removed CPE not to access the other CPEs communication after that, or when a CPE is added from the group, for it not to do before that. If not rekeyed, a removed/added CPE could access

## 5. IANA Considerations

This memo includes no request to IANA.

## 6. References

### 6.1. Normative References

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

### 6.2. Informative References

[GDOI] Weis, B., Rowles, S., and T. Hardjono, "The Group Domain of Interpretation", RFC 6407, October 2011.

[MESH] Wu, J., Cui, Y., Metz, C., and E. Rosen, "Softwire Mesh Framework", RFC 5565, June 2009.

[ad-vpn-problem] Manral, V. and S. Hanna, "Auto-Discovery VPN Problem Statement and Requirements", RFC 7018, September 2013.

[RNH] Van de Velde, G., Patel, K., Rao, D., Raszuk, R., and Bush, R., "BGP Remote-Next-Hop", draft-vandeveld-idr-remote-next-hop-07, June 2014

[ENCAPS] L. Berger, R. White and E. Rosen, "BGP IPsec Tunnel Encapsulation Attribute", RFC 5566, June 2009.

## Authors' Addresses

Arifumi Yamaya

Furukawa Network Solution Corp.  
5-1-9, Higashi-Yawata, Hiratsuka  
Kanagawa 254-0016, JAPAN  
Email: yamaya@fnsc.co.jp

Takafumi Ohya  
NTT Corporation  
Nishi-shinjuku, Shinjuku-ku,  
Tokyo 163-8019, JAPAN  
Email: takafumi.ooya@hco.ntt.co.jp

Tomohiro Yamagata  
KDDI Corporation  
Garden Air Tower  
Iidabashi, Chiyoda-ku,  
Tokyo 102-8460, JAPAN  
Email: to-yamagata@kddi.com

Satoru Matsushima  
Softbank Telecom Corp.  
1-9-1, Higashi-Shimbashi, Minato-Ku  
Tokyo 105-7322, JAPAN  
Email: satoru.matsushima@g.softbank.co.jp