

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: September 12, 2013

D. McGrew
Cisco Systems
W. Feghali
Intel Corp.
March 11, 2013

Cryptographic Algorithm Implementation Requirements and Usage Guidance
for Encapsulating Security Payload (ESP) and Authentication Header (AH)
draft-ietf-ipsecme-esp-ah-reqts-00

Abstract

This Internet Draft is standards track proposal to update to the Cryptographic Algorithm Implementation Requirements for ESP and AH; it also adds usage guidance to help in the selection of these algorithms.

The Encapsulating Security Payload (ESP) and Authentication Header (AH) protocols makes use of various cryptographic algorithms to provide confidentiality and/or data origin authentication to protected data communications in the IP Security (IPsec) architecture. To ensure interoperability between disparate implementations, the IPsec standard specifies a set of mandatory-to-implement algorithms. This document specifies the current set of mandatory-to-implement algorithms for ESP and AH, specifies algorithms that should be implemented because they may be promoted to mandatory at some future time, and also recommends against the implementation of some obsolete algorithms. Usage guidance is also provided to help the user of ESP and AH best achieve their security goals through appropriate choices of cryptographic 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 September 12, 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. Requirements Language | 3 |
| 1.2. Document History | 4 |
| 2. Implementation Requirements | 5 |
| 2.1. ESP Authenticated Encryption (Combined Mode Algorithms) | 5 |
| 2.2. ESP Encryption Algorithms | 5 |
| 2.3. ESP Authentication Algorithms | 5 |
| 2.4. AH Authentication Algorithms | 5 |
| 2.5. Summary of Changes | 5 |
| 3. Usage Guidance | 7 |
| 4. Rationale | 8 |
| 4.1. Authenticated Encryption | 8 |
| 4.2. Encryption Transforms | 8 |
| 4.3. Authentication Transforms | 9 |
| 5. Algorithm Diversity | 10 |
| 6. Acknowledgements | 11 |
| 7. IANA Considerations | 12 |
| 8. Security Considerations | 13 |
| 9. References | 14 |
| 9.1. Normative References | 14 |
| 9.2. Informative References | 14 |
| Authors' Addresses | 17 |

1. Introduction

The Encapsulating Security Payload (ESP) [RFC4303] and the Authentication Header (AH) [RFC4302] are the mechanisms for applying cryptographic protection to data being sent over an IPsec Security Association (SA) [RFC4301].

To ensure interoperability between disparate implementations, it is necessary to specify a set of mandatory-to-implement algorithms. This ensures that there is at least one algorithm that all implementations will have in common. This document specifies the current set of mandatory-to-implement algorithms for ESP and AH, specifies algorithms that should be implemented because they may be promoted to mandatory at some future time, and also recommends against the implementation of some obsolete algorithms. Usage guidance is also provided to help the user of ESP and AH best achieve their security goals through appropriate choices of mechanisms.

The nature of cryptography is that new algorithms surface continuously and existing algorithms are continuously attacked. An algorithm believed to be strong today may be demonstrated to be weak tomorrow. Given this, the choice of mandatory-to-implement algorithm should be conservative so as to minimize the likelihood of it being compromised quickly. Thought should also be given to performance considerations as many uses of IPsec will be in environments where performance is a concern.

The ESP and AH mandatory-to-implement algorithm(s) may need to change over time to adapt to new developments in cryptography. For this reason, the specification of the mandatory-to-implement algorithms is not included in the main IPsec, ESP, or AH specifications, but is instead placed in this document. Ideally, the mandatory-to-implement algorithm of tomorrow should already be available in most implementations of IPsec by the time it is made mandatory. To facilitate this, this document identifies such algorithms, as they are known today. There is no guarantee that the algorithms that we believe today may be mandatory in the future will in fact become so. All algorithms known today are subject to cryptographic attack and may be broken in the future.

1.1. Requirements Language

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].

Following [RFC4835], we define some additional key words:

MUST- This term means the same as MUST. However, we expect that at some point in the future this algorithm will no longer be a MUST.

SHOULD+ This term means the same as SHOULD. However, it is likely that an algorithm marked as SHOULD+ will be promoted at some future time to be a MUST.

SHOULD- This term means the same as SHOULD. However, it is likely that an algorithm marked as SHOULD- will be deprecated to a MAY or worse in a future version of this document.

SHOULD NOT+ This term means the same as SHOULD NOT. However, it is likely that an algorithm marked as SHOULD NOT+ will be deprecated to a MUST NOT in a future version of this document.

1.2. Document History

This is the initial version of this draft. It is based on an earlier individual submission [draft-mcgrew-ipsec-me-esp-ah-reqts], and incorporates feedback provided during the last IPsec ME meeting at IETF85. Triple-DES is now a MAY (instead of SHOULD NOT) and HMAC-MD5 is now ignored (instead of a SHOULD NOT), and "MAY" is no longer called out, except for algorithms that were previously listed as SHOULD, SHOULD+, or MUST.

This revision also adds a section discussing algorithm diversity, and references to new work on the selection of future cryptographic standards [draft-mcgrew-standby-cipher] and technical work showing the insecurity of 64-bit block ciphers (such as the Triple-DES algorithm) used to encrypt more than a gigabyte of data [M13].

2. Implementation Requirements

This section specifies the cryptographic algorithms that MUST be implemented, and provides guidance about ones that SHOULD or SHOULD NOT be implemented.

2.1. ESP Authenticated Encryption (Combined Mode Algorithms)

ESP combined mode algorithms provide both confidentiality and authentication services; in cryptographic terms, these are authenticated encryption algorithms [RFC5116]. Authenticated encryption transforms are listed in the ESP encryption transforms IANA registry.

| Requirement | Authenticated Encryption Algorithm |
|-------------|------------------------------------|
| ----- | ----- |
| SHOULD+ | AES-GCM [RFC4106] |
| MAY | AES-CCM [RFC4309] |

2.2. ESP Encryption Algorithms

| Requirement | Encryption Algorithm |
|-------------|-------------------------|
| ----- | ----- |
| MUST | NULL [RFC2410] |
| MUST | AES-128-CBC [RFC3602] |
| MAY | AES-CTR [RFC3686] |
| MAY | TripleDES-CBC [RFC2451] |
| SHOULD NOT+ | DES-CBC [RFC2405] |

2.3. ESP Authentication Algorithms

| Requirement | Authentication Algorithm (notes) |
|-------------|----------------------------------|
| ----- | ----- |
| MUST | HMAC-SHA1-96 [RFC2404] |
| SHOULD+ | AES-GMAC [RFC4543] |
| SHOULD | AES-XCBC-MAC-96 [RFC3566] |
| MAY | NULL [RFC4303] |

2.4. AH Authentication Algorithms

The requirements for AH are the same as for ESP Authentication Algorithms, except that NULL authentication is inapplicable.

2.5. Summary of Changes

| Old Requirement ----- | New Requirement ----- | Algorithm (notes) ----- |
|-----------------------------|-----------------------------|----------------------------|
| MAY | SHOULD+ | AES-GCM [RFC4106] |
| MAY | SHOULD+ | AES-GMAC [RFC4543] |
| MUST- | MAY | TripleDES-CBC [RFC2451] |
| SHOULD+ | SHOULD | AES-XCBC-MAC-96 [RFC3566] |
| SHOULD | MAY | AES-CTR [RFC3686] |

3. Usage Guidance

Since ESP and AH can be used in several different ways, this note provides guidance on the best way to utilize these mechanisms.

ESP can provide confidentiality, data origin authentication, or the combination of both of those security services. AH provides only data origin authentication. Background information on those security services is available [RFC4949]. In the following, we shorten 'data origin authentication' to 'authentication'.

Both confidentiality and authentication SHOULD be provided. If confidentiality is not needed, then authentication MAY be provided. Confidentiality without authentication is not effective [DP07] and SHOULD NOT be used. We describe each of these cases in more detail below.

To provide confidentiality and authentication, an authenticated encryption transform SHOULD be used in ESP, in conjunction with NULL authentication. Alternatively, an ESP encryption transform and ESP authentication transform MAY be used together (provided that neither transform is NULL). If authentication on the IP header is needed in conjunction with confidentiality of higher-layer data, then AH SHOULD be used in addition to the transforms recommended above. It is NOT RECOMMENDED to use ESP with NULL authentication in conjunction with AH; some configurations of this combination of services have been shown to be insecure [PD10].

To provide authentication without confidentiality, an authentication transform MUST be used in either ESP or AH. It is not possible to provide effective confidentiality without authentication, because the lack of authentication undermines the efficacy of encryption [B96][V02]. An encryption transform MUST NOT be used with a NULL authentication transform (unless the encryption transform is an authenticated encryption transform).

Triple-DES SHOULD NOT be used in any scenario in which multiple gigabytes of data will be encrypted with a single key. As a 64-bit block cipher, it leaks information about plaintexts above that "birthday bound" [M13]. Triple-DES CBC is listed as a MAY implement for the sake of backwards compatibility, but its use is discouraged.

4. Rationale

This section explains the principles behind the implementation requirements described above.

The algorithms listed as MAY-implement are not meant to be endorsed over other non-standard alternatives. All of the algorithms that appeared in [RFC4835] are included in this note, for the sake of continuity. In some cases, these algorithms have moved from being SHOULD-implement to MAY-implement algorithms.

4.1. Authenticated Encryption

This note encourages the use of authenticated encryption algorithms because they can provide significant efficiency and throughput advantages, and the tight binding between authentication and encryption can be a security advantage [RFC5116].

AES-GCM [RFC4106] brings significant performance benefits [KKGECD], has been incorporated into IPsec recommendations [RFC6379] and has emerged as the preferred authenticated encryption method in IPsec and other standards.

4.2. Encryption Transforms

Since ESP encryption is optional, support for the "NULL" algorithm is required to maintain consistency with the way services are negotiated. Note that while authentication and encryption can each be "NULL", they MUST NOT both be "NULL" [RFC4301] [H10].

AES Counter Mode (AES-CTR) is an efficient encryption method, but it provides no authentication capability. The AES-GCM authenticated encryption method has all of the advantages of AES-CTR, while also providing authentication. Thus this note moves AES-CTR from a SHOULD to a MAY.

The Triple Data Encryption Standard (TDES) is obsolete because of its small block size; as with all 64-bit block ciphers, it SHOULD NOT be used to encrypt more than one gigabyte of data with a single key [M13]. Its key size is smaller than that of the Advanced Encryption Standard (AES), while at the same time its performance and efficiency is worse. Thus, its use in new implementations is discouraged.

The Data Encryption Standard (DES) is obsolete because of its small key size and small block size. There have been publicly demonstrated and open-design special-purpose cracking hardware. Therefore, its use is discouraged.

4.3. Authentication Transforms

AES-GMAC provides good security along with performance advantages, even over HMAC-MD5. In addition, it uses the same internal components as AES-GCM and is easy to implement in a way that shares components with that authenticated encryption algorithm.

The MD5 hash function has been found to not meet its goal of collision resistance; it is so weak that its use in digital signatures is highly discouraged [RFC6151]. There have been theoretical results against HMAC-MD5, but that message authentication code does not seem to have a practical vulnerability. Thus, it may not be urgent to remove HMAC-MD5 from the existing protocols.

SHA-1 has been found to not meet its goal of collision resistance. However, HMAC-SHA-1 does not rely on this property, and HMAC-SHA-1 is believed to be secure.

The HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 are believed to provide a good security margin, and they perform adequately on many platforms. However, these algorithms are not recommended for implementation in this note, because HMAC-SHA-1 support is widespread and its security is good, AES-GMAC provides good security with better performance, and Authenticated Encryption algorithms do not need any authentication methods.

AES-XCBC has not seen widespread deployment, despite being previously being recommended as a SHOULD+ in RFC4305. Thus this draft lists it only as a SHOULD.

5. Algorithm Diversity

When the AES cipher was first adopted, it was decided to continue encouraging the implementation of Triple-DES, in order to provide algorithm diversity. But the passage of time has eroded the viability of Triple-DES as an alternative to AES. As it is a 64-bit block cipher, its security is inadequate at high data rates (see Section 4.2). Its performance in software and FPGAs is considerably worse than that of AES. Since it would not be possible to use Triple-DES as an alternative to AES in high data rate environments, or in environments where its performance could not keep up the requirements, the rationale of retaining Triple-DES to provide algorithm diversity is disappearing. (Of course, this does not change the rationale of retaining Triple-DES in IPsec implementations for backwards compability.)

It may be prudent to begin considering the selection of a different cipher that could provide algorithm diversity in IPsec and other IETF standards. There are many important criteria to consider, which are out of scope for this note. These issues have been taken up in recent work [draft-mcgrew-standby-cipher].

It is important to bear in mind that it is very highly unlikely that an exploitable flaw will be found in AES (e.g., a flaw that required less than a terabyte of known plaintext, when AES is used in a conventional mode of operation). The only reason that algorithm diversity deserves any consideration is because the problems that would be caused if such a flaw were found would be so large.

6. Acknowledgements

Much of the wording herein was adapted from [RFC4835], the parent document of this document. That RFC itself borrows from [RFC4305], which borrows in turn from [RFC4307]. RFC4835, RFC4305, and RFC4307 were authored by Vishwas Manral, Donald Eastlake, and Jeffrey Schiller respectively.

Thanks are due to Scott Fluhrer, Dan Harkins, Brian Weis, and Cheryl Madson for insightful feedback on this draft.

7. IANA Considerations

This memo includes no request to IANA.

8. Security Considerations

The security of a system that uses cryptography depends on both the strength of the cryptographic algorithms chosen and the strength of the keys used with those algorithms. The security also depends on the engineering and administration of the protocol used by the system to ensure that there are no non-cryptographic ways to bypass the security of the overall system.

This document concerns itself with the selection of cryptographic algorithms for the use of ESP and AH, specifically with the selection of mandatory-to-implement algorithms. The algorithms identified in this document as "MUST implement" or "SHOULD implement" are not known to be broken at the current time, and cryptographic research so far leads us to believe that they will likely remain secure into the foreseeable future. However, this is not necessarily forever. We would therefore expect that new revisions of this document will be issued from time to time that reflect the current best practice in this area.

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.
- [RFC2403] Madson, C. and R. Glenn, "The Use of HMAC-MD5-96 within ESP and AH", 1998.
- [RFC2404] Madson, C. and R. Glenn, "The Use of HMAC-SHA-1-96 within ESP and AH", 1998.
- [RFC2405] Madson, C. and N. Doraswamy, "The ESP DES-CBC Cipher Algorithm With Explicit IV", 1998.
- [RFC2410] Glenn, R. and S. Kent, "The NULL Encryption Algorithm and Its Use With IPsec", 1998.
- [RFC3566] Frankel, S. and H. Herbert, "The AES-XCBC-MAC-96 Algorithm and Its Use With IPsec", 2003.
- [RFC3602] Frankel, S., Glenn, R., and S. Kelly, "The AES-CBC Cipher Algorithm and Its Use with IPsec", 2003.
- [RFC3686] Housley, R., "Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)", 2004.
- [RFC4106] Viega, J. and D. McGrew, "The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)", 2005.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", 2005.
- [RFC4302] Kent, S., "IP Authentication Header", 2005.
- [RFC4303] Kent, S., "IP Encapsulating Security Payload", 2005.

9.2. Informative References

- [B96] Bellovin, S., "Problem areas for the IP security protocols (Proceedings of the Sixth Usenix Unix Security Symposium)", 1996.
- [DP07] Degabriele, J. and K. Paterson, "Attacking the IPsec Standards in Encryption-only Configurations (IEEE

Symposium on Privacy and Security)", 2007.

- [H10] Hoban, A., "Using Intel AES New Instructions and PCLMULQDQ to Significantly Improve IPsec Performance on Linux", 2010.
- [KKGECD] Kounavis, M., Kang, X., Grewal, K., Eszenyi, M., Gueron, S., and D. Durham, "Encrypting the Internet (SIGCOMM)", 2010.
- [M13] McGrew, D., "Impossible plaintext cryptanalysis and probable-plaintext collision attacks of 64-bit block cipher modes", 2012.
- [PD10] Paterson, K. and J. Degabriele, "On the (in)security of IPsec in MAC-then-encrypt configurations (ACM Conference on Computer and Communications Security, ACM CCS)", 2010.
- [RFC4305] Eastlake, D., "Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH)".
- [RFC4307] Schiller, J., "Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2)", 2005.
- [RFC4309] Housley, R., "Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)", 2005.
- [RFC4835] Manral, V., "Cryptographic Algorithm Implementation Requirements for Encapsulating Security Payload (ESP) and Authentication Header (AH)".
- [RFC4949] Shirley, R., "Internet Security Glossary, Version 2", 2007.
- [RFC5116] McGrew, D., "An Interface and Algorithms for Authenticated Encryption", 2008.
- [RFC6151] Turner, S. and L. Chen, "Updated Security Considerations for the MD5 Message-Digest and the HMAC-MD5 Algorithms", 2011.
- [RFC6379] Law, L. and J. Solinas, "Suite B Cryptographic Suites for IPsec", 2011.
- [V02] Vaudenay, S., "Security Flaws Induced by CBC Padding - Applications to SSL, IPSEC, WTLS ... (EUROCRYPT)", 2002.

[draft-mcgrew-ipsec-me-esp-ah-reqts]

McGrew, D., "Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)", 2012.

[draft-mcgrew-standby-cipher]

McGrew, D., Grieco, A., and Y. Sheffer, "Selection of Future Cryptographic Standards", 2013.

Authors' Addresses

David McGrew
Cisco Systems
13600 Dulles Technology Drive
Herndon, Virginia 20171
USA

Phone: 408 525 8651
Email: mcgrew@cisco.com

Wajdi Feghali
Intel Corp.
75 Reed Road
Hudson, Massachusetts
USA

Phone:
Email: wajdi.k.feghali@intel.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: January 3, 2014

V. Smyslov
ELVIS-PLUS
July 2, 2013

IKEv2 Fragmentation
draft-ietf-ipsecme-ikev2-fragmentation-00

Abstract

This document describes the way to avoid IP fragmentation of large IKEv2 messages. This allows IKEv2 messages to traverse network devices that don't allow IP fragments to pass through.

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 3, 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 | 3 |
| 1.1. Conventions Used in This Document | 3 |
| 2. Protocol details | 4 |
| 2.1. Overview | 4 |
| 2.2. Limitations | 4 |
| 2.3. Negotiation | 4 |
| 2.4. Using IKE Fragmentation | 5 |
| 2.5. Fragmenting Message | 6 |
| 2.5.1. Selecting Fragment Size | 7 |
| 2.5.2. Fragmenting Messages containing unencrypted Payloads | 8 |
| 2.6. Receiving IKE Fragment Message | 9 |
| 2.6.1. Changes in Replay Protection Logic | 10 |
| 3. Interaction with other IKE extensions | 11 |
| 4. Security Considerations | 12 |
| 5. IANA Considerations | 13 |
| 6. Acknowledgements | 14 |
| 7. References | 15 |
| 7.1. Normative References | 15 |
| 7.2. Informative References | 15 |
| Author's Address | 16 |

1. Introduction

The Internet Key Exchange Protocol version 2 (IKEv2), specified in [RFC5996], uses UDP as a transport for its messages. When IKE message size exceeds path MTU, it gets fragmented by IP level. The problem is that some network devices, specifically some NAT boxes, don't allow IP fragments to pass through. This apparently blocks IKE communication and, therefore, prevents peers from establishing IPsec SA.

The solution to the problem described in this document is to perform fragmentation of large messages by IKE itself, replacing them by series of smaller messages. In this case the resulting IP Datagrams will be small enough so that no fragmentation on IP level will take place.

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. Protocol details

2.1. Overview

The idea of the protocol is to split large IKE message into the set of smaller ones, calling Fragment Messages. On the receiving side Fragment Messages are collected and merged together to get original message. In general this approach increases receiver's vulnerability to Denial of Service attack. To reduce this vulnerability Fragment Messages are individually encrypted and authenticated. This implies that message cannot be fragmented until shared secret is calculated.

2.2. Limitations

In general, original message can be fragmented if and only if it contains Encrypted Payload. It means that messages in IKE_SA_INIT Exchange cannot be fragmented. In most cases this is not a problem, since IKE_SA_INIT messages are usually small enough to avoid IP fragmentation. But in some cases (advertising a badly structured long list of algorithms, using large MODP Groups, etc.) those messages may become fairly large and get fragmented by IP level. In these cases the described solution won't help.

Another limitation is that the minimal size of IP Datagram bearing IKE Fragment Message is about 100 bytes depending on the algorithms employed. According to [RFC0791] the minimum IP Datagram size that is guaranteed not to be further fragmented is 68 bytes. So, even the smallest IKE Fragment Messages could be fragmented by IP level in some circumstances. But such extremely small PMTU sizes are very rare in real life.

2.3. Negotiation

Initiator MAY indicate its support for IKE Fragmentation and willingness to use it by including Notification Payload of type IKE_FRAGMENTATION_SUPPORTED in IKE_SA_INIT request message. If Responder also supports this extension and is willing to use it, it includes this notification in response message.

| Initiator | Responder |
|---|--|
| ----- | ----- |
| HDR, SAi1, KEi, Ni, [N(IKE_FRAGMENTATION_SUPPORTED)] | --> |
| | <-- HDR, SAR1, KEr, Nr, [CERTREQ], [N(IKE_FRAGMENTATION_SUPPORTED)] |

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(=0)| SPI Size (=0) |      Notify Message Type      |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

- o Protocol ID (1 octet) MUST be 0.
- o SPI Size (1 octet) MUST be 0, meaning no SPI is present.
- o Notify Message Type (2 octets) - MUST be xxxxxx, the value assigned for IKE_FRAGMENTATION_SUPPORTED by IANA.

This Notification contains no data.

2.4. Using IKE Fragmentation

After IKE Fragmentation is negotiated, it is up to Initiator of each Exchange, whether to use it or not. In most cases IKE Fragmentation will be used in IKE_AUTH Exchange, especially if certificates are employed. Initiator may first try to send unfragmented message and resend it fragmented only if it didn't receive response after several retransmissions, or it may always send messages fragmented (but see Section 3), or it may fragment only large messages and messages causing large responses.

In general the following guidelines are applicable:

- o Initiator MAY fragment outgoing message if it suspects that either request or response message may be fragmented by IP level.
- o Initiator SHOULD fragment outgoing message if it suspects that either request or response message may be fragmented by IP level and IKE Fragmentation was already used in one of previous Exchanges in the context of the current IKE SA.
- o Initiator SHOULD NOT fragment outgoing message if both request and response messages of the Exchange are small enough not to cause fragmentation on IP level (for example, there is no point in fragmenting Liveness Check messages).

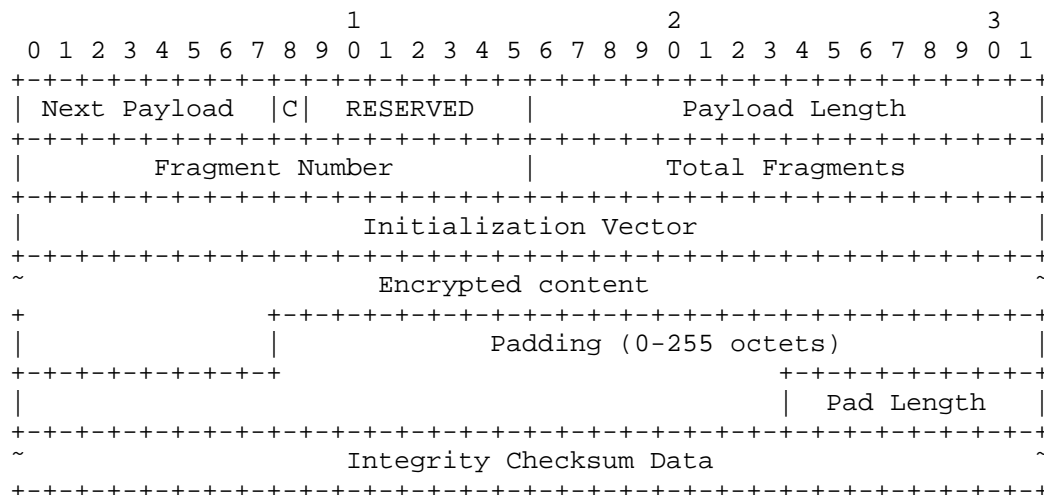
Responder MUST send response message in the same form (fragmented or not) as corresponded request message. If it received unfragmented request message, responded with unfragmented response message and then received fragmented retransmission of the same request, it MUST resend its response back to Initiator fragmented.

2.5. Fragmenting Message

Message to be fragmented MUST contain Encrypted Payload. For the purpose of IKE Fragment Messages construction original (unencrypted) content of Encrypted Payload is broken down into parts. Its content is treated as a binary blob and is broken down regardless of inner Payloads boundaries. Each of resulting parts is treated as a content for Encrypted Fragment Payload.

The Encrypted Fragment Payload, denoted SKF{...}, contains other payloads in encrypted form. The Encrypted Fragment Payload, as well as Encrypted Payload from [RFC5996], if present in a message, MUST be the last payload in the message.

The payload type for an Encrypted Fragment payload is XXX (TBA by IANA).



Encrypted Fragment Payload

- o Next Payload (1 octet) - in the very first fragment MUST be set to Payload Type of the first inner Payload (as in Encrypted Payload). In the rest fragments MUST be set to zero.
- o Fragment Number (2 octets) - current fragment number starting from 1. This field MUST be less than or equal to the next field, Total Fragments.
- o Total Fragments (2 octets) - number of fragments original message was divided into. This field MUST NOT be zero.

Other fields are identical to those specified in Section 3.14 of [RFC5996].

When prepending IKE Header, Length field MUST be adjusted to reflect the length of constructed message and Next Payload field MUST reflect payload type of the first Payload in the constructed message (that in most cases will be Encrypted Fragment Payload). All newly constructed messages MUST retain the same Message ID as original message. After prepending IKE Header and possibly any of Payloads that precedes Encrypted Payload in original message (see Section 2.5.2), the resulting messages are sent to the peer.

Below is an example of fragmenting some message.

HDR(MID=n), SK(NextPld=PLD1) {PLD1 ... PLDN}

Original Message

HDR(MID=n), SKF(NextPld=PLD1, Frag#=1, TotalFrag=m) {...},
HDR(MID=n), SKF(NextPld=0, Frag#=2, TotalFrag=m) {...},
...
HDR(MID=n), SKF(NextPld=0, Frag#=m, TotalFrag=m) {...}

IKE Fragment Messages

2.5.1. Selecting Fragment Size

When breaking content of Encrypted Payload down into parts sender SHOULD chose size of those parts so, that resulting IP Datagram size not exceed some fragmentation threshold - be small enough to avoid IP fragmentation.

If sender has some knowledge about PMTU size it MAY use it. If sender is a Responder in the Exchange and it has received fragmented request, it MAY use maximum size of received IKE Fragment Message IP Datagrams as threshold when constructing fragmented response.

Otherwise for messages to be sent over IPv6 it is RECOMMENDED to use value 1280 bytes as a maximum IP Datagram size ([RFC2460]). For messages to be sent over IPv4 it is RECOMMENDED to use value 576 bytes as a maximum IP Datagram size.

For IPv4 Encrypted Payload content size is less than IP Datagram size by the sum of the following values:

- o IPv4 header size (typically 20 bytes, up to 60 if IP options are present)

- o UDP header size (8 bytes)
- o non-ESP marker size (4 bytes if present)
- o IKE Header size (28 bytes)
- o Encrypted Payload header size (4 bytes)
- o IV size (varying)
- o padding and its size (at least 1 byte)
- o ICV size (varying)

The sum may be estimated as 61..105 bytes + IV + ICV + padding. For IPv6 this estimation is difficult as there may be varying IPv6 Extension headers included.

According to [RFC0791] the minimum IPv4 datagram size that is guaranteed not to be further fragmented is 68 bytes, but it is generally impossible to use such small value for solution, described in this document. Using 576 bytes is a compromise - the value is large enough for the presented solution and small enough to avoid IP fragmentation in most situations. Several other UDP-based protocol assume the value 576 bytes as a safe low limit for IP datagrams size (Syslog, DNS, etc.). Sender MAY use other values if they are appropriate.

Initiator MAY try to discover path MTU by using several values of fragmentation threshold, provided that it starts with larger values and fragments message again with next smaller value if it doesn't receive response in a reasonable time after several retransmissions. In this case using next smaller value MUST result in increasing Total Fragments field.

2.5.2. Fragmenting Messages containing unencrypted Payloads

Currently no one of IKEv2 Exchanges defines messages, containing both unencrypted payloads and payloads, protected by Encrypted Payload. But IKEv2 doesn't forbid such messages. If some future IKEv2 extension defines such a message and it needs to be fragmented, all unprotected payloads MUST be in the first fragment, along with Encrypted Fragment Payload, which MUST be present in any IKE Fragment Message.

Below is an example of fragmenting message, containing both encrypted and unencrypted Payloads.

HDR(MID=n), PLD0, SK(NextPld=PLD1) {PLD1 ... PLDN}

Original Message

HDR(MID=n), PLD0, SKF(NextPld=PLD1, Frag#=1, TotalFrag=m) {...},
HDR(MID=n), SKF(NextPld=0, Frag#=2, TotalFrag=m) {...},
...
HDR(MID=n), SKF(NextPld=0, Frag#=m, TotalFrag=m) {...}

IKE Fragment Messages

Note, that the size of each IP Datagram bearing IKE Fragment Messages SHOULD NOT exceed fragmentation threshold, including the very first, which contains unprotected Payloads. This will reduce the size of Encrypted Fragment Payload content in the first IKE Fragment Message to accommodate unprotected Payloads. In extreme cases Encrypted Fragment Payload will contain no data, but it is still MUST be present in the message, because only its presence allows receiver to distinguish IKE Fragment Message from regular IKE message.

2.6. Receiving IKE Fragment Message

Receiver identifies IKE Fragment Message by the presence of Encrypted Fragment Payload in it. Note, that it is possible for this payload to be not the first (and the only) payload in the message (see Section 2.5.2). But for all currently defined IKEv2 exchanges this payload will be the first and the only payload in the message.

Upon receiving IKE Fragment Message the following actions are performed:

- o Check message validity - in particular, check whether values of Fragment Number and Total Fragments in Encrypted Fragment Payload are valid. If not - message MUST be silently discarded.
- o Check, that this IKE Fragment Message is new for the receiver and not a replay. If IKE Fragment message with the same Message ID, same Fragment Number and same Total Fragments fields was already received and successfully processed, this message is considered a replay and MUST be discarded.
- o Verify IKE Fragment Message authenticity by checking ICV in Encrypted Fragment Payload. If ICV check fails message MUST be silently discarded.
- o If reassembling isn't finished yet and Total Fragments field in received IKE Fragment Message is greater than this field in

previously received fragments, receiver MUST discard all received fragments and start reassembling over with just received IKE Fragment Message.

- o Store message in the list waiting for the rest of fragments to arrive.

When all IKE Fragment Messages (as indicated in the Total Fragments field) are received, content of their Encrypted Fragment Payloads is decrypted and merged together to form content of original Encrypted Payload, and, therefore, along with IKE Header, original message. Then it is processed as if it was received, verified and decrypted as as regular unfragmented message.

2.6.1. Changes in Replay Protection Logic

According to [RFC5996] IKEv2 MUST reject message with the same Message ID as it has seen before (taking into consideration Response bit). This logic has already been updated by [RFC6311], which deliberately allows any number of messages with zero Message ID. This document also updates this logic: if message contains Encrypted Fragment Payload, the values of Fragment Number and Total Fragments fields from this payload MUST be used along with Message ID to detect retransmissions and replays.

If Responder receives IKE Fragment Message after it received, successfully verified and processed regular message with the same Message ID, it means that response message didn't reach Initiator and it activated IKE Fragmentation. If Fragment Number in Encrypted Fragment Payload in this message is equal to 1, Responder MUST fragment its response and retransmit it back to Initiator in fragmented form.

If Responder receives a replay IKE Fragment Message for already reassembled, verified and processed fragmented message, it MUST retransmit response back to Initiator, but only if Fragment Number field in Encrypted Fragment Payload is equal to 1 and MUST silently discard received message otherwise.

3. Interaction with other IKE extensions

IKE Fragmentation is compatible with most of defined IKE extensions, like IKE Session Resumption [RFC5723], Quick Crash Detection Method [RFC6290] and so on. It neither affect their operation, nor is affected by them. It is believed that IKE Fragmentation will also be compatible with most future IKE extensions, if they follow general principles of formatting, sending and receiving IKE messages, described in [RFC5996].

The notable exception that requires a special care is [RFC6311] - Protocol Support for High Availability of IKEv2. As it deliberately allows any number of synchronization Exchanges to have the same Message ID - zero, standard replay detection logic, based on checking Message ID is not applicable for such messages, and receiver has to check message content to detect replays. When implementing IKE Fragmentation along with [RFC6311], IKE Message ID Synchronization messages MUST NOT be sent fragmented to simplify receiver's task of detecting replays. Fortunately, these messages are small and there is no point in fragmenting them anyway.

4. Security Considerations

Most of the security considerations for IKE Fragmentation are the same as those for base IKEv2 protocol described in [RFC5996]. This extension introduces Encrypted Fragment Payload to protect content of IKE Message Fragment. This allows receiver to individually check authenticity of fragments, thus protecting itself from Denial of Service attack.

5. IANA Considerations

This document defines new Payload in the "IKEv2 Payload Types" registry:

| | | |
|-------|----------------------------|-----|
| <TBA> | Encrypted Fragment Payload | SKF |
|-------|----------------------------|-----|

This document also defines new Notify Message Types in the "Notify Messages Types - Status Types" registry:

| | |
|-------|-----------------------------|
| <TBA> | IKE_FRAGMENTATION_SUPPORTED |
|-------|-----------------------------|

6. Acknowledgements

We would like to thank Tero Kivinen, Yoav Nir, Paul Wouters for their review comments.

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.
- [RFC6311] Singh, R., Kalyani, G., Nir, Y., Sheffer, Y., and D. Zhang, "Protocol Support for High Availability of IKEv2/IPsec", RFC 6311, July 2011.

7.2. Informative References

- [RFC0791] Postel, J., "Internet Protocol", STD 5, RFC 791, September 1981.
- [RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, December 1998.
- [RFC5723] Sheffer, Y. and H. Tschofenig, "Internet Key Exchange Protocol Version 2 (IKEv2) Session Resumption", RFC 5723, January 2010.
- [RFC6290] Nir, Y., Wierbowski, D., Detienne, F., and P. Sethi, "A Quick Crash Detection Method for the Internet Key Exchange Protocol (IKE)", RFC 6290, June 2011.

Author's Address

Valery Smyslov
ELVIS-PLUS
PO Box 81
Moscow (Zelenograd) 124460
RU

Phone: +7 495 276 0211
Email: svan@elvis.ru

IPsecME Working Group
INTERNET-DRAFT

Intended Status: Proposed Standard
Expires: February 19, 2014

Y. Mao
Z. Wang
Hangzhou H3C Tech. Co., Ltd.
V. Manral
HP
August 18, 2013

Auto Discovery VPN Protocol
draft-mao-ipsecme-ad-vpn-protocol-02

Abstract

This document describes the Auto Discovery VPN (ADVPN) protocol, the use case and problem statement for which is described in [ADVPN_Problem]. The ADVPN protocol is used for enabling a large of number of entities to communicate directly among the peers, with minimal configuration and operator intervention. The solution uses IPsec[RFC4301] to protect communication between the peers.

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), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/lid-abstracts.html>

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>

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 | 3 |
| 1.1. ADVPN Protocol Overview | 3 |
| 1.2. Terminology | 5 |
| 1.3. Conventions Used in This Document | 5 |
| 2. Design Overview | 5 |
| 3. How ADVPN Works | 7 |
| 4. ADVPN protocol | 8 |
| 4.1. Client Information Registration | 8 |
| 4.2. Client Information Resolution | 9 |
| 4.3. Private Network Information Management | 10 |
| 4.4. Shortcut Decision | 11 |
| 4.5. Redirect protocol | 11 |
| 4.6. Keepalive protocol | 12 |
| 4.7. Session Protocol | 12 |
| 5. ADVPN Message Formats | 13 |
| 5.1. ADVPN Fixed Header | 13 |
| 5.2. Mandatory Part | 15 |
| 5.2.1. Client Identification Header | 15 |
| 5.2.2. Session Header | 16 |
| 5.3. Payload Part | 17 |
| 5.3.1. Client Information Payload | 17 |
| 5.3.2. Destination Client Payload | 19 |
| 5.3.3. Network Information Payload | 20 |
| 5.3.4. Traffic Flow Payload | 21 |
| 5.3.5. Keepalive Parameter Payload | 23 |
| 5.3.6. Redirect Payload | 23 |
| 6. Implementation Status | 25 |
| 7. Security Considerations | 26 |
| 8. IANA Considerations | 26 |
| 9. Acknowledgments | 26 |
| 10. References | 26 |
| 10.1. Normative References | 26 |
| 10.2. Informative References | 27 |
| Appendix A. Comparison Against ADVPN Requirements | 27 |
| Authors' Addresses | 29 |

1. Introduction

The large scale deployment of IPsec[RFC4301] leads to lots of difficulties, such as configuration for each tunnel, adding or removing IPsec peer, etc. all need a lot of configuration/reconfiguration. Therefore, a protocol to establish IPsec tunnel dynamically without having the large overhead of configuration is needed. Auto Discovery VPN Problem Statement and Requirement [ADVPN_Problem] defines all requirements for the large scale IPsec deployment problem. This document defines the Auto Discovery VPN (ADVPN) protocol to satisfy these requirements.

1.1. ADVPN Protocol Overview

The overall ADVPN solution has control plane and data plane elements. The ADVPN protocol operates in the control plane and uses the standard IPsec [RFC4301] for the data plane. The IPsec data plane establishes and protects all the traffic in the ADVPN network.

The ADVPN protocol is a client and server protocol. The ADVPN Client(ADC) registers its information to the ADVPN Server(ADS). When the ADC wants to establish an IPsec tunnel with another ADC, the ADC requests and queries another ADC's information on the ADS. The ADVPN Client(ADC) is the forwarding device in the data plane. ADC implements IPsec to protect its own traffic or the traffic flowing through ADC.

The ADVPN Server(ADS) is ADVPN information controller, which is responsible for collection, maintenance and distribution of ADVPN Client(ADC) information and the control policy. The client information is registered by the ADVPN client. The client information includes private IP address, public IP address and private network information etc. The control policy is used to decide the topology should be star topology or full mesh topology, and the control policy is pushed to hub ADC from ADS.

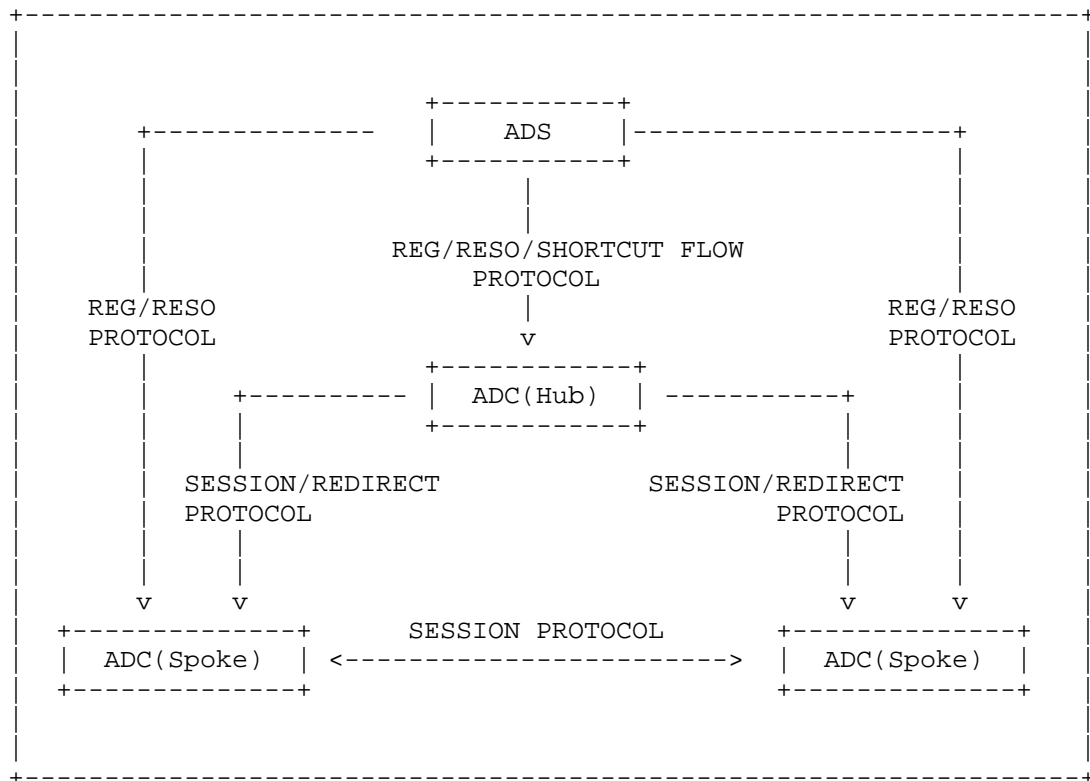


Figure 1. ADVPN Protocol Model

In this diagram, ADVPN protocol is implemented between ADC and ADS. All the ADCs register information on the ADS. When the ADC tries to build a direct IPsec tunnel with another ADC, it sends the Resolution message to ADS to query the information. In addition, to allow the shortcut path establishment between the ADCs, the ADS sends the Shortcut Traffic Flow message to the hub ADCs. Which ADC is hub ADC SHOULD be specified in the ADS by administrator.

ADVPN protocol can also be implemented between ADC and ADC. The ADC sends the Session message to another ADC to transfer ADC information; otherwise the other ADC has to query the ADS if there is a reverse traffic, which may not be practical in some cases. The hub ADC sends Redirect message to spoke ADC to trigger the spoke ADC send Resolution message to ADS.

With ADVPN protocol, the IPsec gateways and endpoints can obtain the IPsec peer's remote address from ADS. Therefore, establishing IPsec tunnel between them can scale well.

1.2. Terminology

Most terminology has been specified in the [ADVPN_Problem]. However, there are also some additional terminology in this document needs to be clarified.

ADVPN Server - This is an entity as ADVPN network controller. It maintains ADVPN client information database. It also can decide whether the spoke can directly communicate with the other spoke.

ADVPN Client - This is an entity as ADVPN peer. It registers its information to ADVPN server.

Hub ADC - This ADC is hub in the forwarding path.

Spoke ADC - This ADC is spoke in the forwarding path.

Private IP Address - Each ADVPN client has a logical private IP address. This address is static and as identity of ADVPN client. Through the private IP address, the Public IP address is discovered.

Public IP Address - This is IPsec peer address. This address can be dynamic and attached with Private IP Address. There is private IP address to public IP address mapping.

Private Network Information - This information includes the network behind the spoke and the next hop which is private IP address.

Shortcut Path - This is direct connectivity between spoke and spoke.

ADVPN Network - This is network composed of ADVPN peers

Source ADVPN Peer - This ADVPN peer is the initiator to establish IPsec tunnel.

Destination ADVPN Peer - This ADVPN peer is the responder to establish IPsec tunnel.

1.3. 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].

2. Design Overview

In the ADVPN network, there are thousands of ADVPN peers. It is difficult to pre-configure the SPD and PAD for each ADVPN peer

manually. In the use cases in the [ADVPN_Problem], the endpoints and gateways can use a dynamic IP address, it is impossible to specify the IP address towards this ADVPN peer in the remote ADVPN peer. Therefore, the design goal of ADVPN protocol described in this document is each ADVPN peer only configures its own SPD and PAD. If the ADVPN peer tries to setup an IPsec tunnel with another ADVPN peer, it queries the ADS and obtains the client information of another peer.

To discover the remote ADVPN peer, a private IP address is used as the search key. The search key is private IP address. The private IP address is logical Virtual Private Network (VPN) IP address. Although IPsec peer address is dynamic, the private IP address is static. The private IP address is also next hop towards the network behind the ADVPN peer. When the traffic flows through the source ADVPN peer, routing table is looked up, the next hop is the private IP address of destination ADVPN peer. The source ADVPN peer looks up session table to find the public IP address of destination ADVPN peer. Session table is the ADC information cache in the forwarding path, and it has the private IP address to public IP address mapping.

The private IP address with the private network information SHOULD be transferred via routing protocol (e.g. OSPF, BGP or RIP) or another means in the ADVPN network. A routing protocol can run over the IPsec tunnel between the ADCs and ADS. The routing protocol helps distribute routing information to all ADC's towards about the destination network behind the ADCs.

After a spoke ADC finishes the registration on ADS, this ADC also obtains the information of hub ADC from ADS. An IPsec tunnel is then established automatically between the spoke ADC and hub ADC. The routing protocol messages are protected and transferred in this IPsec tunnel and exchanged between the spoke ADC and hub ADC.

In the full mesh ADVPN network, there are tens of thousands of ADVPN peers. The spokes cannot be populated with a full routing table due to constraints on the device capabilities; therefore the network is left as a hub-and-spoke network initially. After routing information exchanges between the hub and spokes, routing table in the source ADVPN peer has the private IP address of hub ADVPN peer as the next hop to networks behind the destination ADVPN peer. Therefore, the traffic from source ADVPN peer to destination ADVPN peer is forwarded to hub first.

In order to have direct connectivity between the ADVPN peers, the ADVPN peer queries the direct route information on ADS. The spoke ADC SHOULD register its private IP address and network behind the ADC to the ADS. The private IP address is the next hop to network behind the

ADC. When the traffic is transferred through hub, the hub sends a redirect message to source ADVPN peer. When the source ADVPN peer receiving Redirect message, it send a Resolution message to ADS to query the direct route information and ADC information for the destination ADVPN peer. Receiving the resolution reply message, the source ADVPN peer adds a direct route in the route table and setup a direct IPsec tunnel with the destination ADVPN peer.

The hub decides whether the spoke can be allowed to have a direct communication with other spoke. The decision depends on the control policy pushed by the ADS after the hub ADC finishes registration in the ADS. The control policy is flow information. If the traffic matches the flow information, the hub sends a redirect message to source ADVPN peer to find a shortcut path to destination ADVPN peer.

3. How ADVPN Works

In the ADVPN solution, ADVPN protocol uses the IPsec protocol [RFC4301] data plane, as well as routing protocols to fulfill the ADVPN function. This section describes the process to establish a shortcut spoke-to-spoke IPsec tunnel in a mesh topology.

1. When the ADC device comes up, it registers its information to ADS. The information includes the private IP address, the public IP address and the network behind the ADC.

2. The ADS sends the registration reply message to the ADC. The spoke ADC obtains the information of hub ADC. The ADC creates the hub session in the session table. After that, the spoke ADC establishes an IPsec tunnel with hub ADC.

3. The ADS sends Shortcut Flow message to hub ADC in order to determine whether sending redirect message or not.

4. The spoke ADC send a Session Setup message to hub ADC protected by IPsec tunnel, the hub ADC has the spoke ADC's information.

5. All the route protocol packets run over IPsec tunnel between the spoke ADC and hub ADC. The route protocol packet is copied and sent to hub ADC. The ADVPN network has spoke-hub topology.

6. When the traffic towards destination ADVPN peer arrives in the source ADVPN peer device, from the routing table, the next hop is the private IP address of hub ADVPN peer. Match the private IP address in the session table to obtain the public IP address of hub ADVPN peer. By the public IP address, the spoke-to-hub IPsec SA is chosen to encapsulate the traffic.

7. The traffic arrives in the hub, after processing IPsec packet, the hub looks up routing table to determine if incoming and outgoing interface is in the same ADVPN network. If it is, this traffic is transferred through hub towards the destination spoke and there is shortcut path between them. If the traffic matches the Shortcut Flow table, the hub send a redirect message to source ADVPN peer.

8. The source ADVPN peer receives the redirect message, it sends Resolution Request message with destination IP address to ADS. ADS looks in the ADC information database to find out the next hop to the destination IP address and related network information. The ADS sends a Resolution Response message to the source ADVPN peer.

9. The source ADVPN peer receives the Resolution Response message. Firstly, a route towards destination network is added into the route table. Secondly, the source ADVPN peer establishes an IPsec tunnel with the destination ADVPN peer. Lastly, the source ADVPN peer send a session message to destination ADVPN peer. The destination ADVPN peer can add the reverse route in its route table and the ADC information in session table.

4. ADVPN protocol

ADVPN protocol listens and sends on UDP port 2013(pending assignment by IANA). Since the UDP is a datagram(unreliable) protocol, all messages in ADVPN exist in pairs: a request and a response.

In the following descriptions, the payloads contained in the message are indicated by names as listed below.

| Notation | Description |
|----------|------------------------------|
| ----- | |
| HDR | ADVPN header |
| CIDHdr | Client Identification Header |
| SessHdr | Session Header |
| CI | Client Information payload |
| DC | Destination Client Payload |
| NI | Network Information Payload |
| TF | Traffic Flow Payload |
| KP | Keepalive Parameter Payload |
| Red | Redirect Payload |

The details of the contents of each header and payload are described in section 5. Payloads that may optionally appear will be shown in brackets, such as [CI]; this indicates that a Client Information payload can optionally be included.

4.1. Client Information Registration

The ADC sends a Registration Request message to ADS to register its ADVPN information. The ADVPN information is contained in the message using CI payload and NI payload. When receiving the Registration Request message, ADS adds this ADVPN client information to ADC information database and sends a Registration Request Response message to the ADC. The Registration Request Response message includes KP payload, which make ADC send a keepalive message to ADS periodically after registration. If the hub ADC has been registered in the ADS, CI payload SHOULD be also contained with hub's ADC information.

The registration messages with payloads are as follows:

| Client | | Server |
|-----------------------|-----|-----------------------|
| ----- | | |
| HDR, CIDHdr, CI, [NI] | --> | |
| | <-- | HDR, CIDHdr, KP, [CI] |

When the hub ADC is registered, a Hub Information message with CI payload should be sent to all the ADCs in the registration status. The hub Information Acknowledgement message has no payload, only contains ADVPN header and Client Identification Header.

The hub messages with payloads are as follows:

| Client | | Server |
|-------------|-----|-----------------|
| ----- | | |
| | <-- | HDR, CIDHdr, CI |
| HDR, CIDHdr | --> | |

4.2. Client Information Resolution

There are two scenarios that the ADC needs to send Resolution Request message to ADS to query the client information. 1. During the packet processing in the forwarding path, the session table is consulted with private IP address, If there is no session, the spoke ADC sends a Resolution Request message to ADS to get the remote ADC's information related with the private IP address. Before a Resolution Request Response comes, the data packets are forwarded to hub. Note that a Resolution Request message for the private IP address MUST NOT be triggered by every packet. 2. The spoke ADC received a Redirect message from the hub ADC, the spoke ADC sends a Resolution Request to ADS to get the remote ADC's information with the destination address.

The Resolution Request message includes DC payload. If the next hop address and destination address both are contained in the payload, the ADS should loop up ADC information database with destination address firstly.

When the ADS receives a Resolution Request packet, it searches the ADC information database by private IP address or destination address. If an ADC is found, a Resolution Response message containing CI payload and NI payload is send to the spoke ADC. If there is no match, the error code is set in the header and no payload is included in the response message.

The resolution messages with payloads are as follows:

| Client | | Server |
|-----------------|-----|-------------------------|
| ----- | | |
| HDR, CIDHdr, DC | --> | |
| | <-- | HDR, CIDHdr, [CI], [NI] |

After receiving the remote ADC's information, the ADC create session cache based on the information in the CI payload. If there is NI payload, the route towards the destination address is added in the route table.

4.3. Private Network Information Management

To help the ADC find a shortcut path, the private network information database is collected and distributed by the ADS. The private network information is like a private network route table. The ADC can query the next hop towards the private network.

In the Registration Request message, the private network information can include in the NI payload and registered to ADS. After registration on the ADS, if the ADC's network information is changed(e.g. add, delete or modify), a Network Information Registration packet including the NI payload is send to the ADS to update the private network information database. After updating, a Network Information Registration Response is send back to the ADC, the response message has no payload, only contains ADVPN header and Client Identification Header.

If the private network information is updated on ADS, a Network Information Update message containing the NI payload MUST be send to all ADCs in order to these ADCs have the correct route in their route table. All the ADC MUST send back a Network Information Update Response message to ADS, the response message has no payload, only contains ADVPN header and Client Identification Header.

The private network information can also be transferred in the Session Setup message. In the session establish process, the private network route can be added in the remote ADC's route table in order to avoid ADS query for reverse traffic.

If receiving a Session Delete message from remote ADC, the ADC MUST tear down the session and clear the route added by the Session Setup message.

4.4. Shortcut Decision

Whether the shortcut path can be established depends on the control policy in the ADS. The ADS defines the flow information to allow the direct connectivity. After the hub ADC finishes the registration, the ADS send the Shortcut Flow message contains TF payload to hub ADC. After receiving the message and add the flow information to shortcut flow table, the hub ADC send back Shortcut Flow Acknowledgement message to ADS.

The hub ADC has a shortcut flow table to match the traffic through hub in the ADVPN network. If there is a match, the hub ADC send a Redirect message to source ADVPN peer.

If the control policy is changed(e.g. add, delete or modify) on the ADS, the ADS MUST send a Shortcut Flow message to the hub ADC to update the shortcut flow table. If the shortcut flow item is deleted, the hub ADC send a Redirect message to source ADVPN peer to tear down the direct IPsec Tunnel.

4.5. Redirect protocol

The hub ADC sends a Redirect message to spoke ADC means there is another path for traffic forwarding. In the hub ADC, when the traffic matches the shortcut flow table, a Redirect message containing the Red payload is sent to spoke ADC. The spoke ADC receives the Redirect message, sends a Redirect Response message to hub ADC and subsequently sends a Resolution message to ADS to query the shortcut information. The Redirect Response message has no payload, only contains ADVPN header and Session Header.

The redirect messages with payloads are as follows:

| Hub Client | | Spoke Client |
|-------------------|-----|--------------|
| ----- | | |
| HDR, SessHdr, Red | --> | |
| | <-- | HDR, SessHdr |

If a shortcut flow is deleted in the hub ADC, the hub ADC send a Redirect message to the ADCs which has received the Redirect message to trigger the shortcut resolution before. When receiving this Redirect message, the spoke ADC deletes the route related with the flow information. The shortcut IPsec tunnel is cleared up and the

traffic goes through the hub to transfer.

4.6. Keepalive protocol

After the ADC finishes registration on the ADS, the ADC SHOULD send a Keepalive Request message to ADS to prove its liveness. The Keepalive Request message contains the ADVPN header and Client Identification Header. The number of retries and length of timeouts depend on the keepalive parameter pushed from ADS by the Registration Response message. If the number of retry attempts is reached but the ADC does not receive Keepalive Response message, the connection between ADC and ADS is considered broken. The ADC clears up all the resources and registered to ADS again.

On ADS, receipt of any ADVPN message from ADC can prove the ADC's liveness. If the timeout is reached while the ADS does not receive Registration Response message, the ADS will clear all ADC information from ADC information database. If the ADC is hub, the ADS sends the Hub Information message to all the ADCs to notify the hub removing.

4.7. Session Protocol

Session table is an remote ADC information cache in the forwarding path. It is composed of private IP address, public IP address, and the index of IPsec SA. In the forwarding process, the session table MUST be consulted with private IP address. Each session matches to only one IPsec SA.

The function of session protocol is to facilitate the forwarding process, the session information transported to the remote peer avoids to query the ADS when reverse traffic flows.

There are two type sessions: permanent session established with hub, and dynamic session established between spokes. The permanent session is static and established after the spoke ADC and hub ADC finish registration. The dynamic session will be deleted if there is no traffic between spokes,

The permanent session is established between the spoke and hub or between the hubs. After receiving the Hub's ADC information from ADS by Registration Response message or Hub Information message, the spoke establishes an IPsec Tunnel with Hub firstly and then sends a Session Setup message to the hub, which is protected via IPsec tunnel. When receiving the Session Setup message, the hub ADC create a session for spoke ADC with the spoke's ADC information, and a Session Setup Response message will be send back to the spoke ADC.

When the spoke ADC receives the Resolution Response message for

shortcut path and obtains the remote spoke ADC's information, the spoke creates a IPsec tunnel with the remote spoke. After that, the spoke sends a Session Setup message to the remote spoke. The remote spoke creates a session information towards the spoke and sends back a Session Setup Response message. If there is private network information in the source spoke, a NI payload SHOULD be contained in the Session Setup message. The Session Setup Response message has no payload, only contains ADVPN header and Session Header.

The session messages with payloads are as follows:

| Client | | Client |
|--------------------|-----|--------------|
| ----- | | |
| HDR, SessHdr, [NI] | --> | |
| | <-- | HDR, SessHdr |

If there is no traffic in the spoke-to-spoke session, the spoke will send a Session Delete message to the remote spoke to remove the session item. After the session is removed, the IPsec tunnel is also cleared up.

5. ADVPN Message Formats

This section describes the format of ADVPN message. ADVPN messages begin immediately following the UDP header. An ADVPN message is composed of a Fixed Part, a Mandatory Part and a Payload Part. The Fixed Part is common to all ADVPN message types. The Mandatory Part MUST be present, but varies depending on message type. The Payload Part also varies depending on message type.

The length of the Fixed Part is fixed at 12 octets. The length of the Mandatory Part is determined on message type. The Payload Part length depends on the payload type.

5.1. ADVPN Fixed Header

The Fixed Part of the ADVPN message contains those elements of the ADVPN message which are always present and do not vary in size with the type of message.

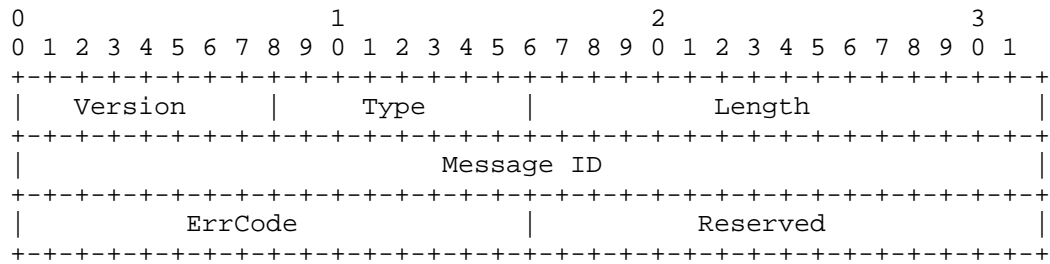


Figure 2. ADVPN Header Format

- o Version (1 octet) -- Indicates the version of the ADVPN protocol in use. Implementations based on this version of ADVPN MUST set the version to 0.
- o Type (1 octet) -- Indicates the type of ADVPN message being used.

| Message Type | Value |
|--------------------------------------|-------|
| Registration Request | 1 |
| Registration Response | 2 |
| Resolution Request | 3 |
| Resolution Response | 4 |
| Delete Request | 5 |
| Delete Response | 6 |
| Keepalive Request | 7 |
| Keepalive Response | 8 |
| Network Information Registration | 9 |
| Network Information Response | 10 |
| Shortcut Flow | 11 |
| Shortcut Flow Acknowledgement | 12 |
| Hub Information | 13 |
| Hub Information Acknowledgement | 14 |
| Redirect | 15 |
| Redirect Acknowledgements | 16 |
| Session Setup Request | 17 |
| Session Setup Response | 18 |
| Session Keepalive Request | 19 |
| Session Keepalive Response | 20 |
| Session Delete Request | 21 |
| Session Delete Response | 22 |
| Session Network Information Request | 23 |
| Session Network Information Response | 24 |

- o Length (2 octet) -- Length of total message beginning with the Version element in octets.

- o Message ID (4 octet) -- Used to provide a unique identifier for the information contained in a Request message. This value is copied directly from a Request message into the Response message. When a sender receives the Response packet, it will match the Message ID with the Request packet of local sent. When a match is found then the Request is considered to be acknowledged.

The value is incremented each time a new Request message is sent. The same value **MUST** be used when resending a Request message. It is **RECOMMENDED** that the initial value for this number be 0.

- o ErrCode (2 octet) -- An error code indicating the type of error detected, chosen from the follow list:

| Error description | Code |
|------------------------|-------|
| ----- | ----- |
| No ADC Found | 1 |
| Management Reject | 2 |
| Insufficient Resources | 3 |

- o Reserved (2 octet) -- **MUST** be zero.

5.2. Mandatory Part

Different Mandatory Part of the ADVPN message exists in different message types, and is behind the ADVPN Header.

5.2.1. Client Identification Header

The Client Identification Header, denoted CIhdr in this document, is contained in the messages that are sent and received between ADC an ADS.

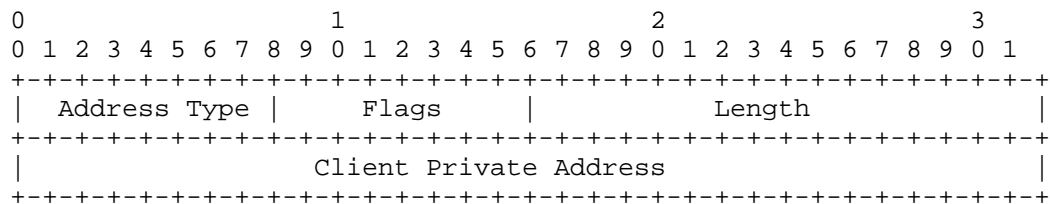


Figure 3. Client Identification Header Format

- o Address Type (1 octet) -- Identifies the address type of the client private Address.

| Address Type | Value |
|--------------|-------|
| ----- | ----- |
| IPv4 Address | 1 |
| IPv6 Address | 2 |

- o Flags (1 octet) -- The flags field is coded as follows:

```

      0               1
    0 1 2 3 4 5 6 7
+---+---+---+---+---+
|   Unused       |H|
+---+---+---+---+---+

```

H-bit - The Hub bit. When set to 1, the Client is Hub, otherwise it is spoke.

- o Length (2 octet) -- Length in octets of the current mandatory part.
- o Client Private Address (variable length) -- The ADC's logical private IP address. The value for this field is specified by the IP version field. If the message is sent from ADC to ADS, this address is private IP address of ADC. If the message is sent from ADS to ADC, this address is private IP address of destination ADC.

5.2.2. Session Header

The Session Header, denoted Sesshdr in this document, is contained in the messages that are sent and received between ADCs.

```

      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
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|  Role ID   | Address Type |           Length           |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                     Source IP Address                                     |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|                                     Destination IP Address                                    |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+

```

Figure 4. Session Header Format

- o Role ID (1 octet) -- Identifies the forwarding role of the ADC.
 - 1 - Hub.
 - 2 - Spoke.
- o Address Type (2 octet) -- Identifies the type of Source and

Destination IP Address type.

- o Length (2 octet) -- Length in octets of the current mandatory part.
- o Source IP Address (variable length) -- Identifies the sender's logical private IP address. The address type is specified by the Address Type field.
- o Destination IP Address (variable length) -- Identifies the receiver's logical private IP address. The address type is specified by the Address Type field.

5.3. Payload Part

Each payload defined in the section 5.3.1 through 5.3.6 has the general payload TLV(Type, Length, Value) format.

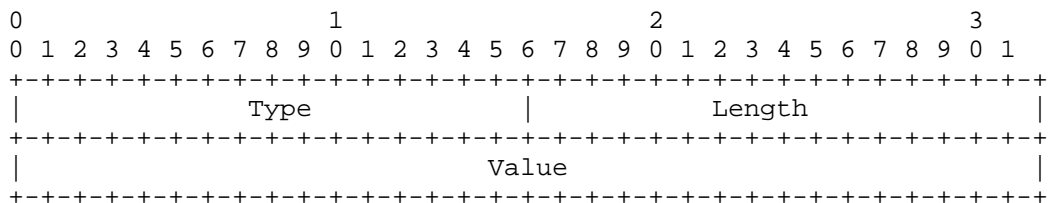


Figure 5. Payload TLV Format

The TLV fields are defined as follows:

- o Type (2 octet) -- The type of this payload.

| Payload Type | Value |
|----------------------------------|-------|
| Client Information payload (CI) | 1 |
| Destination Client Payload (DC) | 2 |
| Network Information Payload (NI) | 3 |
| Traffic Flow Payload (TF) | 4 |
| Keepalive Parameter Payload (KP) | 5 |
| Redirect Payload (Red) | 6 |

- o Length (2 octet) -- Length in octets of the current payload, including the type and length.
- o Value (variable octet) -- The value of this payload. Each payload has different content.

5.3.1. Client Information Payload

The Client Information Payload, denoted CI in this document, is used to be as part of Registration message to notify the ADS of ADC's information, or as part of Resolution Response message to notify the ADC of the remote ADC's information.

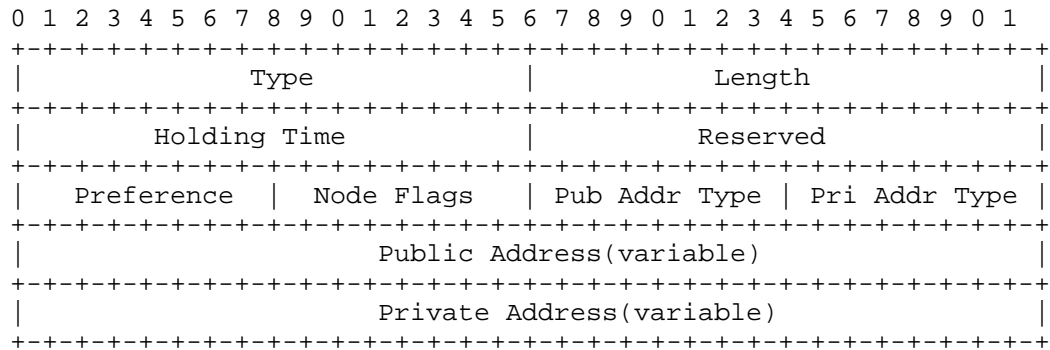
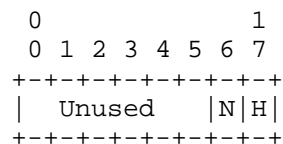


Figure 6. Client Information Payload Format

- o Holding Time (2 octet) -- Identifies the expire time(seconds) of the ADC's information obtained from ADS. The ADS SHOULD ignore this field when receiving the Registration Request message.
- o Reserved (2 octet) -- MUST be sent as zero; MUST be ignored on receipt.
- o Preference (1 octet) -- Identifies the ADC's preference. Higher values in the range 1 to 255 indicates higher preference. A zero value indicates no preference.
- o Node Flags (1 octet) -- The flags field is coded as follows:



H-bit - The hub bit. When set to 1, this current ADC is a hub, otherwise a spoke.

N-bit - The NAT bit. When set to 1, it indicates the ADC is behind the NAT.

- o Pub Addr Type (1 octet) -- Identifies the ADC's Public Address type.

1 -- IPv4

- 2 -- IPv6
- o Pri Addr Type (1 octet) -- Identifies the ADC's Private Address type.
 - 1 -- IPv4
 - 2 -- IPv6
- o Public Address (variable length) -- Identifies the ADC's IPsec peer address. The type of address for this field is specified by the Pub Addr Type field.
- o Private Address (variable length) -- Identifies the ADC's logical private IP address. The values for this field is specified by the Pri Addr Type field.

5.3.2. Destination Client Payload

The Destination Client Payload, denoted DC in this document, is used to be search key to discover the remote ADC's information.

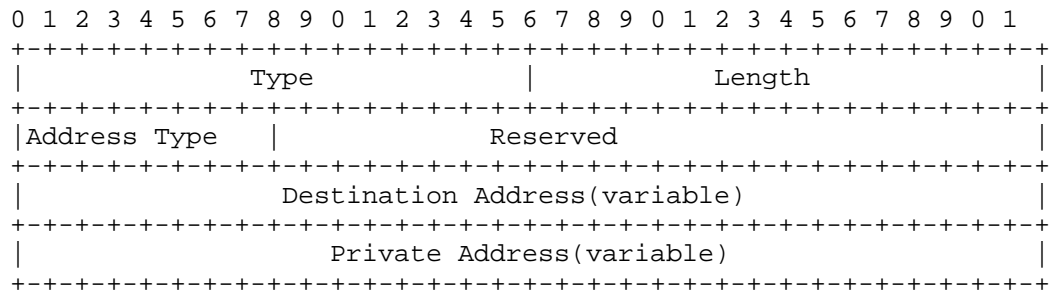


Figure 7. Destination Client Payload Format

- o Address Type (1 octet) -- Identifies the type of destination address and next hop address.
 - 1 -- IPv4
 - 2 -- IPv6
- o Reserved (3 octet) -- MUST be zero.
- o Destination IP Address (variable length) -- Identifies the destination IP address of communication. The value for this field is specified by the Address Type field.

- o Private Address (variable length) -- Identifies the next hop address to the destination network. The value for this field is specified by the Address Type field.

5.3.3. Network Information Payload

The Network Information Payload, denoted NI in this document, is used to carry private network information.

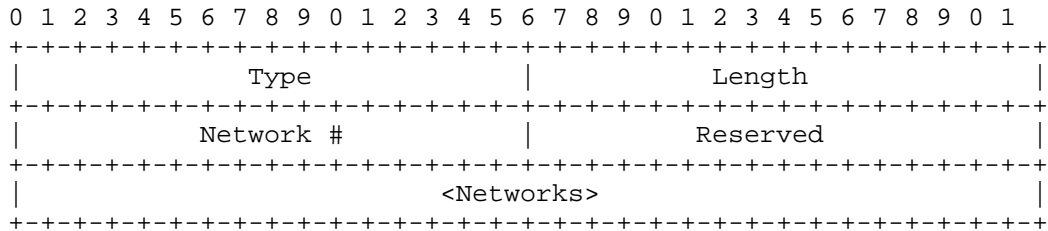


Figure 8. Network Information Payload Format

- o Network # (2 octet) -- Identifies the number of network this message contains.
- o Reserved (2 octet) -- MUST be zero.

Each network has the following formats:

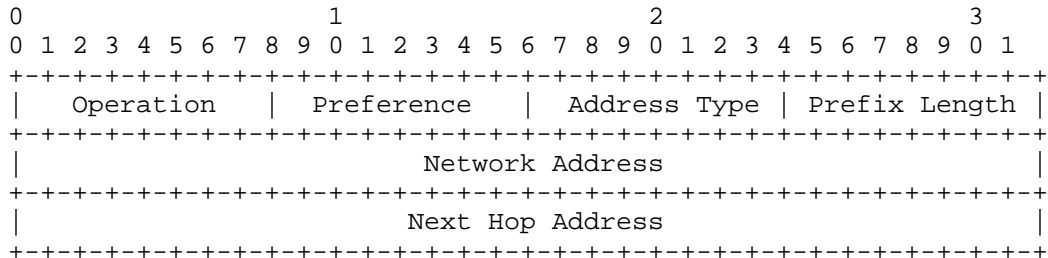


Figure 9. Network Format

- ```
o Operation (1 octet) -- Indicates the operation for the current
 network.

 0 - Reserved.

 1 - Add.

 2 - Delete.
```

3 - Update.

- o Preference (1 octet) -- Indicates the priority of network.
- o Address Type (1 octet) -- Indicates the Address type of network address and next hop address.

1 - IPv4 Address.

2 - IPv6 Address.

- o Prefix Length (1 octet) -- Is the octet length of the routing prefix.
- o Network Address (variable length) -- Identifies the address of the ADC's private network. The value for this field is specified by the Address Type field.
- o Next Hop Address(variable length) -- Identifies the next hop to the Network Address in the routing path. The Next Hop is also the ADC's private IP address. The value for this field is specified by the Address Type field.

#### 5.3.4. Traffic Flow Payload

The Traffic Flow Payload is denoted TF in this document. This payload contains the data flow information.



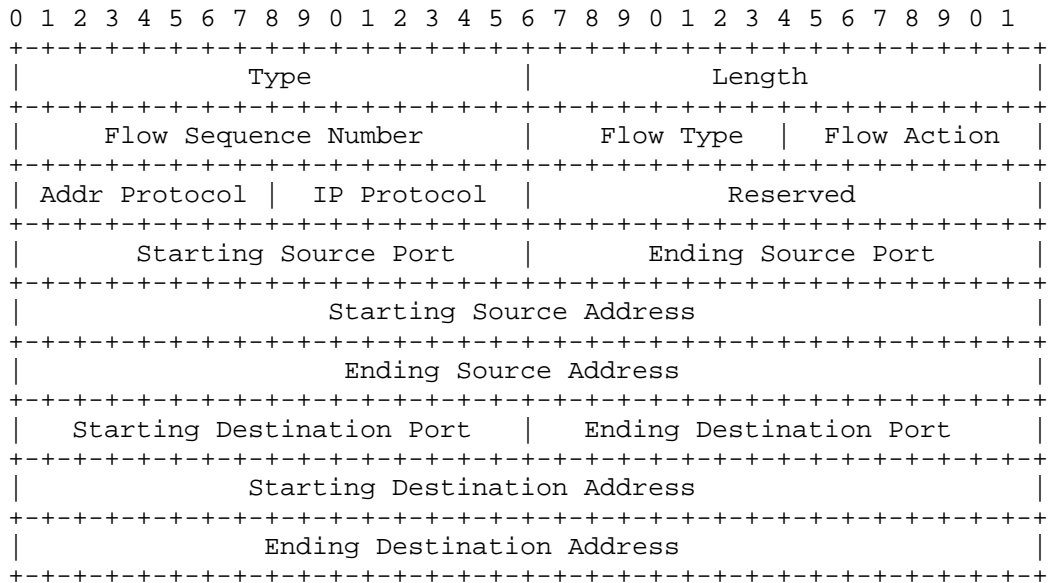


Figure 10. Shortcut Flow Payload Format

- o Flow Sequence Number (2 octet) -- Identifies the preference of the data flow. Lower values (in the range 0 to 65534) indicates higher preference.
- o Flow Type (1 octet) -- Identifies the traffic flow type.
  - 1 - Shortcut Data Flow.
- o Flow Action (1 octet) -- Identifies which action will to do when matching this flow.
  - 1 - Permit. The packet which matching the flow will trigger the specified function, such as redirection.
  - 2 - Deny. The packet which matching the flow will not trigger any function.
  - 3 - Discard. The packet which matching the flow will be discarded.
- o Addr Type (1 octet) -- Identifies which the data flow address type is, IPv4 or IPv6.
  - 1 - IPv4 Address.
  - 2 - IPv6 Address.

- o IP Protocol (1 octet) -- Identifies IP protocol of this data flow, such as UDP, TCP or ICMP etc.
- o Reserved (2 octet) -- MUST be zero.
- o Starting Source Port (2 octet) -- Value specifying the smallest source port number allowed.
- o Ending Source Port (2 octet) -- Value specifying the largest source port number allowed.
- o Starting Source Address (2 octet) -- The smallest source address.
- o Ending Source Address (2 octet) -- The largest source address.
- o Starting Destination Port (2 octet) -- Value specifying the smallest destination port number allowed.
- o Ending Destination Port (2 octet) -- Value specifying the largest destination port number allowed.
- o Starting Destination Address (2 octet) -- The smallest destination address.
- o Ending Destination Address (2 octet) -- The largest destination address.

### 5.3.5. Keepalive Parameter Payload

The Keepalive Parameter Payload is denoted KP in this document. This payload contains the parameter to sending keepalive message.

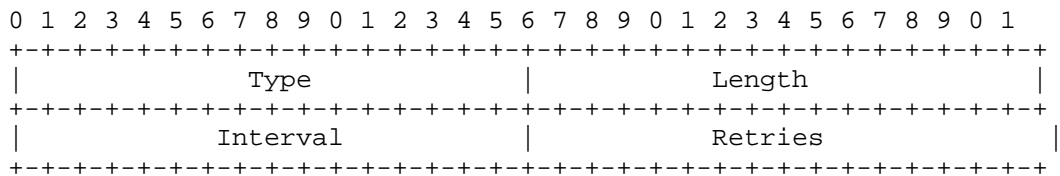


Figure 11. Keepalive Parameter Payload Format

- o Interval (2 octet) -- Identifies the timeout(seconds) of sending a Keepalive Request again.
- o Retries (2 octet) -- Identifies the number of retry attempts.

### 5.3.6. Redirect Payload

The Redirect Payload is denoted Red in this document. This payload contains the original packet information. The original packet is used for the spoke to send a Resolution Request packet to the ADS to get the peer's ADVPN information.

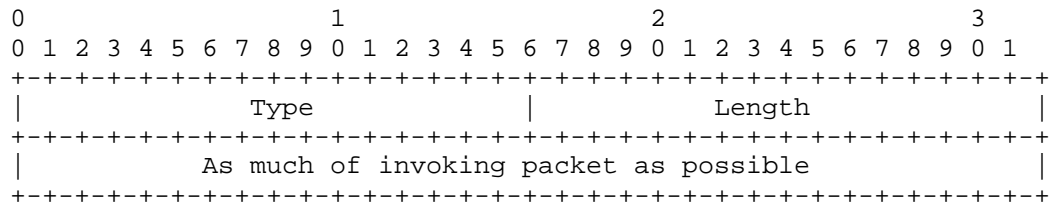


Figure 12. Redirect Payload Format

## 6. Implementation Status

This section records the status of known implementations of the protocol defined by this specification at the time of posting of this Internet-Draft, and is based on a proposal described in RFC 6982. The description of implementations in this section is intended to assist the IETF in its decision processes in progressing drafts to RFCs. Please note that the listing of any individual implementation here does not imply endorsement by the IETF. Furthermore, no effort has been spent to verify the information presented here that was supplied by IETF contributors. This is not intended as, and must not be construed to be, a catalog of available implementations or their features. Readers are advised to note that other implementations may exist.

According to RFC 6982, "this will allow reviewers and working groups to assign due consideration to documents that have the benefit of running code, which may serve as evidence of valuable experimentation and feedback that have made the implemented protocols more mature. It is up to the individual working groups to use this information as they see fit".

This draft is based on H3C/HP DVPN(Dynamic Virtual Private Network) solution. DVPN has been widely used since 2005. DVPN solution is not a single protocol, but an architecture. DVPN helps enterprises simplify the configuration and management of IPsec VPN tunnels. DVPN policies share security access, management, and quality of service (QoS) policies to easily connect thousands of remote branch and regional offices to the corporate headquarters or data centers. Administrators no longer need to login to each VPN device to manually set up site-to-site VPN tunnels at each branch or regional office, corporate headquarters, and data centers. DVPN is an innovation to simplify secure WAN connectivity for the enterprise.

DVPN is a complete and cost-effective solution that is ideal for the hub-and-spoke topology, the most common topology for enterprises, where you also have an option for mesh connectivity. DVPN spans across various domains such as routing, security, and address management.

The H3C/HP DVPN website link is:  
<http://h17007.www1.hp.com/us/en/networking/solutions/technology/dvpn/index.aspx>.

Although this ADVPN protocol comes from DVPN solution, it has been simplified and optimized to meet the requirements. The main differences is:

- o VAM(VPN Address Management) protocol and DVPN protocol is merged

into a single ADVPN protocol.

- o Security mechanism in VAM protocol is deleted, the ADVPN protocol should be protected by IPsec.
- o In DVPN, all packets and frames must be encapsulated in GRE first, and then be protected by IPsec. However, the ADVPN supports a pure IPsec encapsulation.

## 7. Security Considerations

The ADVPN protocol has no protocol-internal security mechanism, it relies on other security protocol to protect the ADVPN messages.

The messages between the ADC and ADS can be protected by the IPsec or SSL/TLS. The messages between ADCs is protected by IPsec.

It is highly recommended that the wildcard pre-shared-key in IKEv1 or IKEv2 is not used in the ADVPN, the attacker can access the ADVPN network if one ADVPN peer is compromised.

## 8. IANA Considerations

IANA may need to allocate additional values for the options presented in this document. The values of the protocol field needed to be assigned from the numbering space.

## 9. Acknowledgments

## 10. References

### 10.1. Normative References

- [KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC1776] Crocker, S., "The Address is the Message", RFC 1776, April 1 1995.
- [TRUTHS] Callon, R., "The Twelve Networking Truths", RFC 1925, April 1 1996.
- [ADVPN\_Problem]  
Hanna, S., "Auto Discovery VPN Problem Statement and Requirements", draft-ietf-ipsecme-p2p-vpn-problem-07.txt, June 2013.
- [IPSECARCH]  
Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

## 10.2. Informative References

- [NHRP] J. Luciani, "NBMA Next Hop Resolution Protocol (NHRP)", RFC2332, April 1998.
- [RFC5513] Farrel, A., "IANA Considerations for Three Letter Acronyms", RFC 5513, April 1 2009.
- [RFC5514] Vyncke, E., "IPv6 over Social Networks", RFC 5514, April 1 2009.

## Appendix A. Comparison Against ADVPN Requirements

## Requirement #1:

In this ADVPN protocol, each ADC only needs to configure its own SPD and PAD. Adding or removing an ADC in the ADVPN topology does not need configuration change for any other ADC.

## Requirement #2:

Each ADC registers its public address(peer address) and private address to ADS, and the other ADC query the ADS to obtain the public address. Therefore, even the public address of one ADC is updated every time the device comes up, the other ADC can communicate with it without any configuration change.

## Requirement #3:

The tunneling protocol(e.g. GRE) and routing protocol(e.g. OSPF, BGP) can run over the spoke-to-hub and spoke-to-spoke IPsec tunnel with minimal configuration. These protocols do not need to aware of IPsec tunnel, and also IPsec does not need to be aware of these protocol packets.

## Requirement #4:

The ADS is the controller of the ADVPN network. It has control policy which decides whether the spoke can be allowed to have a direct communication with other spoke. The control policy is pushed to hub from the ADS after the hub ADC finishes registration. The detailed description about it is given in Section 4.4.

## Requirement #5:

To mitigate the affect of compromised ADVPN peer, each ADVPN peer SHOULD have unique and long term authentication credential such as

certificate used for IKEv1 and IKEv2 negotiation. The wildcard pre-shared-key SHOULD NOT be used for ADVPN connection.

Requirement #6:

When the endpoint roams, if its public address changes, it will be as a new ADC to rejoin ADVPN network. After re-registering to ADS, it connects the hub gateway again. The data traffic can be transferred through new spoke-to-hub IPsec tunnel and spoke-to-spoke IPsec tunnel.

Requirement #7:

The information of hub ADCs is maintained by ADS. if the endpoints roams across the hub gateway, the new hub ADC' information will be pushed to the spoke ADCs, and new IPsec tunnel is established between the ADC and new hub ADC. The traffic is migrated from one gateway to another gateway.

Requirement #8:

All the ADVPN peers can be located behind NAT boxes. After ADCs registration, the ADS detects which ADC is behind NAT box. The ADS updates the public IP address/Port information of spoke ADC with the modifying IP address/Port by NAT box from hub ADC after the IPsec tunnel is established between the spoke and hub. Therefore, even two spokes are both behind NAT boxes, they can also establish the direct connectivity.

Requirement #9:

All the tables in the ADVPN protocol are reportable and manageable, such as IP Address Mapping Database, Private Network Information Database, Session Table and Shortcut Flow Table etc. Change of IPsec SA such as establishment and expiration can be logged and reported as necessary events. This document does not create a MIB.

Requirement #10:

To support allied and federated environments, the ADVPN peer SHOULD use the certificate as the credential for connections. With the PKI trust architecture, the ADVPN peer from different organizations can be able to connect to each other.

Requirement #11:

The ADS is the controller of ADVPN network. The administrator can configure the control policy on ADS to determine the ADVPN network is

a Star, Full mesh or a partial full mesh topology.

Requirement #12:

The ADVPN protocol can cooperated with IGMP and PIM to provide multicast function. PIM packets run over spoke-to-hub IPsec tunnel to establish the PIM neighbor. Multicast traffic is selective replicated to spokes on the hub.

Requirement #13:

In the ADVPN protocol, all the changes such as IPsec SA establishment, shortcut route injection etc. can be monitored, logged and reported to help trouble shooting.

Requirement #14:

When L3VPN runs over IPsec tunnel, GRE or other tunnel protocol can be transport-link protocol. These tunneling protocols can run over the spoke-to-hub and spoke-to-spoke IPsec tunnel in ADVPN network.

Requirement #15:

The ADC can register its QoS policy information to ADS, and when the other ADC query the ADC's information, it can obtain the related QoS policy information.

Requirement #16:

In the ADVPN protocol, the administrator can specify more than one hub in the ADS for the ADC. The ADC gets all the hub ADC information after registration and establishes IPsec tunnel with each hub ADC.

Authors' Addresses

Yu Mao(Toby Mao)  
Hangzhou H3C Tech. Co., Ltd.  
Oriental Electronic Bld., No. 2  
Chuangye Road  
Shang-Di Information Industry  
Hai-Dian District  
Beijing 100085  
China

EMail: yumao9@gmail.com

ZhanQun Wang  
Hangzhou H3C Tech. Co., Ltd.



Oriental Electronic Bld., No. 2  
Chuangye Road  
Shang-Di Information Industry  
Hai-Dian District  
Beijing 100085  
China

EMail: wangzhanqun.ietf@gmail.com

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

Email: vishwas.manral@hp.com

IPSECME  
Internet-Draft  
Intended status: Standards Track  
Expires: January 06, 2014

D. Migault (Ed)  
Francetelecom - Orange  
July 05, 2013

KEEP\_OLD\_IKE\_SA Extension  
draft-mglt-ipsecme-keep-old-ike-sa-00.txt

## Abstract

This document considers a VPN Client setting a VPN with a security gateway where at least one of the peer has multiple interfaces.

With the current IKEv2, the outer IP addresses of the VPN are determined by those used by IKEv2 channel. As a result using multiple interface requires to set an IKEv2 channel on each interface, and then on each paths if both the VPN Client and the security gateway have multiple interfaces. Setting multiple IKEv2 channel involves multiple authentications which MAY each require multiple round trips and delay the VPN establishment. In addition multiple authentications unnecessarily load the VPN client and the authentication infrastructure.

This document presents the KEEP\_OLD\_IKE\_SA extension, where an additional IKEv2 channel from an already authenticated IKEv2 channel. The newly created IKEv2 channel is set without the IKEv2 authentication exchange. The newly created IKEv2 channel can then be assigned to another interface using MOBIKE.

## 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 06, 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. Requirements notation . . . . .                         | 2  |
| 2. Introduction . . . . .                                  | 3  |
| 3. Terminology . . . . .                                   | 4  |
| 4. Protocol Overview . . . . .                             | 4  |
| 5. Payload Description . . . . .                           | 6  |
| 6. IANA Considerations . . . . .                           | 7  |
| 7. Security Considerations . . . . .                       | 7  |
| 8. Acknowledgment . . . . .                                | 7  |
| 9. References . . . . .                                    | 7  |
| 9.1. Normative References . . . . .                        | 7  |
| 9.2. Informational References . . . . .                    | 8  |
| Appendix A. Document Change Log . . . . .                  | 8  |
| Appendix B. Setting a VPN on Multiple Interfaces . . . . . | 8  |
| B.1. Setting VPN_0 . . . . .                               | 8  |
| B.2. Creating an additional IKEv2 Channel . . . . .        | 10 |
| B.3. Creation of the Child SA for VPN_1 . . . . .          | 11 |
| B.4. Moving VPN_1 on Interface_1 . . . . .                 | 12 |
| B.5. Reduced Exchange . . . . .                            | 13 |
| Author's Address . . . . .                                 | 14 |

## 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

This document considers a VPN End User setting its VPN with a Security Gateway, and at least one of the peers has multiple interfaces. Figure 1 represents the case where the VPN has multiple interfaces, figure 2 represents the case where the Security Gateway has multiple interfaces, and figure 3 represents the case where both the VPN End User and the Security Gateway has multiple interfaces. With figure 1 and figure 2, one of the peer has  $n = 2$  interfaces and the other has a single interface. This results in the creating of up to  $n = 2$  VPNs. With figure 3, the VPN End User has  $n = 2$  interfaces and the Security Gateway has  $m = 2$  interfaces. This can lead to up to  $m \times n$  VPNs.

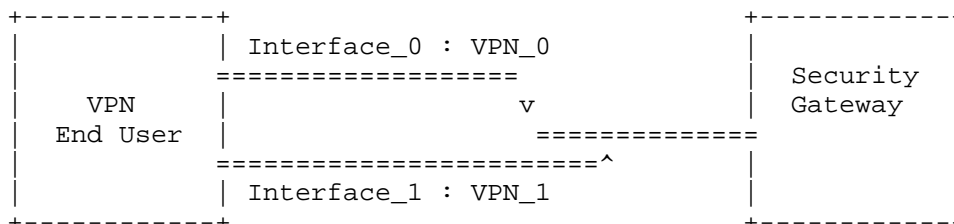


Figure 1: VPN End User with Multiple Interfaces

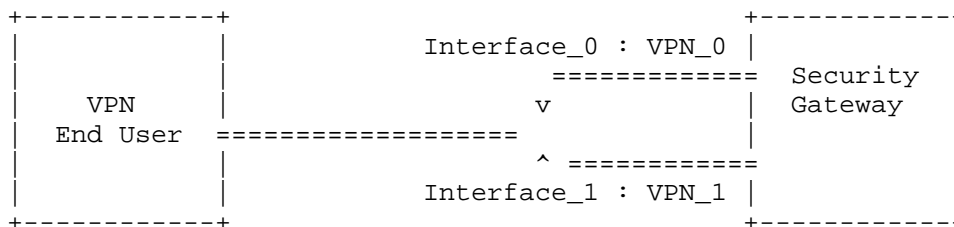


Figure 2: Security Gateway with Multiple Interfaces

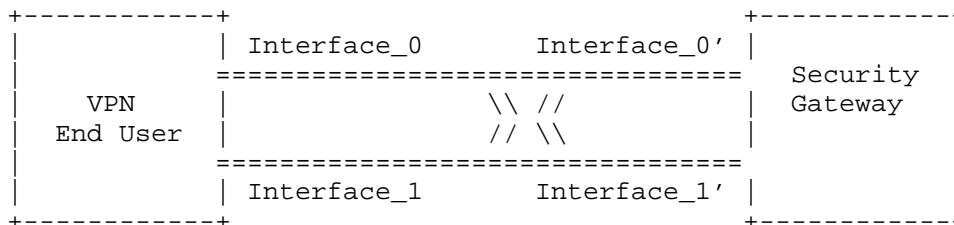


Figure3: VPN End User and Security Gateway

## with Multiple Interfaces

With the current IKEv2 [RFC5996], each VPN requires an IKEv2 channel, and setting an IKEv2 channel requires an authentication. Authentication can involve multiple round trips like EAP-SIM [RFC4186] as well as crypto operations that MAY delay the connectivity.

This document presents the KEEP\_OLD\_IKE\_SA extension. The main idea is that the peer with multiple interfaces sets an first authenticated IKEv2 channel. Then it takes advantage of this authentication and derives as many parallel IKEv2 channels as VPNs. On each IKEv2 channel a VPN is negotiated. This results in parallel VPNs. Then the VPN End User moves the VPNs to their proper places using MOBIKE. Alternatively, the VPN End User can also move the IKEv2 channels and then negotiate the VPNs.

[I-D.mglt-mif-security-requirements] provides a problem statement for IPsec and multiple interfaces. [I-D.arora-ipsecme-ikev2-alt-tunnel-addresses] and [I-D.mglt-ipsecme-alternate-outer-address] have been proposed so tunnel outer IP address can differ from those of the IKEv2 channel. The advantage of the KEEP\_OLD\_IKE\_SA extension is that it requires very few modification to already existing IKEv2 implementation. Then, it is reusing already existing and widely deployed protocol such as MOBIKE [RFC4555]. Finally by keeping a dedicated IKEv2 channel for each VPN, it eases reachability tests.

### 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. Protocol Overview

The goal of the document is to specify how to create a new IKEv2 channel. IKEv2 [RFC5996] specifies the CREATE\_CHILD\_SA that makes possible to rekey an IKE\_SA, create or rekey a new Child SA.

The difference between rekeying an IKE\_SA and creating a new IKE\_SA is that the old IKE\_SA MUST NOT be deleted, either by starting a Delete exchange or removing the IKE\_SA without the Delete exchange.

Note that IKEv2 [RFC5996] Section 1.3.2 or Section 2.18 does not explicitly mentions that the old IKE\_SA MUST be deleted. However, there are currently no signaling advertising the IKE\_SA has not been deleted. The purpose of this document is to avoid this uncertainty when rekeying the IKE\_SA. In other words, the document avoids that one peer expects a additional IKE\_SA to be created whereas the other simply proceed to a replacement of the old IKE\_SA.

Currently, one MAY check whether or not the old IKE\_SA has been deleted or not by waiting a for a given time and then initiate and empty INFORMATIONAL exchange using the old IKE\_SA. The absence of response MAY indicate the old IKE\_SA has been removed.

This document introduces KEEP\_OLD\_IKE\_SA Notify Payload. The initiator sends the KEEP\_OLD\_IKE\_SA Notify Payload in a CREATE\_CHILD\_SA request for rekeying the IKE\_SA. The KEEP\_OLD\_IKE\_SA Notify Payload is placed before the concerned SA and indicates what is expected for the old IKE\_SA. Motivation of this draft is to indicate the old IKE\_SA MUST NOT be deleted once the new IKE\_SA is rekeyed. Alternatively, the initiator MAY use the KEEP\_OLD\_IKE\_SA Notify Payload to indicate the old IKE\_SA is not expected to be re-used.

Initiator

Responder

-----  
HDR, SK {N(KEEP\_OLD\_IKE\_SA) SA, Ni, KEi} -->

The responder finds a KEEP\_OLD\_IKE\_SA, if it does not understand the extension it ignores the payload as defined in [RFC5996] Section 3.10.1. Similarly, the KEEP\_OLD\_IKE\_SA Notify Payload MUST be ignored if the CREATE\_CHILD\_SA request does not concern a IKE\_SA rekey. If the initiator wants to check whether the IKE\_SA has been deleted or not, it SHOULD proceed to additional empty INFORMATIONAL exchange as described in [RFC5996] Section 2.4. In this case, the responder's response will be:

<-- HDR, SK {SA, Nr, KEr}

In any other case, the responder understands the KEEP\_OLD\_IKE\_SA Notify Payload and the CREATE\_CHILD\_SA request concerns a IKE\_SA rekey. The responder MUST proceed to the IKE\_SA rekey. If the KEEP\_OLD\_IKE\_SA indicates the old IKE\_SA MUST be kept, the responder MUST keep the old IKE\_SA active. Alternatively, if it indicates the old IKE\_SA is not supposed to be used, the responder MAY delete the old IKE\_SA after a given time out. The responder MUST respond and indicate if the old IKE\_SA has been kept or not. The exchange will be:

```
<-- HDR, SK { N(KEEP_OLD_IKE_SA)
 SA, Nr, KEr}
```

## 5. Payload Description

Figure 7 illustrates the KEEP\_OLD\_IKE\_SA Notify Payload packet format.

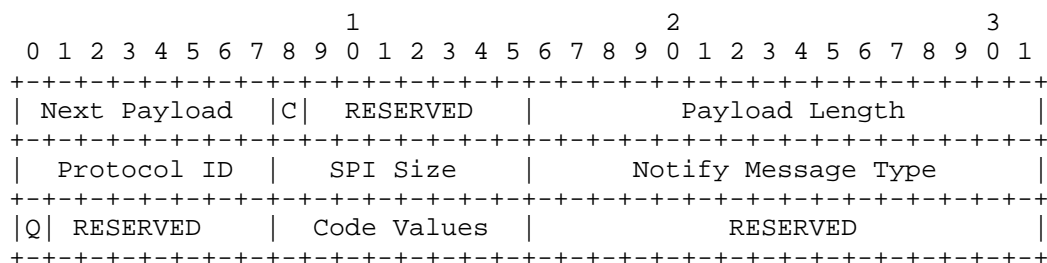


Figure 7: KEEP\_OLD\_IKE\_SA Notify Payload

- 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.
- RESERVED (7 bits): MUST be set 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. It is set to KEEP\_OLD\_IKE\_SA in this document.
- Question Bit (1 bit): set to one by the initiator and set to zero by the responder.
- RESERVED (7 bits): set to zero.
- Code Values: Code that indicates what action is expected to be done with the newly negotiated IKE\_SA.

#### Code Values

```

Keep Old IKE_SA 0
Unused Old IKE_SA 1
Unassigned 2-255
```

## 6. IANA Considerations

The new fields and number are the following:

#### IKEv2 Notify Message Types - Status Types

```

KEEP_OLD_IKE_SA - TBD
```

## 7. Security Considerations

The protocol defined in this document does not modifies IKEv2. It signalizes what has been implementation dependent on how to manage an old IKE\_SA after a rekey.

## 8. Acknowledgment

The ideas of this draft came from various inputs from the ipsecme and discussions with Tero Kivinen and Michael Richardson.

## 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.
- [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.

## 9.2. 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-alternate-outer-address]  
Migault, D., "IKEv2 Alternate Outer IP Address Extension", draft-mglt-ipsecme-alternate-outer-address-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.
- [RFC4186] Haverinen, H. and J. Salowey, "Extensible Authentication Protocol Method for Global System for Mobile Communications (GSM) Subscriber Identity Modules (EAP-SIM)", RFC 4186, January 2006.

## Appendix A. Document Change Log

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

-00: First version published.

## Appendix B. Setting a VPN on Multiple Interfaces

This section is informational and exposes how a VPN End User as illustrated in Figure 1 can build two VPNs on its two interfaces without multiple authentications. Other cases represented in figure 2 and 3 are similar and can be easily derived from the case. The mechanism is based on the KEEP\_OLD\_IKE\_SA extension and the MOBIKE extension [RFC4555].

### B.1. Setting VPN\_0

First, the VPN End User negotiates a VPN using one interface. This involves a regular IKEv2 setting. In addition, the VPN End User and

the Security Gateway advertise they support MOBIKE. At the end of the exchange, VPN\_0 is set as represented in figure 4.

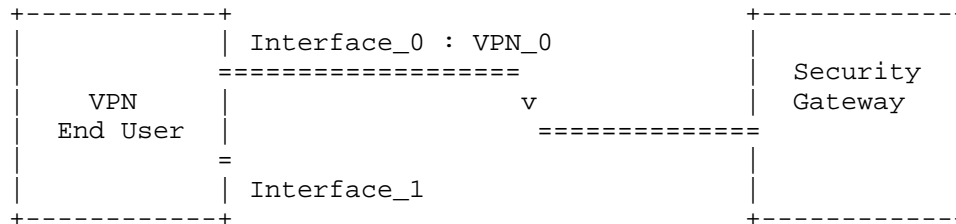


Figure 4: VPN End User Establishing VPN\_0

The exchange is completely described in [RFC4555]. First the negotiates the IKE\_SA. In the figure below peers also proceed to NAT detection because of the use of MOBIKE.

```

Initiator Responder

(IP_I1:500 -> IP_R1:500)
HDR, SAi1, KEi, Ni,
 N(NAT_DETECTION_SOURCE_IP),
 N(NAT_DETECTION_DESTINATION_IP) -->

<-- (IP_R1:500 -> IP_I1:500)
 HDR, SAR1, KEr, Nr,
 N(NAT_DETECTION_SOURCE_IP),
 N(NAT_DETECTION_DESTINATION_IP)

```

The initiators and the responder proceed to the authentication exchange, advertise they support MOBIKE and negotiate the SA for VPN\_0. Optionally, the initiator and the Security Gateway MAY advertise their multiple interfaces using the ADDITIONAL\_IP4\_ADDRESS and/or ADDITIONAL\_IP6\_ADDRESS Notify Payload

```

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

<-- (IP_R1:4500 -> IP_I1:4500)
 HDR, SK { IDr, CERT, AUTH,
 CP(CFG_REPLY),

```

```

Sar2, TSi, TSr,
N(MOBIKE_SUPPORTED),
N(ADDITIONAL_IP*_ADDRESS)+}

```

## B.2. Creating an additional IKEv2 Channel

In our case the the initiator wants to set establish a VPN with its Interface\_1 between the VPN End User and the Security Gateway. The VPN End User will first establish a parallel IKE\_SA using a CREATE\_CHILD\_SA that concerns an IKE\_SA rekey associated to a KEEP\_OLD\_IKE\_SA Notify Payload. This results in two different IKE\_SA between the VPN End User and the Security Gateway. Currently both IKE\_SA are set using Interface 0 of the VPN End User.

In this section we consider the creation of the additional IKE\_SA as a separate exchange. However, there are several situations where this extra round trips MAY be avoided. First if the VPN End User knows multiple interfaces MAY be involved, it can combine this exchange with the previous one (IKE\_AUTH, CREATE\_CHILD\_SA concerning the creation of the SA). Secondly, the Security Gateway MAY also start the CREATE\_CHILD\_SA exchange to create an additional IKE\_SA. This reduces the delay to half a round trip.

The CREATE\_CHILD\_SA exchange to create an additional IKE\_SA MAY be combined with the IKE\_AUTH exchange if the VPN End User estimates with a high probability that multiple interfaces MAY be involved in the communication. This MAY be the case if the VPN End User has multiple interfaces, or if the VPN End User guesses that the Security Gateway has multiple interfaces. In the case the KEEP\_OLD\_IKE\_SA Notify Payload is not supported by the Security Gateway or that the Security Gateway has only one interface, this will result in rekeying the IKE\_SA, and thus does not compromise the communication.

Similarly, the CREATE\_CHILD\_SA exchange to create an additional IKE\_SA MAY be initiated by the responder and combined with the IKE\_AUTH exchange if the Security Gateway wants to reduce the number of round trips, and supposes the VPN End User will use its multiple interfaces. Note that the Security Gateway knows if multiple interfaces are involved in the communication. What remains uncertain is whether the VPN End User has the ability to use these multiple interfaces simultaneously.

Initiator

Responder

```

(IP_I1:4500 -> IP_R1:4500)
HDR, SK { N(KEEP_OLD_IKE_SA),

```

```

SA, Ni, KEi} -->
<-- (IP_R1:4500 -> IP_I1:4500)
 HDR, SK { N(KEEP_OLD_IKE_SA),
 SA, Nr, KEr}

```

### B.3. Creation of the Child SA for VPN\_1

Once the new IKEv2 channel has been created, the VPN End User MAY initiate a CREATE\_CHILD\_SA exchange that concerns the creation of a Child SA for VPN\_1. The newly created VPN\_1 will use Interface\_0 of the VPN End User.

It is out of scope of the document to define how the VPN End User MAY handle traffic with its multiple interfaces. The VPN End User MAY use the same IP inner address on its multiple interfaces. In this case, the same Traffic Selectors (that is the IP address used for VPN\_0 and VPN\_1) MAY match for both VPNs VPN\_0 and VPN\_1. The End User VPN SHOULD be aware of such match and be able to manage it. It MAY for example use distinct Traffic Selectors on both VPNs using different ports, manage the order of its SPD or have SPD defined per interfaces. Defining these mechanisms are out of scope of this document. Alternatively, the VPN End User MAY uses a different IP address for each interface. In the latter case, if the inner IP address is assigned by the Security Gateway, the Configuration Payload (CP) MUST be placed before the SA Payload as specified in [RFC5996] Section 2.19.

The creation of VPN\_1 is performed via the newly created IKE\_SA as follows:

| Initiator                              | Responder                            |
|----------------------------------------|--------------------------------------|
| -----                                  |                                      |
| (IP_I1:4500 -> IP_R1:4500)             |                                      |
| HDR(new), SK(new) { [CP(CFG_REQUEST)], |                                      |
| SAi2, TSi, TSr } -->                   |                                      |
|                                        | <-- (IP_R1:4500 -> IP_I1:4500)       |
|                                        | HDR(new), SK(new) { [CP(CFG_REPLY)], |
|                                        | SAr2, TSi, TSr}                      |

The resulting configuration is depicted in figure 5. VPN\_0 and VPN\_1 have been created, but both are using the same Interface: Interface\_0.

```

+-----+
| | Interface_0 : VPN_0, VPN_1 | |
+-----+

```

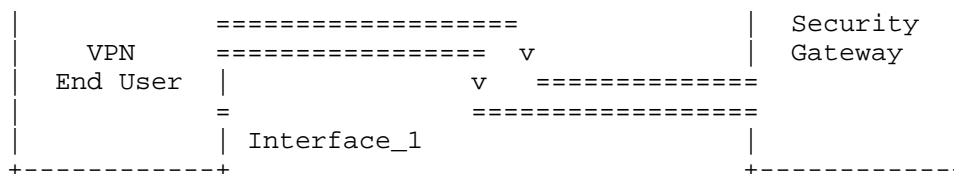


Figure 5: VPN End User Establishing VPN\_0 and VPN\_1

#### B.4. Moving VPN\_1 on Interface\_1

In this section, MOBIKE is used to move VPN\_1 on interface\_1. The exchange is described in [RFC4555]. All exchanges are using the new IKE\_SA. Eventually, the VPN End User MAY check if the Security Gateway is reachable via Interface\_1. The exchanges are described below:

```

Initiator Responder

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

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

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

(IP_I2:4500 -> IP_R1:4500)
HDR(new), SK(new) { N(UPDATE_SA_ADDRESSES),
 N(NAT_DETECTION_SOURCE_IP),
 N(NAT_DETECTION_DESTINATION_IP),
 N(COOKIE2) } -->

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

```

This results in the situation as described in figure 6.

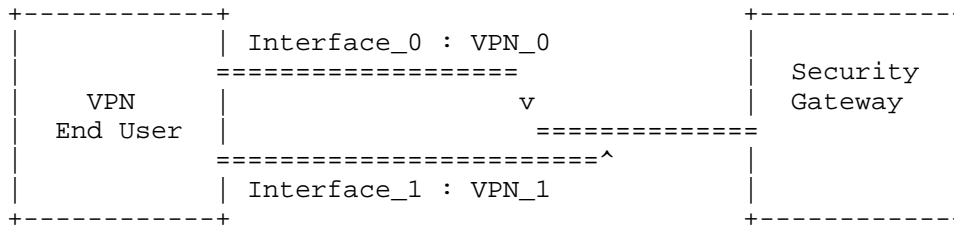


Figure 6: VPN End User with Multiple Interfaces

### B.5. Reduced Exchange

The previous sections detail the various exchanges between the VPN End User and the Security Gateway. This section shows an example where the number of exchanges are limited, thus limiting the delay to set up a multiple interface VPN communication.

```

Initiator Responder

(IP_I1:500 -> IP_R1:500)
HDR, SAi1, KEi, Ni,
 N(NAT_DETECTION_SOURCE_IP),
 N(NAT_DETECTION_DESTINATION_IP) -->

<-- (IP_R1:500 -> IP_I1:500)
 HDR, SAR1, KEr, Nr,
 N(NAT_DETECTION_SOURCE_IP),
 N(NAT_DETECTION_DESTINATION_IP)

(IP_I1:4500 -> IP_R1:4500)
HDR, SK { IDi, CERT, AUTH,
 CP(CFG_REQUEST),
 SAi2, TSi, TSr,
 N(MOBIKE_SUPPORTED),
 N(ADDITIONAL_IP*_ADDRESS)+,
 N(KEEP_OLD_IKE_SA),
 SA, Ni, KEi} -->

<-- (IP_R1:4500 -> IP_I1:4500)
 HDR, SK { IDr, CERT, AUTH,
 CP(CFG_REPLY),
 SAR2, TSi, TSr,
 N(MOBIKE_SUPPORTED),
 N(ADDITIONAL_IP*_ADDRESS)+},
 N(KEEP_OLD_IKE_SA),
 SA, Nr, KEr}

```

```
 <-- (IP_R1:4500 -> IP_I2:4500)
 HDR(new), SK(new)
 { [CP(REQUEST)],
 SAi2, TSi, TSr,
 N(UPDATE_SA_ADDRESSES)}
(IP_I2:4500 -> IP_R1:4500)
HDR(new), SK(new) { [CP(CFG_REPLY)],
 SAr2, TSi, TSr}
-->
```

## 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 Working Group  
Internet Draft  
Intended status: Proposed Standard  
Expires: December 2013

P. Sathyanarayan(Ed.)  
S. Hanna  
S. Melam  
Juniper Networks  
Y. Nir  
Check Point  
D. Migault  
Francetelecom - Orange  
K. Pentikousis  
EICT  
October 21, 2013

Auto Discovery VPN Protocol  
draft-sathyanarayan-ipsecme-advpn-03

Abstract

This document defines a protocol for dynamically establishing and tearing down IPsec tunnels as needed without requiring non-scalable configuration.

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), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at  
<http://www.ietf.org/ietf/lid-abstracts.txt>

The list of Internet-Draft Shadow Directories can be accessed at  
<http://www.ietf.org/shadow.html>

This Internet-Draft will expire on April 21, 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.

## Table of Contents

|                                                             |    |
|-------------------------------------------------------------|----|
| 1. Introduction.....                                        | 3  |
| 1.1. Conventions Used In This Document.....                 | 4  |
| 1.2. Terminology.....                                       | 4  |
| 2. Design Considerations.....                               | 5  |
| 3. Auto Discovery VPN Protocol.....                         | 6  |
| 3.1. Prerequisites.....                                     | 7  |
| 3.2. Shortcut Initiation.....                               | 7  |
| 3.3. Shortcut Termination.....                              | 10 |
| 3.4. Peer Address Identification Payload.....               | 10 |
| 3.5. ADVPN_SUPPORTED Notification.....                      | 11 |
| 3.6. ADVPN_INFO Payload.....                                | 12 |
| 3.7. ADVPN_STATUS Notification.....                         | 15 |
| 3.8. The SHORTCUT Exchange.....                             | 17 |
| 3.8.1. Content of the IDi and IDr payloads.....             | 18 |
| 3.9. PROTECTED_DOMAIN Attribute Type.....                   | 19 |
| 3.10. SHORTCUT Response Codes (RCODE).....                  | 20 |
| 3.10.1. SHORTCUT_ACK.....                                   | 20 |
| 3.10.2. SHORTCUT_OK.....                                    | 20 |
| 3.10.3. SHORTCUT_PARTNER_UNREACHABLE.....                   | 21 |
| 3.10.4. TEMPORARILY_DISABLING_SHORTCUT.....                 | 21 |
| 3.10.5. IKEV2_NEGOTIATION_FAILED.....                       | 22 |
| 3.10.6. UNMATCHED_SHORTCUT_SPD.....                         | 22 |
| 3.10.7. UNMATCHED_SHORTCUT_PAD.....                         | 22 |
| 4. Trusted Suggester.....                                   | 23 |
| 4.1. Format of Protected Domain Resource.....               | 24 |
| 4.2. Lifetime of The Data In Protected Domain Resource..... | 24 |
| 5. IPsec Policy.....                                        | 24 |
| 5.1. Security Policy Database (SPD).....                    | 25 |
| 5.1.1. Security Policy Database Cache (SPD Cache).....      | 25 |
| 5.2. Peer Authentication Database (PAD).....                | 26 |
| 6. Security Considerations.....                             | 27 |
| 7. IANA Considerations.....                                 | 27 |
| 8. References.....                                          | 28 |
| 8.1. Normative References.....                              | 28 |
| 8.2. Informative References.....                            | 29 |
| 9. Acknowledgments.....                                     | 29 |

|                                                         |    |
|---------------------------------------------------------|----|
| Appendix A. ADVPN Example Use Cases.....                | 30 |
| A.1. Branch Office Videoconference.....                 | 30 |
| A.2. Optimization for Videoconference with Partner..... | 32 |
| A.3. Heterogeneous Wireless Networks Traffic.....       | 35 |
| Appendix B. Comparison Against ADVPN Requirements.....  | 41 |
| Appendix C. PROTECTED_DOMAIN Example.....               | 48 |

## 1. Introduction

IPsec [IPSECARCH] is currently being deployed in more diverse network environments which exhibit significantly larger numbers of hosts than we have seen before. For example, IPsec is currently used in a broad set of scenarios ranging from remote office VPN deployments to mobile device access to corporate and other sensitive network resources to securing critical telecommunications infrastructure in cellular networks. Large deployments of IPsec may involve thousands of gateways and endpoints with constantly changing traffic patterns. In order to enable efficient and secure traffic flow in such environments, we need to be able to establish tunnels dynamically, as needed. As a result, static IPsec configuration based on presets is no longer deemed adequate. Users expect to be able to connect remotely and securely without compromising their communications quality of experience. In other words, a more dynamic method of establishing and tearing down Security Associations (SAs) [IPSECARCH] than what is currently possible with current standards is desired. This is discussed in [ADVPNreq] where it is shown that, for a variety of use cases, static configuration does not scale for large systems and that a standardized solution is needed where equipment from different vendors may be involved.

Motivated by the problem defined in [ADVPNreq], this document proposes a protocol that can demonstratively scale in large IPsec deployments while ensuring that routing stretch is minimized and network resources are used more optimally. The proposed protocol extends [IKEV2] to meet the requirements spelled out in [ADVPNreq], providing a standard way to dynamically establish and tear down IPsec tunnels, as needed, without requiring non-scalable configuration. The protocol introduces the concept of a "shortcut" which can be used by compliant IPsec gateways to optimize the traffic path between two peers. The protocol has provisions for adhering to established policies and is applicable to single- and multi-domain environments. Shortcuts can be established and torn down dynamically and, as we show in Appendix A, the proposed solution is applicable to a variety of use cases and scenarios, pertaining to both wired and wireless networks.

The remainder of this document is organized as follows. Section 2 presents our design considerations and discusses the salient protocol characteristics we are after. Section 3 specifies the Auto Discovery VPN Protocol (ADVPN), while Section 4 examines the implications of ADVPN on IPsec policy. Security considerations are discussed in Section 5.

Further, this document includes three appendices: Appendix A details several ADVPN use cases, while Appendix B explains how the proposed protocol meets the requirements set in [ADVPNreq]. Finally, Appendix C provides an illustrative example of how the PROTECTED\_DOMAIN response can be created.

### 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 RFC 2119 [MUSTSHOULD].

### 1.2. Terminology

This section defines the terms used throughout this document. The terms introduced in [ADVPNreq] apply here as well. For reader convenience, we repeat in particular the following terms:

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.

In addition, this document defines the following terms:

Peer - A host (gateway or endpoint) with an IPsec Security Association.

Shortcut - An IPsec Security Association established dynamically (at the suggestion of a shortcut suggester).

SHORTCUT exchange - A new IKEv2 exchange that carries all data needed to establish the shortcut IPsec Security Association.

Shortcut suggester - A peer that initiates a SHORTCUT exchange.

Shortcut partners - The pair of peers that received a SHORTCUT exchange suggesting that they should establish a shortcut.

Shortcut initiator - A peer directed by a SHORTCUT exchange to act as the IKEv2 initiator while establishing a shortcut.

Shortcut responder - A peer directed by a SHORTCUT exchange to act as the IKEv2 responder while establishing a shortcut.

## 2. Design Considerations

The protocol described in this document aims at operational environments with possibly tens of thousands (or more) peers. Peers may belong to the same administrative domain, or to different administrative domains which have already established trust relationships. In this kind of network environment one wants to minimize configuration effort overall and, to the degree possible, eliminate manual labor in administering route optimization. This may not only result in better network resource utilization, but also in increased network resilience as reliance on a few centrally-located gateways is reduced. In addition, the automation introduced by the protocol described herein enables administrators to optimize IPsec traffic flows at time scales that are simply not possible with today's tools. In general, the protocol should allow for self-optimization as permitted by established domain policies.

Since IPsec traffic may originate or terminate behind NATs and other policy-enforcing gateways, we aim for a protocol that can work well in this environment. In addition, peers are not expected to be stationary. Given the widespread deployment of wireless networks and the proliferation of mobile devices with multiple interfaces it is reasonable to anticipate that some hosts can join the ADVPN from a range of different access points and this is taken into account in our protocol design.

A central aim of the protocol is to enable peers to setup IPsec tunnels without the need for continuous manual configuration. In addition, the establishment of new tunnels should not inadvertently affect other peers, i.e. it should not call for manual configuration elsewhere in the VPN. Moreover, the establishment of new IPsec tunnels should be easily controlled and managed by the administrator. When new tunnels are operational as well as when they are terminated the administrator should be fully aware of it.

With these considerations in mind, we design a system that can function purely on the basis of local optimizations and policies. In short, the ADVPN protocol enables each individual gateway to act as a shortcut suggester (as per administrator configuration), i.e. to recommend a "shortcut" to appropriate peers with which it has previously established IPsec security associations. These peers, which we refer to as the shortcut partners, can accept, reject or ignore this recommendation, according to their own policies. If the partner(s) reject the recommendation, the partner response indicates the reasons for the rejection, so the shortcut suggester can properly optimize its VPN topology. In addition, responses may also carry informational data that may be handled by the shortcut suggester in various ways. This foundation bestows scaling properties to the Auto Discovery VPN protocol, as described in the following sections.

It is important to highlight that the protocol introduced in this document does not require peers to have a comprehensive understanding of the global network topology. Each peer can act in accordance with its own policy. Taken as a whole, this system optimizes the graph of IPsec Security Associations to match the current traffic flow (subject to policy constraints) and then continuously reoptimizes the IPsec tunnel graph as traffic flows and policies change over time. Appendix A provides illustrative examples of such a (re)optimization.

### 3. Auto Discovery VPN Protocol

The Auto Discovery VPN protocol (ADVPN) enables an IPsec gateway to suggest the establishment of a shortcut, i.e. a direct IPsec tunnel between two of its peers. For example, the shortcut could be used to establish a more optimal path for data delivery.

Whenever an IPsec gateway decides that a shortcut between two of its peers would be beneficial, it initiates a SHORTCUT exchange with both peers, including all information needed to establish the shortcut in the exchange. The peers can reject the SHORTCUT exchange but they can also use the information contained in the exchange to attempt to establish a direct SA between them. We refer to these peers as the shortcut partners. We refer to the gateway offering the shortcut suggestion to both partners as the shortcut suggester.

A shortcut MAY be torn down when it is no longer receiving adequate traffic (as determined by the shortcut partners) or when the timeout for

the shortcut expires. Of course, the shortcut partners MAY decide to explicitly terminate the shortcut at any time.

Note that this protocol works in an exemplary manner in typical hub-and-spoke topologies but it is also well-suited for arbitrary topologies. For example, consider the case of two endpoints exchanging an adequate amount of traffic (as determined by the shortcut suggester) and connected through a series of gateways, all of which support the Auto Discovery VPN protocol. As detailed in Appendix A (Sections A.2. and A.3. , the protocol enables the step-by-step optimization of the traffic flow between two endpoints through the use of shortcut tunnels. The protocol effectively enables direct and secure communication between the two endpoints without any manual configuration involved in setting up the respective tunnels.

### 3.1. Prerequisites

The Auto Discovery VPN protocol MUST only be used with IKEv2.

Before the Auto Discovery VPN protocol can be used, all participants (i.e. the shortcut suggester and the shortcut partners) must indicate support for this protocol by sending ADVPN\_SUPPORTED notification payload as described in Section 3.5. Any IKEv2 peer that sends this notification is indicating that it supports the protocol defined in this draft.

Shortcut partners and shortcut suggesters MUST NOT send any of the messages defined in the remainder of this specification unless the intended recipient of the message has sent the ADVPN\_SUPPORTED notification payload during the IKEv2 exchange. Any party that supports this protocol will send this notification payload in the first IKE\_AUTH request sent in the IKE exchange. However, it may delay sending this payload until later, for example, if it has a policy that restricts the set of peers with which it is willing to establish a shortcut.

### 3.2. Shortcut Initiation

Once the use of the Auto Discovery VPN protocol is enabled, an IPsec gateway which acts as a shortcut suggester can decide that two of its peers (which have indicated support for the ADVPN protocol) should establish a direct IPsec Security Association. Note that the decision-making process for selecting the two peers is outside the scope of this document. As an illustrative example, however, one could consider the observation of excessive transit traffic load between said peers. Another reason could be the realization that certain quality of service (QoS)

requirements would be better served through a shortcut. For instance, some of the traffic between the two peers may be delay-sensitive and would benefit from a more direct route. Alternatively, gateway-, policy- and operation-related reasons, such as overload, scheduled maintenance, energy-saving and so on, could also trigger the initiation of a shortcut recommendation. The reasoning behind the trigger that initiates a shortcut exchange to selected peers is beyond the scope of this document.

Once an IPsec gateway has decided that two peers should establish a direct SA, it acts as a shortcut suggester and uses its already established IKEv2 SAs with these peers to initiate a SHORTCUT exchange to each of the shortcut partners. Each SHORTCUT exchange includes most or all of the information needed to allow the shortcut partners to establish their own SA, such as, the IP address, port number and identity of the other partner, an indication of which partner should be the IKEv2 initiator and which should be the responder, and even an optional Pre-Shared Key, which can facilitate partner authentication with each other.

The shortcut suggester MAY also include Traffic Selectors in the SHORTCUT exchange to indicate which traffic should be sent over the shortcut. This allows traffic for certain destinations to use the ADVPN shortcut while traffic for other destinations continues to flow through the gateway (i.e. the shortcut suggester). Further, it allows traffic destined for certain port numbers (e.g. associated to high-volume, delay-sensitive traffic such as video conferencing applications) to follow the path defined by the shortcut, while other types of traffic carrying, for example, sensitive information that ought to be logged or analyzed, continue to be routed through the gateway.

The shortcut partners MAY decline to act on the SHORTCUT exchange. Although the decision to do so is outside the scope of this document, one could consider, for example, that there may be implementation-specific or operational reasons for rejecting the newly received shortcut suggestion. For instance, the shortcut partners may be low on resources or they may have recently tried to establish this shortcut and failed. Another reason for not accepting the shortcut recommendation could be that doing so would violate local policy. For instance, one of the shortcut partners may accept shortcuts only within its organization.

The shortcut partner(s) MAY ignore the SHORTCUT exchange, but it MUST provide a reason for such refusal to the shortcut suggester by including the ADVPN\_STATUS notification in the SHROTCUT exchange. We note that a shortcut suggester SHOULD NOT reinitiate a SHORTCUT exchange just because the shortcut partners have not set up the requested shortcut tunnel. An ADVPN\_STATUS notification MAY carry a timeout value as an indication by

either of the partners to the shortcut suggester so that the latter defers the re-initiation of a SHORTCUT exchange for this partner pair for the specified amount of time.

The shortcut suggester can indicate, during the SHORTCUT initiation exchange, which shortcut partner should act as the initiator and which as the responder. For example, if only one of the two peers is behind a NAT, the shortcut suggester can indicate the peer behind the NAT as the initiator. Once this decision is made, the shortcut suggester initiates the SHORTCUT exchange with the chosen shortcut responder first. Once the responder's response is received and it indicates acceptance of the suggestion, the shortcut suggester can proceed with the notification to the partner that will act as the shortcut initiator. If, on the other hand, the responder rejects the suggestion, the shortcut suggester MAY change the roles of the partners or terminate the process.

If the shortcut partner identified as the initiator in the SHORTCUT exchange decides to establish the shortcut suggested by the exchange, it will attempt to establish an IKEv2 exchange with its designated shortcut partner (the "shortcut responder") and then to establish an IPsec security association between the two. Ordinarily, the Initiator in an IKE\_AUTH exchange MAY include an IDr payload. In an IKE\_AUTH exchange established because of a SHORTCUT, both the IDi and IDr payloads are mandatory, and their content MUST agree with the ID payloads in the SHORTCUT exchange. Once the SA between the two partners is established, both shortcut partners SHOULD send to the shortcut suggester a SHORTCUT response, indicating that the shortcut tunnel has been established. Details of how this is done are specified in the descriptions of specific Shortcut Types in Section 3.4.

If the shortcut partners are able to establish an IPsec security association, they can use the Traffic Selectors for this SA to determine which traffic should be sent through this tunnel. Shortcut partners MUST ensure that the Traffic Selectors negotiated for the shortcut tunnel are a subset of the Traffic Selectors they have in place for their SA with the shortcut suggester. Since there may be an overlap between the Traffic Selectors for the shortcut SA and for the SA with the shortcut suggester, preference SHOULD be given in this case to sending traffic over the shortcut SA, as described in Section 4.

If a VPN gateway is performing address translation (NAT) for traffic coming from one peer and going to another peer, then it MUST NOT suggest a shortcut between them. Such traffic would have different addresses when flowing directly between the peers, and there is no guarantee that such



flows work with the IPsec policy and with the routing in the remote network.

### 3.3. Shortcut Termination

After establishing an IPsec Security Association triggered by a SHORTCUT exchange, as described in the following subsection, either of the shortcut partners may decide to terminate the shortcut. This may occur at any point of time and for a variety of reasons (outside the scope of this document), such as, for example, due to lack of traffic using the shortcut, local policy, shortage of resources, or other reasons. However, the shortcut SA **SHOULD NOT** be terminated simply because the SA with the shortcut suggester was terminated due to inactivity. On the contrary, dropping the SA with the shortcut suggester while maintaining the shortcut SA may be quite a normal occurrence if the only traffic flowing through the shortcut suggester has now been diverted into the shortcut.

Note that either shortcut partner may terminate a shortcut by closing the corresponding IKE SA (and therefore all child IPsec SAs) by sending an IKEv2 Delete payload to the other shortcut partner, thus indicating that the IKE SA should be deleted.

### 3.4. Peer Address Identification Payload

The Peer Address Identification Payload, denoted as IDa, contains the address of the peer gateway. It is formatted in the same manner as the IDi and IDr payloads which are defined in Section 3.5 of RFC 5996. The ID type in the IDa payload **MUST** be from one of the following types:

- . ID\_IPV4\_ADDR, indicating an IPv4 address
- . ID\_IPV6\_ADDR, indicating an IPv6 address
- . ID\_FQDN, indicating a fully qualified domain name; this is allowed if and only if the peer has indicated the capability "FQDN Resolver" in its ADVPN\_SUPPORTED notification. See Section 3.5.

This payload is currently defined only for the SHORTCUT exchange. It **MUST NOT** be sent to any peer that has not indicated support for SHORTCUT exchanges by sending the ADVPN\_SUPPORTED notification. The Critical bit **MUST** be set.

[RFC EDITOR PLEASE REMOVE THIS PARAGRAPH] For development and interoperability testing while this document is still a draft and IANA actions have not taken place, implementations can use the private-use value of 247 for the payload type of the IDa payload.

### 3.5. ADVPN\_SUPPORTED Notification

The Notify payload for announcing support of ADVPN is included in the Initial Exchange and 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 ! ADVPN Supported Message Type !
+-----+-----+-----+-----+-----+-----+-----+-----+
! Capabilities ... !
+-----+-----+-----+-----+-----+-----+-----+-----+

```

The fields corresponding to the first 4 octets are defined as described in RFC 5996. The remaining fields are defined as follows:

- . Protocol ID (1 octet) MUST be zero, as specified in Section 3.10 of RFC 5996.
- . SPI Size (1 octet) MUST be zero, in conformance with Section 3.10 of RFC 5996.
- . ADVPN Supported Message Type (2 octets) - MUST be xxxxxx. [RFC EDITOR NOTE: value assigned by IANA for ADVPN\_SUPPORTED]
- . The Capabilities field can be two or more octets long, indicating the capabilities that this implementation supports. See the Table below for the capabilities specified in this document. The first of these capabilities MUST indicate the protocol version range (0x01-0x08), and at least one MUST list the features range (0x09-0xff). All version capabilities MUST precede all feature capabilities.

| Value | Name | Comment                                                                                                                               |
|-------|------|---------------------------------------------------------------------------------------------------------------------------------------|
| 0x00  | pad  | Used to pad the notification to any desired length. MAY be sent multiple times at the end of the list, and MUST be ignored on receipt |

|            |                   |                                                                                                             |
|------------|-------------------|-------------------------------------------------------------------------------------------------------------|
| 0x01       | v1                | The version described in this document. MUST be sent first by implementations compliant with this document. |
| 0x02..0x08 | RES1              | Reserved for future versions                                                                                |
| 0x09       | Suggester         | Can act as shortcut suggester in a SHORTCUT exchange                                                        |
| 0x0A       | Shortcut Partner  | Can act as shortcut partner in a SHORTCUT                                                                   |
| 0x0B       | FQDN Resolver     | Can resolve peer locators given as FQDN. Relevant only for the shortcut partner.                            |
| 0x0C       | Trusted Suggester | This peer can act as a trusted suggester as described in Section 4.                                         |
| 0x0D..0x7F | RES2              | Reserved for future extensions                                                                              |
| 0x80..0xDF | RES3              | Reserved                                                                                                    |
| 0xE0..0xFF | RES4              | Reserved for private use                                                                                    |

The IKE exchange Initiator MAY send multiple version capabilities. The Responder MUST send exactly one version capability, and that capability represents the version of the specification to be used.

A receiver MUST ignore any capabilities it does not recognize. Extension documents SHOULD consider the effects of the peer not recognizing such capabilities. If such extensions are critical for the operation of the protocol, a new version number may also be needed.

[RFC EDITOR PLEASE REMOVE THIS PARAGRAPH] For development and interoperability testing while this document is still a draft and IANA actions have not taken place, implementations can use the private-use value of 47831 as the ADVPN\_SUPPORTED Notify type.

### 3.6. ADVPN\_INFO Payload

The ADVPN\_INFO payload is used in the SHORTCUT exchange to send information from the suggester to a shortcut partner about the shortcut.

```

 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 !
+-----+-----+-----+-----+-----+-----+-----+-----+
! SHORTCUT Identifier !
+-----+-----+-----+-----+-----+-----+-----+-----+
! Lifetime !
+-----+-----+-----+-----+-----+-----+-----+-----+
! R | Reserved | PSK Length | Peer Port !
+-----+-----+-----+-----+-----+-----+-----+-----+
| | |
| Pre-Shared Key (variable length) |
| | |
+-----+-----+-----+-----+-----+-----+-----+-----+
| | |
| Peer Description (variable length) |
| | |
+-----+-----+-----+-----+-----+-----+-----+-----+

```

The fields corresponding to the first 4 octets are defined as described in RFC 5996. The remaining fields are defined as follows:

- . SHORTCUT Identifier (4 octets) - a 32-bit identifier for this shortcut. The same value MUST be used for both shortcut peers and it MUST be unique per suggester and partners pair. The value SHOULD be unique per suggester, i.e. there should not exist in the VPN two SHORTCUTs initiated by the same suggester with the same identifier.
- . Lifetime (4 octets) - a 32-bit integer that indicates the maximum number of seconds that this shortcut recommendation should last. After this period of time lapses, the shortcut partners SHOULD tear down the shortcut SA. If the field is 0, the shortcut suggestion MAY last indefinitely. The shortcut partners MAY use a smaller timeout value than given here based on their policies.
- . R - Role (2 bits) - this field indicates to the partner its designated role in the upcoming exchange (i.e. shortcut initiator or responder). Role assignment is decided by the shortcut suggester and, as mentioned earlier, it is outside the scope of this document. Value 00 is reserved and MUST NOT be used by implementations compliant to this specification. Value 01 indicates that the receiving peer MUST act as shortcut responder. Value 10 indicates that the peer receiving this payload MUST act as the shortcut initiator. Value 11 indicates that the receiving peer SHOULD NOT initiate the exchange immediately and MAY initiate the exchange at later stage. The waiting time before the peer initiates the exchange could be several minutes.

- . PSK Length (1 octet) indicates the length in octets of the Pre-Shared Key. It MUST be set to zero if certificate authentication is to be used between the shortcut peers.
- . Peer Port (16 bit) - is set to zero when none of the shortcut partners are behind a NAT. The suggester has IKEv2 and IPsec channels with both shortcut partners, so it is aware whether partners are behind a NAT or not. If one of the shortcut partners is behind a NAT, the Peer Port MUST be a non-zero value. This value is used for UDP encapsulation as defined in [RFC3948] between the partner that has received the shortcut payload and the peer shortcut partner.

If the peer shortcut partner is behind a NAT, the Peer Port designates the port by which the NAT identifies the peer within its private network. The global IP address of the NAT is provided by the Peer Address Identification Payload (IDa). When the shortcut partner receiving the shortcut payload sends an IKEv2 or IPsec ESP packet to the peer shortcut partner, it MUST UDP encapsulate the packet with IP source set to its local IP address, source port set to 4500, IP destination set to the IP provided by the Peer Address Identification Payload, and destination port set to Peer Port. When the packet reaches the NAT gateway, the IP destination and destination port are translated by the NAT to private IP address and 4500. Similarly, IKEv2 and IPsec ESP packets sent by the peer shortcut partner are UDP encapsulated with IP source set to the private IP address of the peer shortcut partner, source port set to 4500. IDa determines the destination payload sent to the peer shortcut partner and the counterpart shortcut payload Peer Port field sent to the peer shortcut partner defines the destination port. This field MAY be 4500 or another value depending on whether the shortcut partner of the shortcut payload (described in this section) is also behind a NAT.

If the peer shortcut partner is not behind a NAT, but the partner receiving this shortcut payload is behind a NAT, then the Peer Port value is set to 4500. When the shortcut partner sends an IKEv2 or an IPsec ESP packet, it MUST UDP encapsulate the packet with IP source set to its private IP, the port source set to 4500, IP destination set to the IP provided by the Peer Address Identification Payload (IDa) and the destination port 4500. When the packet reaches the NAT gateway, IP source and port are translated by the NAT with the global IP address and the port used to identify the shortcut partner. These two values are provided to the peer shortcut partner, by the Peer Address Identification Payload (IDa) and shortcut payload sent by the suggester to the peer shortcut partner.

If the Peer Port is set to zero and the partner is behind a NAT, this is obviously a misconfiguration. The partner MAY return a RCODE set to SHORTCUT\_PARTNER\_UNREACHABLE. If the partner initiates the IKEv2 negotiation, it MUST ignore the Peer Port value and proceed to UDP encapsulation. The negotiation MAY be successful only if the peer shortcut partner is not behind a NAT. Similarly if the partner is not initiating the IKEv2 negotiation, the negotiation MAY be successful if the shortcut partner has received a properly configured shortcut payload.

- . Pre-Shared Key - An octet string used as the PSK in the IKE\_AUTH exchange between the shortcut partners. This field MUST be the same in the ADVPN INFO payloads sent to the two shortcut partners. It MUST be randomly generated using a good random source, and it SHOULD be long enough to meet the security requirements of the deployment. In practice, this means that the length of the randomly generated data should be at least 16 octets long.
- . Peer Description (variable length) - The length of this field is calculated by subtracting the combined lengths of the other fields from the value of the Payload Length field. It contains a description of the other peer, in a format that is simply a name, suitable for logging, and encoded in UTF-8.

[RFC EDITOR PLEASE REMOVE THIS PARAGRAPH] For development and interoperability testing while this document is still a draft and IANA actions have not taken place, implementations can use the private-use value of 248 as the ADVPN\_INFO payload type.

### 3.7. ADVPN\_STATUS Notification

The ADVPN\_STATUS payload is used by a shortcut partner for conveying to the suggester and to the other SHORTCUT peer the status of the shortcut.

```

 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 ! ADVPN Status Message Type !
+-----+-----+-----+-----+-----+-----+-----+-----+
! SHORTCUT Identifier !
+-----+-----+-----+-----+-----+-----+-----+-----+

```

```

!F|C|E| Reserved | RCODE |
+-----+-----+-----+-----+-----+-----+
! Timeout !
+-----+-----+-----+-----+-----+-----+

```

The fields corresponding to the first 4 octets are defined as described in RFC 5996. The remaining fields are defined as follows:

- . Protocol ID (1 octet) MUST be zero, as specified in Section 3.10 of RFC 5996.
- . SPI Size (1 octet) MUST be zero, in conformance with Section 3.10 of RFC 5996.
- . ADVPN Status Message Type (2 octets) - MUST be yyyyyy. [RFC EDITOR NOTE: value assigned by IANA for ADVPN\_STATUS]
- . SHORTCUT Identifier - the 32-bit field from the SHORTCUT\_INFO notification.
- . F (1 bit) - Fatal. Set if this notification means that the SHORTCUT no longer exists.
- . C (1 bit) - Critical. Set if the peer must understand this RCODE, or else delete the shortcut if the F bit is not set.
- . E (1 bit) - Error. Indicates an error condition (rather than a policy change).
- . RCODE (2 octets) - a 16-bit field as described in Section 3.8.1.
- . Timeout - this 32-bit field indicates the maximum number of seconds that the shortcut service is not available by the peer.

The SHORTCUT\_STATUS notification may be sent to the suggester in the SHORTCUT exchange response message. If either the suggester or a partner needs to communicate a change in status after the original SHORTCUT exchange is over, the same can be communicated in an INFORMATIONAL request. Additionally, the same notification is used in the IKE exchange between the shortcut partners, so that they can match that exchange and the resulting Security Associations to the shortcut. Since there may be more than one active shortcut between a pair of shortcut partners, the notification is inserted into the exchanges that create child SAs, either the IKE\_AUTH exchange or the CREATE\_CHILD\_SA exchange.

[RFC EDITOR PLEASE REMOVE THIS PARAGRAPH] For development and interoperability testing while this document is still a draft and IANA actions have not taken place, implementations can use the private-use value of 47833 as the ADVPN\_STATUS notify type.

### 3.8. The SHORTCUT Exchange

This new exchange type defines how the suggestions are conveyed. It is designed for sending the SHORTCUT data. The suggester initiates the SHORTCUT exchange and each of the shortcut partners responds to the notification. The shortcut partner acts as the [RFC5996] Responder. The exchange is constructed as follows:

```
HDR, SK {IDa, ADVPN_INFO, IDi,
 IDr[, TSi][, TSr][, VID]} -->
 <== HDR, SK {N(ADVPN_STATUS)}
```

The IDa payload, defined in Section 3.4. provides the location of the other shortcut peer and it may not necessarily have the same data as the IDi or IDr payloads.

The ADVPN\_INFO payload contains PSK and any other information needed for establishing the SHORTCUT IKE SA, as well as the optional time out information.

The IDi Identification Payload contains the identity of the shortcut initiator. The shortcut initiator MUST use this identifier when establishing the shortcut and the shortcut responder MUST verify that this identifier was used.

The IDr Identification Payload contains the identity of the shortcut responder. The shortcut initiator MUST use this payload in the subsequent IKE\_AUTH exchange with the shortcut responder.

The TSi and TSr Traffic Selector Payloads (when present) contain, respectively, traffic selectors the intended Initiator and intended Responder in the IKE\_AUTH exchange between the shortcut partners. The content is as specified in Section 3.13 of RFC 5996 [IKEV2].

The responder checks that everything in the request is acceptable according to local policy. If not, it MUST return an error RCODE. If the shortcut is acceptable, then it either tries to establish the shortcut IKE SA, and reports the results in the SHORTCUT response, or it immediately responds with a SHORTCUT\_ACK RCODE, and follows up with more detailed status reports in future Informational exchanges.

In the subsequent IKE\_AUTH exchange between the two partners, the initiator MUST use the IDi and IDr payloads as specified in the exchange described in this section, and MUST use a subset (not necessarily a proper



subset) of the traffic selector payload data, if such data has been included in the SHORTCUT exchange.

When shortcut are performed within a single administrative domain, IKE PAD are likely to be configured to trust all partners associated to a given domain. This will most probably be reflected by a common field in the certificate, and the ID is determined by the certificate. On the other end, PSK and a random ID MUST be added in the PAD of the partners. Creation of a new entry in the PAD is done since the suggester is trusted. If not the partner MUST return a SHORTCUT Response with UNMATCHED SHORTCUT PAD.

### 3.8.1. Content of the IDi and IDr payloads

The identification payloads in the SHORTCUT request message are later used in the IKE\_AUTH exchange between the shortcut partners. They are required to follow the rules in RFC 5996 for authentication of the IKE SA. However, those rules are vague and left to specific profiles, specific implementations and specific administrative decisions.

Since AD-VPN is intended to work with multiple implementations and multiple administrative domains, we believe it is necessary to specify the content of these payloads more strictly.

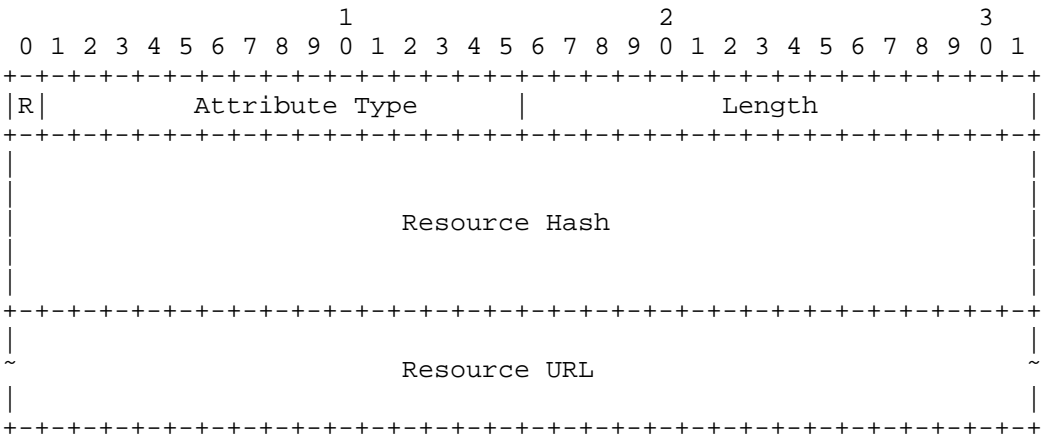
If a shortcut partner is supposed to authenticate using a certificate (and, therefore, the PSK Length field in the ADVPN\_INFO payload is set to zero), then the ID payload matching this partner MUST match the certificate that it has. In this case, the ID payload MUST be from one of the following types:

- . ID\_IPV4\_ADDR or ID\_IPV6\_ADDR. This ID type MAY be used only if the certificate contains an alternate name extension of type ipAddress, and MUST contain the same IP address as the extension.
- . ID\_FQDN. This ID type MAY be used only if the certificate contains an alternate name extension of type dNSName, and MUST contain the same FQDN as the extension.
- . ID\_RFC822\_ADDR. This ID type MAY be used only if the certificate contains an alternate name extension of type rfc822Name, and MUST contain the same NAI as the extension.
- . ID\_DER\_ASN1\_DN. This ID type can always be used, and if used, MUST contain an exact copy of the subject field of the certificate.

If the partners are using a PSK to authenticate, the ID payloads need not reflect any real name of the partners, as that information is conveyed in the ADVPN\_INFO payload. The ID payloads do, however, need to be unique so as to allow a quick lookup of the ID payload. To ensure this, the ID payload MUST be of type ID\_KEY\_ID, and the content MUST be unique. To ensure uniqueness across administrative domains, the content of the ID payload in such cases MUST be randomly or pseudo-randomly generated and MUST have 128 bits.

3.9. PROTECTED\_DOMAIN Attribute Type

This is a new attribute for the CFG payload. In a request, its size MUST be a constant zero. In a response, it MUST be at least 20 octets long, having the following format:



PROTECTED\_DOMAIN Configuration Attribute Format

In the above diagram, Attribute Type is ZZZ, [RFC EDITOR NOTE: value assigned by IANA for PROTECTED\_DOMAIN], Resource Hash is the SHA-1 hash of the resource, and Resource URL is an HTTP URL pointing at the resource.

The corresponding entry for this attribute as in the table in section 3.15.1 of RFC 5996 is as follows:

| Attribute Type | Value | Multi-Valued | Length |
|----------------|-------|--------------|--------|
| -----          |       |              |        |

PROTECTED\_DOMAIN            1            NO            0, or 20+ octets

The format of the resource itself is described in Section 4.1.

### 3.10. SHORTCUT Response Codes (RCODE)

This section provides more information on the use of the response codes (RCODE). RCODE is a 16-bit field, used by the shortcut partners to indicate the status on the SHORTCUT notification received.

The RCODEs we consider in this document are the following:

| Value | Description                    |
|-------|--------------------------------|
| ----- |                                |
| 0     | SHORTCUT_ACK                   |
| 1     | SHORTCUT_OK                    |
| 2     | SHORTCUT_PARTNER_UNREACHABLE   |
| 3     | TEMPORARILY_DISABLING_SHORTCUT |
| 4     | SHORTCUT_PARTNER_UNREACHABLE   |
| 5     | IKEv2_NEGOTIATION_FAILED       |
| 6     | UNMATCHED_SHORTCUT_SPD         |
| 7     | UNMATCHED_SHORTCUT_PAD         |

#### 3.10.1. SHORTCUT\_ACK

The RCODE value for SHORTCUT\_ACK is: 0 (zero)

This RCODE indicates that the receiving partner has accepted the shortcut, but has yet to establish the shortcut IKE SA. This RCODE MUST be used only in the response for a SHORTCUT exchange.

#### 3.10.2. SHORTCUT\_OK

The RCODE value for SHORTCUT\_OK is: 1

This RCODE indicates that the shortcut has been successfully established between the shortcut partners.

### 3.10.3. SHORTCUT\_PARTNER\_UNREACHABLE

The RCODE value for SHORTCUT\_PARTNER\_UNREACHABLE is: 2

This RCODE indicates that the attempt to establish the recommended shortcut has failed because the partner peer was unreachable. This may happen, for example, if the partner peers are behind separate NATs, or a firewall drops packets between the shortcut partners. It may also be that the partner peer is only available through a specific interface. In addition, the partner peer may have been temporarily disconnected or its shortcut service has been temporarily disabled as explained in subsection 3.10.4.

It is the responsibility of the shortcut suggester to determine the reason of the observed unreachability as well as what policy to apply. However, the shortcut suggester SHOULD NOT initiate another SHORTCUT exchange to the shortcut partners before the Timeout indicated in the shortcut Data. If the Timeout value is not present, then it is up to the shortcut suggester to decide when a new SHORTCUT exchange should be initiated.

### 3.10.4. TEMPORARILY\_DISABLING\_SHORTCUT

The RCODE for TEMPORARILY\_DISABLING\_SHORTCUT is: 3

This RCODE indicates that the shortcut recommendation is refused by the shortcut peer because it has deactivated the shortcut service. In other words, this RCODE indicates that any attempt to establish shortcuts will be refused independently of the SHORTCUT exchange sent. For example, the shortcut service could be disabled when the shortcut peer is overloaded.

If the shortcut initiator generates this response code, then it SHOULD NOT initiate the shortcut negotiation.

When receiving an ADVPN\_STATUS notification with this response code, the shortcut suggester SHOULD NOT initiate any other SHORTCUT exchange before the Timeout indicated in the shortcut Data. If the Timeout value is not present, then it is up to the shortcut suggester to decide when a new SHORTCUT exchange should be initiated.

### 3.10.5. IKEV2\_NEGOTIATION\_FAILED

The Shortcut Type for IKEV2\_NEGOTIATION\_FAILED is: 4

This RCODE indicates that the IKEv2 negotiation between the two partner peers did not complete successfully. That is, the shortcut recommendation was accepted and acted upon, but the IKEv2 negotiation failed. This RCODE does not provide information on the reasons the shortcut establishment failed, and thus other more specific RCODEs (see below) SHOULD be preferred by implementations when this is possible.

### 3.10.6. UNMATCHED\_SHORTCUT\_SPD

The RCODE for UNMATCHED\_SHORTCUT\_SPD is: 5

This RCODE indicates an error resulting from the analysis of the SHORTCUT exchange. Before establishing a shortcut, the shortcut initiator MUST check that the shortcut partner's IP address matches its Security Policy Database (SPD). If a mismatch occurs with the shortcut initiator's SPD, the shortcut initiator MUST NOT initiate the shortcut. In this case, the initiator MUST use the UNMATCHED\_SHORTCUT\_SPD RCODE in its ADVPN\_STATUS notification.

If the mismatch occurs with the shortcut responder, it MUST send to the shortcut suggester the UNMATCHED\_SHORTCUT\_SPD RCODE in its ADVPN\_STATUS notification. Eventually the shortcut initiator will start an IKEv2 negotiation. The shortcut responder SHOULD terminate the IKEV2 negotiation with a TS\_UNACCEPTABLE. At this stage the shortcut cannot be established and the shortcut initiator MUST respond to the shortcut suggester with the IKEV2\_NEGOTIATION\_FAILED Shortcut Type in its ADVPN\_STATUS Notify Payload.

When receiving ADVPN\_STATUS with this RCODE, the shortcut suggester SHOULD NOT reinitiate a SHORTCUT exchange with the shortcut partners with the same Traffic Selectors in short order.

### 3.10.7. UNMATCHED\_SHORTCUT\_PAD

The Shortcut Type for UNMATCHED\_SHORTCUT\_PAD is: 6

This RCODE indicates an error resulting from the analysis of the SHORTCUT exchange. Before establishing a shortcut, the shortcut initiator MUST

check that the shortcut partner's IP address and Identities IDi/IDr match its Peer Authentication Database (PAD). If a mismatch occurs with the shortcut initiator's PAD, the shortcut initiator MUST NOT initiate the establishment of the recommended shortcut. The initiator then sends the UNMATCHED\_SHORTCUT\_PAD RCODE in its ADVPN\_STATUS notification

If the mismatch occurs with the shortcut responder, it MUST send to the shortcut suggester the UNMATCHED\_SHORTCUT\_PAD RCODE in its ADVPN\_STATUS notification. Eventually the shortcut initiator will start an IKEv2 negotiation. The shortcut responder SHOULD terminate the IKEv2 negotiation with a TS\_UNACCEPTABLE. Thus, the shortcut cannot be established and the shortcut initiator MUST return the shortcut suggester the IKEv2\_NEGOTIATION\_FAILED Shortcut Type in its ADVPN\_STATUS.

When receiving ADVPN\_STATUS with this RCODE, the shortcut suggester SHOULD NOT reinitiate a SHORTCUT exchange with the shortcut partners with the same Traffic Selectors in short order.

#### 4. Trusted Suggester

Advertising this capability means that the sender supports sending its entire protected domain through the PROTECTED\_DOMAIN attribute in the CFG payload.

The intended way to use this is that a spoke node, whether a remote access client or a gateway, is configured only with credentials for a single hub gateway. On the first Initial exchange, the hub gateway advertises the TRUSTED\_SUGGESTER capability. In the IKE\_AUTH exchange or in a later Informational exchange, the spoke sends a CFG\_REQUEST, asking for the PROTECTED\_DOMAIN. The reply (see Section 3.8.1. gives the spoke a URL for a resource that contains the entire protected domain of the hub, which is the entire protected domain of the VPN. The CFG\_REPLY also contains a hash of the resource, which has an important security benefit. Since there are attacks against HTTP, and no obvious way to secure HTTPS in this context, using a protected exchange to send a hash of the resource makes sure that the retrieved resource is in fact the intended one.

The list of IP addresses and ranges returned in the resource is used to populate the SPD, and provides an initial configuration with all VPN traffic going through the hub gateway. Normal ADVPN protocol operation can later optimize the traffic, but this mechanism ensures that bootstrapping does not require extensive manual configuration.

The procedure described above begins with the bare minimum of configuration. Some nodes will begin with some initial configuration, in

which case it is up to them to harmonize their static configuration with the data from the trusted suggester. It is perfectly valid to use only a subset of the protected domain in populating the SPD. Appendix C provides an illustrative example of the use of the PROTECTED\_DOMAIN attribute.

#### 4.1. Format of Protected Domain Resource

The protected domain resource is formatted much like a CFG reply. It is a series of Configuration Attributes (see section 3.15.1 in RFC 5996), but the only attribute types allowed are INTERNAL\_IP4\_SUBNET (13) and INTERNAL\_IP6\_SUBNET (15). The last Configuration Attribute has the R bit set to mark it as the last one.

For an example, the following resource indicates a protected domain comprised of all three test networks from [RFC5737] and [RFC3849]:

```
0x0D 0x08 0xC0 0x00 0x02 0x00 0xff 0xff 0xff 0x00
- 8-octet IP4_SUBNET: 192.0.2.0 (255.255.255.0)
0x0D 0x08 0xC6 0x33 0x64 0x00 0xff 0xff 0xff 0x00
- 8-octet IP4_SUBNET: 198.51.100.0 (255.255.255.0)
0x0D 0x08 0xCB 0x00 0xD1 0x00 0xff 0xff 0xff 0x00
- 8-octet IP4_SUBNET: 203.0.113.0 (255.255.255.0)
0x8F 0x11 0x20 0x01 0x0D 0xB8 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00
0x00 0x00 0x00 0x00 0x20
- Last 17-octet IP6_SUBNET: 2001:DB8::/32
```

#### 4.2. Lifetime of The Data In Protected Domain Resource

As the Protected Domain Resource is delivered over HTTP/1.1 or above, the Cache-Control ([HTTPCache]) directives can be used to assign a lifetime. The IPsec node MUST expire the entries from the SPD as soon as the data would expire from the cache, or MUST fetch fresh data beforehand.

### 5. IPsec Policy

This section discusses the implications of the use of the ADVPN Protocol on IPsec policy.

### 5.1. Security Policy Database (SPD)

The SHORTCUT exchange described in Section 3.2. conveys policy in the sense of Section 4.4.1 of RFC 4301 ([IPSECARCH]). In the terms of that document, these are SPD elements. Assuming these elements are accepted, they update the existing security policy of the receiver.

The entries specified in a SHORTCUT exchange are inserted into the SPD immediately before the entry that they are updating, so that these new entries take precedence over existing ones.

RFC 4301 does not specify time limits for SPD entries. In that sense, this document updates RFC 4301. SPD entries now come in two flavors: static entries, which have no expiration time and are defined by an administrator, and dynamic entries which have an expiration time as specified in the SHORTCUT exchange.

For example, suppose a static entry exists for the remote subnet 192.0.2.0/23, and the local subnet is 192.0.1.0/24. Initially, the entry looks like this:

```
Local=192.0.1.0/24,Remote=192.0.2.0/23,PROT,Peer=vpngw.example.com
```

Assume now that a SHORTCUT exchange is received which describes gateway foo.example.com, and remote subnet 192.0.2.0/24. The database will look as follows:

```
Local=192.0.1.0/24,Remote=192.0.2.0/24,PROT,Peer=foo.example.com
```

```
Local=192.0.1.0/24,Remote=192.0.2.0/23,PROT,Peer=vpngw.example.com
```

Because of the rules of processing as specified in Section 5.1 of RFC 4301, the earlier entry takes precedence, and overrides the second entry for subnet 192.0.2.0/24. The second entry still applies to 192.0.3.0/24.

#### 5.1.1. Security Policy Database Cache (SPD Cache)

The SPD Cache also needs to be updated. With the above entry, a cache entry was created reflecting the matching SA. If no change to the cache is



made, the IPsec stack will continue to use the existing SA despite the change in policy. Since implementations of the SPD cache vary widely, we do not specify the exact way to handle this change, but discuss below some implementation suggestions.

One way to handle this would be to narrow the existing SPD Cache entry so as to cover only the selectors which are not affected by the SHORTCUT. This has the advantage of forcing the SPD cache entry to a failure match with the negotiated SA. Whether this is a problem depends on the implementation of these databases. It is likely not a good idea to also narrow the existing SAs. While it should be fine for outbound SAs, it will cause the IPsec stack to drop validly encrypted packets on inbound processing.

Another way to handle this would be to simply delete the SPD cache entry, forcing a re-evaluation of the SPD for the next packets. This causes an even more serious discrepancy between the Security Association Database (SAD) and SPD Cache. This should only be done if it is possible to match existing SAs to new SPD cache entries, which, again, depends on the implementation details.

The one foolproof way is to erase both SPD cache entries and SAs, sending the appropriate DELETE payloads to the peer. This is perfectly compliant and perfectly functional, but will create more work for the IKE daemon.

## 5.2. Peer Authentication Database (PAD)

This database will also be updated with a temporary entry when a SHORTCUT exchange is received. The entry includes the name, IP address and a specification of either PSK or certificate authentication. This entry MUST also expire when the SHORTCUT expires.

It is conceivable that peers will appear in both static and dynamic entries. It is also possible that the same peer will be mentioned in multiple SHORTCUT exchanges, each with a different expiration time. An implementation of this specification MUST track all such entries. Two entries will be considered to represent the same entity if either they share both ID and certificate, or if they share ID and IP address.

If all entries matching a particular entity expire, then the implementation MUST delete all IKE and child SAs associated with that entity.

## 6. Security Considerations

No lifetime is specified for the Pre-Shared Key (PSK) so the shortcut suggester SHOULD generate the PSK value with plenty of entropy. See [RANDOMNESS] for advice on generating random numbers for cryptographic purposes. The shortcut partners may rekey as needed and may even use the PSK value for reauthentication, although it is not clear that there is much value in doing so. If one of the shortcut partners decides that the PSK is too old (recognizing that it is only used for authentication), it may simply tear down the shortcut SA. Eventually, the shortcut suggester will set up the shortcut again, if it is needed.

To head off this situation, the shortcut suggester may periodically initiate a new SHORTCUT exchange to each of the two shortcut partners. If a shortcut partner receives a SHORTCUT exchange suggesting a shortcut that already exists with new parameters, the shortcut partner SHOULD establish a new shortcut SA with the peer partner using the new parameters and then tear down the old shortcut SA.

## 7. IANA Considerations

IANA is requested to allocate a new payload type from the "IKEv2 Payload Types" registry with the name "Identification - Peer Address", i.e. "IDa", and this document as the corresponding reference document.

IANA is requested to allocate a new Configuration Payload Attribute Type with name "PROTECTED\_DOMAIN", not multivalued, with a length of "0 or more octets".

IANA is requested to allocate a new payload type from the "IKEv2 Payload Types" registry with the name "ADVPN Information", i.e. "ADVPN\_INFO", and this document as the corresponding reference document.

IANA is requested to allocate a new exchange type from the "IKEv2 Exchange Types" registry with name "SHORTCUT" and this document as reference.

IANA is requested to assign two code points from the "IKEv2 Notify Message Types - Status Types" registry, as follows. The reference document for all three shall be this document:

. SHORTCUT\_SUPPORTED

## . SHORTCUT\_STATUS

IANA is requested to set up a new registry called "IKEv2 ADVPN Capabilities". The value is 8-bit, and the range is partitioned as follows:

- . 0x00 reserved for padding
- . 0x01-0x08 used for version indication and negotiation
- . 0x09-0x07F used for supported features
- . 0x80-0xDF reserved
- . 0xE0-0xFF reserved for private use

The policy for allocations from the version range shall be "Standards Action". The policy for allocation from the features range shall be "Expert Review". The initial allocation for this registry is as defined in the table in Section 3.5. with the additional column of reference specification, which shall be set to this document.

IANA is requested to set up a new registry called "IKEv2 AD-VPN Response Codes". The value is 16-bit, and the range is partitioned as follows:

- . 0x0000-0xBFFF Used for RCODEs
- . 0xC000-0xFFFF Reserved for private use

The policy for allocations from this registry shall be FCFS. The initial allocation is given in the table in Section 3.8.1.

## 8. References

### 8.1. Normative References

- [IKEV2] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.
- [IPSECARCH] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.
- [MUSTSHOULD] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC1034] Mockapetris, P., "Domain Names - Concepts and Facilities", RFC 1034, November 1987.

- [RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, August 2010.
- [HTTPCache] Fielding, R., Nottingham, M., and Reschke, J., "Hypertext Transfer Protocol (HTTP/1.1): Caching", draft-ietf-httpbis-p6-cache-23 (work in progress), July 2013.

## 8.2. Informative References

- [ADVPNreq] Manral, V., "Auto Discovery VPN Problem Statement and Requirements", RFC 7018.
- [RANDOMNESS] Eastlake, 3rd, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", BCP 106, RFC 4086, June 2005.
- [RFC3492] Costello, A., "Punycode: A Bootstring encoding of Unicode for Internationalized Domain Names in Applications (IDNA)", RFC 3492, March 2003.
- [MEDIATION] Brunner, T., "IKEv2 Mediation Extension draft-burner-ikev2-mediation-00"
- [RFC5737] Arkko, J., Cotton, M., and Vegoda, L., "IPv4 Address Blocks Reserved for Documentation", RFC 5737, January 2010.
- [RFC3849] Huston, G., Lord, A., and Smith, P., "IPv6 Address Prefix Reserved for Documentation", RFC 3849, July, 2004.

## 9. Acknowledgments

This document was prepared using 2-Word-v2.0.template.dot.

The authors of this draft would like to acknowledge the following people who have contributed to or provided substantial input on the preparation of this document or predecessors to it: Scott McKinnon, Vishwas Manral, Valery Smyslov, Michael Richardson and Yaron Sheffer.

## Appendix A. ADVPN Example Use Cases

This appendix presents a few example situations where the ADVPN protocol may be useful and illustrates how it works.

## A.1. Branch Office Videoconference

In this example, users have initiated a videoconference between two branch offices of SmithCo located in Ashby and Bedford. Each branch office has an IPsec gateway that is configured to send all traffic to the IPsec gateway at the main SmithCo office in Paris. Figure 6 illustrates this initial situation, showing these three IPsec gateways and the IPsec SAs in place when the videoconference starts.

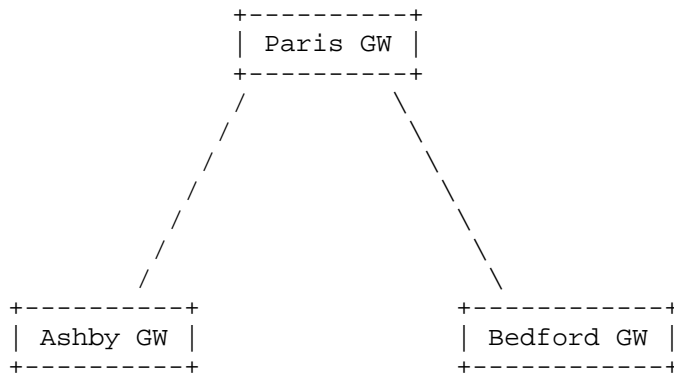


Figure 6: Initial SmithCo IPsec SAs

All of these gateways support SHORTCUT exchanges and have been configured to use them within SmithCo. Therefore, they all sent the ADVPN\_SUPPORTED notification payload described in Section 3.4 to each other in their initial IKE exchanges. This means that they are all aware that SHORTCUT exchange may be used on the IPsec SAs illustrated in Figure 6.

Once the videoconference begins, the Paris GW notices a large amount of videoconference traffic between the Ashby GW and the Bedford GW. The Paris GW has been configured to permit videoconference traffic to trigger a shortcut between two branch gateways so it initiates a SHORTCUT exchange to the Ashby and Bedford GWs, suggesting that they establish a shortcut. In this instance, it identifies the Ashby GW as the shortcut initiator by

setting the I bit in the exchange sent to that gateway and leaving that bit cleared in the exchange sent to Bedford GW.

Because all gateways in SmithCo have certificates from the same CA and have been configured to trust that CA to issue certificates, there is no need to use a PSK. The Paris GW simply sets the IDi Identification Payload field of the SHORTCUT exchange to the subject DN of the Ashby GW and the IDr Identification Payload field of the SHORTCUT exchange to the subject DN of the Bedford GW.

The Paris GW sets the TSi Traffic Selector Payload and TSr Traffic Selector Payload fields in the SHORTCUT exchange to indicate that the Ashby GW should only use this shortcut for videoconferencing traffic destined for the network behind the Bedford GW and vice versa.

After receiving the SHORTCUT exchange, the Ashby GW establishes an IKEv2 exchange with the Bedford GW and then establishes an IPsec security association between the two. Figure 7 shows the SAs in use after the shortcut has been established.

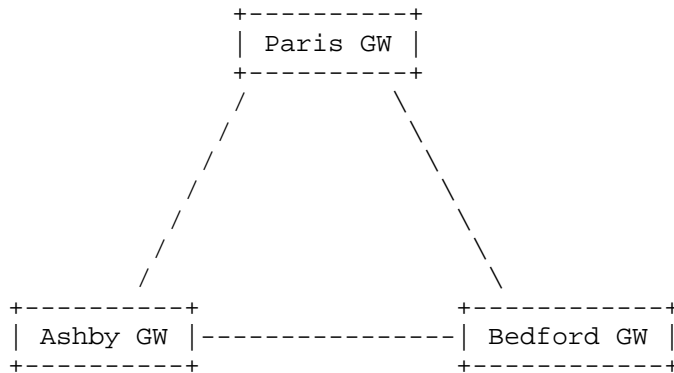


Figure 7: SmithCo IPsec SAs with the shortcut established

After the timeout period specified by the Paris GW, the shortcut between Ashby GW and Bedford GW will be terminated. If the videoconference is finished before that time, the shortcut may also be terminated due to inadequate traffic, at the discretion of the Ashby GW and Bedford GW.

## A.2. Optimization for Videoconference with Partner

In this example, SmithCo has added a partner JonesCo and established an IPsec SA between Paris GW (the main SmithCo office) and Tokyo GW (the main JonesCo office).

Users have initiated a videoconference between the Ashby branch office of SmithCo and the Concord branch office of JonesCo. Each branch office has an IPsec gateway that is configured to send all traffic to the IPsec gateway at the main office for that company. Figure 8 illustrates this initial situation, showing these four IPsec gateways and the IPsec SAs in place when the videoconference starts.

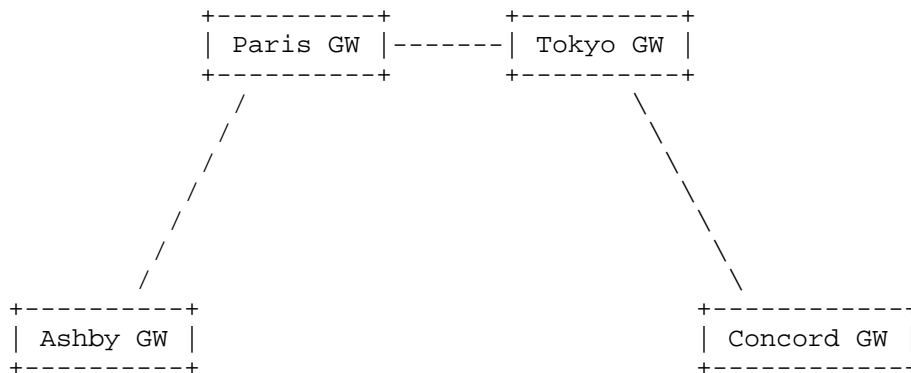


Figure 8: Initial IPsec SAs within SmithCo and JonesCo

All gateways in this example support SHORTCUT exchange. Therefore, they all sent the ADVPN\_SUPPORTED notification payload described in Section 3.4 to each other in their initial IKE exchanges. This means that they are all aware that SHORTCUT exchange may be used on the IPsec SAs illustrated in Figure 8. Further, these gateways have been configured to use SHORTCUT exchange to optimize routing for video traffic within their organizations and among SmithCo and JonesCo gateways.

Once the videoconference begins, the Paris GW notices a large amount of videoconference traffic transiting the Paris GW between the Ashby GW and the Tokyo GW. Therefore, the Paris GW initiates a SHORTCUT exchange to the Ashby GW and the Tokyo GW, suggesting that they establish a shortcut. We will not cover all the details of this process because most are similar to the previous example. However, assume that the Ashby GW and the Tokyo GW have certificates from different CAs and may not be configured to trust each other's CA. Therefore, the Paris GW generates a PSK and sends it to

both the Ashby GW and the Tokyo GW. The Ashby GW and the Tokyo GW use this PSK to establish a shortcut SA, as shown in Figure 9. Because of the Traffic Selectors sent by the Paris GW in the SHORTCUT exchange, this shortcut SA may only be used for video traffic between the Ashby GW and JonesCo.



Figure 9: SmithCo and JonesCo with First Shortcut

After this first shortcut SA has been established, Tokyo GW notices large volumes of video traffic between Ashby GW and Concord GW. Therefore, Tokyo GW initiates a SHORTCUT exchange to the Ashby GW and the Concord GW, suggesting that they establish a shortcut. We do not cover all details of this process because they are mostly similar to the previous example. Again, the Ashby GW and the Concord GW probably have certificates from different CAs so the Tokyo GW generates a PSK and sends it to both the Ashby GW and the Concord GW. The Ashby GW and the Concord GW use this PSK to establish a shortcut SA, as shown in Figure 10. Because of the Traffic Selectors sent by the Tokyo GW in the SHORTCUT exchange, this shortcut SA may only be used for video traffic between the Ashby GW and the Concord GW.



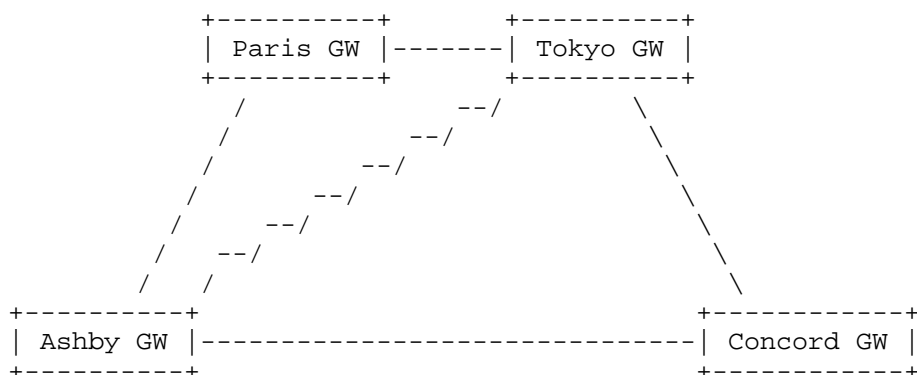


Figure 10: SmithCo and JonesCo with Second Shortcut

After some period, the Ashby GW or the Tokyo GW may realize that no traffic is flowing over the SA between them and therefore decide to terminate this SA. This will result in the SA configuration shown in Figure 11.

Note that this optimal SA configuration has been reached without needing to have any special configuration or global knowledge and it involves multiple domains. The only requirement is a policy on the Paris GW and the Tokyo GW indicating that video traffic between SmithCo and JonesCo should be optimized by creating shortcut SAs.

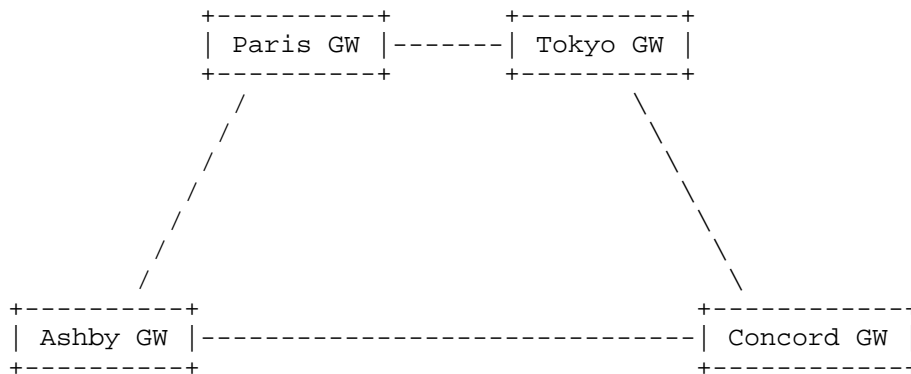


Figure 11: SmithCo and JonesCo in Final Configuration

The shortcut between the Ashby GW and the Concord GW will remain up until its timeout is reached or traffic levels on this SA drop off because the videoconference has finished.

### A.3. Heterogeneous Wireless Networks Traffic

As wireless networks increase their access capacities, denser deployments will become the norm. In addition, we observe an increasing number of cases where operators, for various reasons that are outside of the scope of this document, opt for network deployments that use a variety of coverage sites. In practice, this means that, for instance, macro cells are complemented by smaller cells (pico cells, femto cells, etc.) that boost capacity and improve end-user experience. Today's cellular networks can provide access rates in the order of tens of Mb/s with high quality of service guarantees, and can thus be used as connections where small and medium enterprises can base their VPNs. Within this context, the operator may use different gateways for securing subscriber VPN traffic.

Consider, for example, the case illustrated in Figure 12 where two colleagues from different departments of the same company use multimedia conferencing to collaborate with some customers. Dotted lines in the Figure indicate IP connectivity, while dashed lines indicate an established SA. All gateways and endpoints in the Figure support the protocol described in this document, i.e., they have indicated so to each other as described in Section 3.4. One of them, Peer 1 has joined the teleconference while on the go, but will be arriving at the company office prior to the conclusion of the teleconference. As Peer 1 roams in the

mobile network, changing cell sites as it travels towards the office, the multimedia traffic flows through the Macro GW.

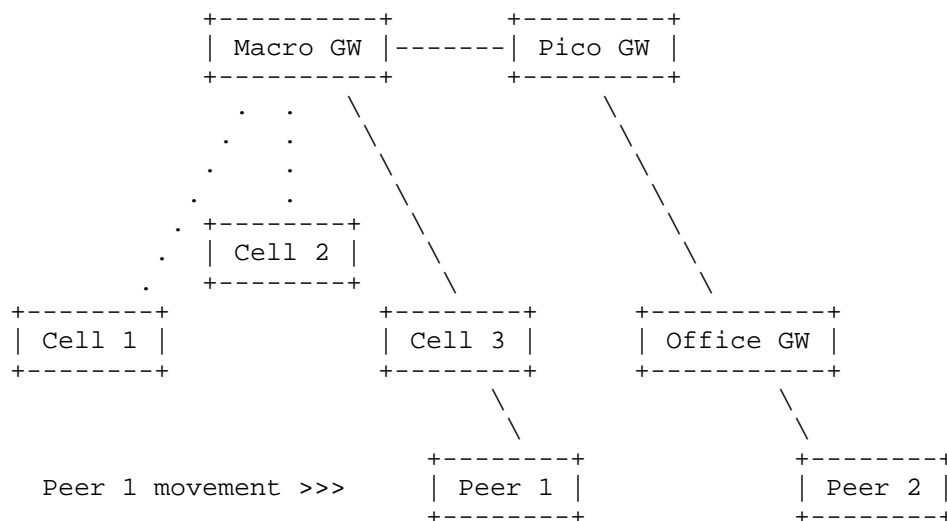


Figure 12: Initial IPsec SAs within the HetNet

Note that both the Macro GW and the Pico GW are in the realm of the mobile operator, while the Office GW is in the realm of the company.

The company and the mobile operator have an already established trust relationship. Moreover, for end-user experience reasons as well as traffic flow optimization both the company network administrators and the mobile operator have policies that favor traffic routes that are contained in the local company network.

Once Peer 1 enters the area of the company campus the wireless network small-cell deployment covering the company buildings is the preferred means of connecting to the network, both from the perspective of the company and the mobile operator. At this stage in our scenario, the fact that Peer 1 is in the coverage area of the Pico GW is recognized by the Macro GW, which initiates (as a shortcut suggester) the procedure described in Section 3. As a result, the first step in the route optimization is performed and Peer 1 sets up the shortcut with the Pico GW, which becomes its shortcut partner.

Figure 13 illustrates the newly established shortcut as well as the fact that Peer 1 continues to use the same radio interface as before, i.e. this scenario does not involve vertical handovers.

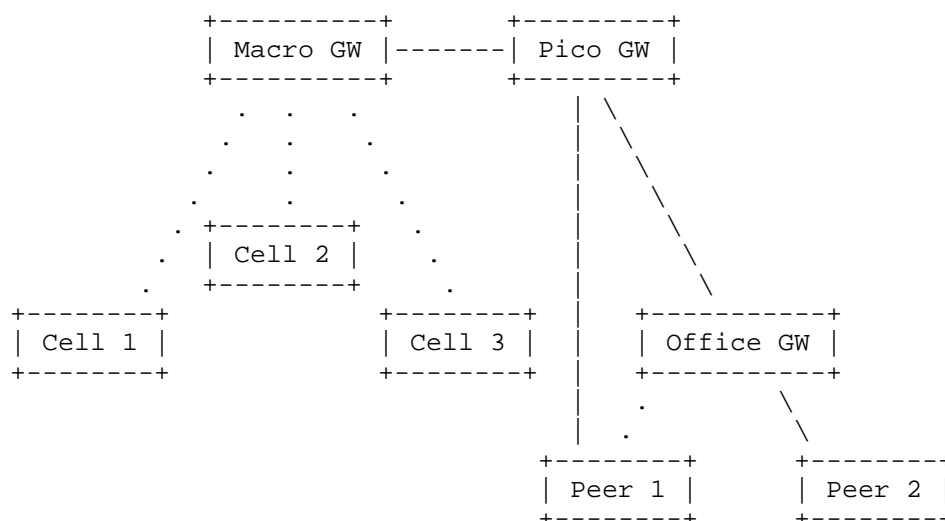


Figure 13: First route optimization within the HetNet

Once Peer 1 moves within the company premises and establishes the shortcut with the operator Pico GW more route optimization opportunities arise, and the ADVPN protocol can implement them without requiring any additional manual configuration neither by the operator nor by the company administrator.

At this stage, we assume that the Pico GW can determine the fact that Peer 1 could become a shortcut partner of the Office GW. Similarly to what was mentioned above, the Pico GW initiates the shortcut (i.e. acts as a shortcut suggester) indicating to Peer 1 and the Office GW that they should establish an SA with each other. The partners agree to these recommendations, as per their respective local policies, and proceed with the establishment. At the end of this process, the configuration is as illustrated in Figure 14.

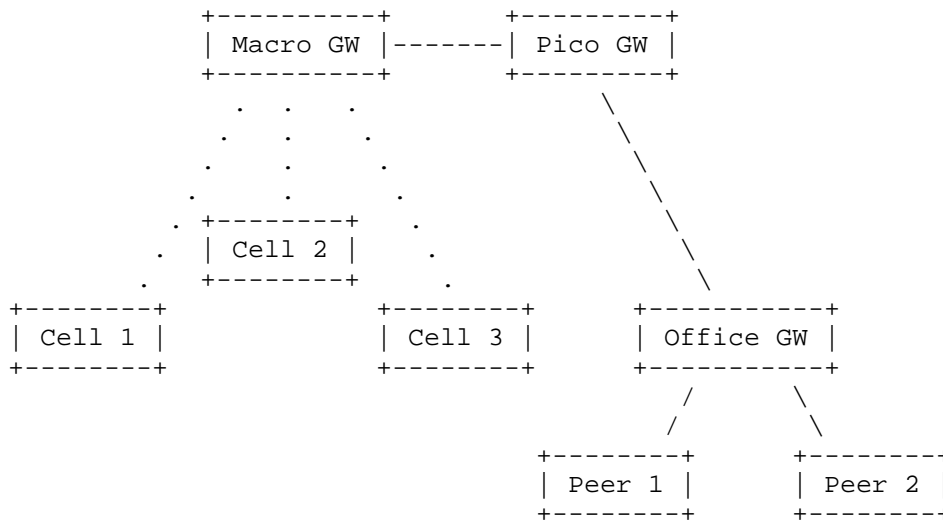


Figure 14: Second route optimization within the HetNet

After this optimization all IPsec traffic is contained within the local small-cell wireless network. Note that the company network may include several pico cells, all of which can establish SAs with the Office GW.

In principle, the protocol can be used to proceed with a further traffic optimization. Namely, Peer 1 and Peer 2 can establish a direct shortcut between each other, i.e. become shortcut partners and thus avoid routing through the Office GW. This is a decision that the Office GW may take based on local connectivity information. In this case, after following the same procedure described earlier, the two Peers will establish an SA, as illustrated in Figure 15.

As Figure 15 shows, traffic may still flow through the Office routers but Peer 1 and Peer 2 do not need to maintain an SA with the Office GW (if there is no other traffic).

Finally, note that, in principle, the Office GW could determine that since no traffic is flowing through its SA with the Pico GW, the respective SA could be temporarily terminated and initiated later on when the need arises. This final configuration is illustrated in Figure 16.

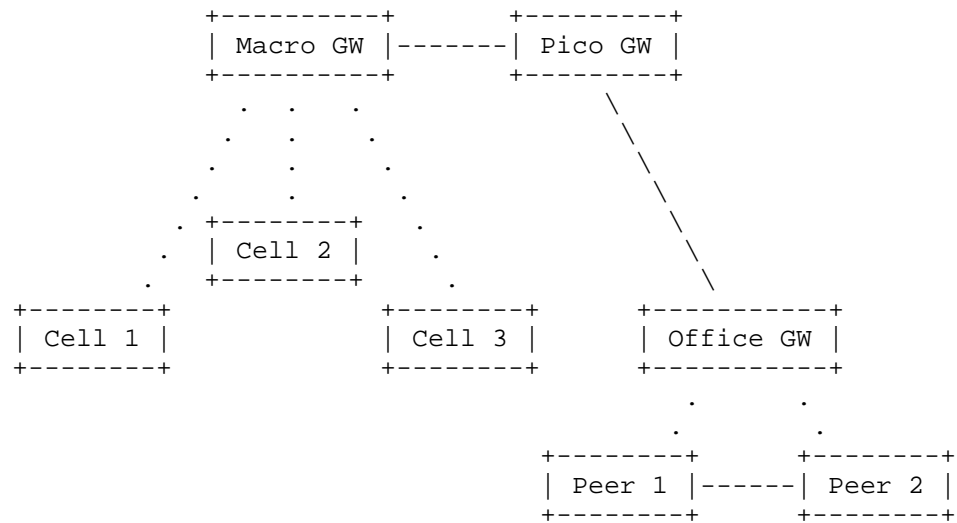


Figure 15: Third route optimization within the HetNet

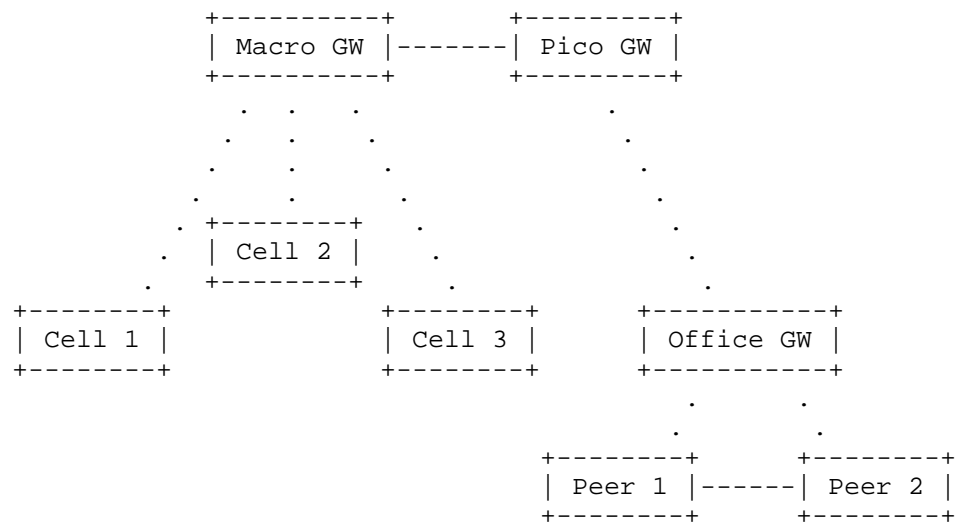


Figure 16: Final configuration



## Appendix B. Comparison Against ADVPN Requirements

This section compares the ADVPN protocol specified in this document against requirements set by [ADVPNreq] (Section 4).

## Requirement #1 :

This section details modifications when an endpoint, a gateway, a spoke and a hub is added or removed or changed.

End points establish a tunnel with a gateway to communicate with another endpoint. The gateway may use the ADVPN protocol to optimize communication and either set up endpoint-to-endpoint communication if both endpoints are attached to the "initial gateway", or to point to a "closer alternative gateway". The ADVPN protocol described in this document, impacts either the two endpoints or the endpoint and the "closer alternative gateway". Hubs or gateways other than the "initial gateway" or the "closer alternative gateway" IPsec configuration are not impacted.

An ADVPN is changed means that its IP address is modified. Updating the outer IP address is the purpose of MOBIKE and involves the two peers connected with their outer IP addresses.

Similarly, removing an endpoint only impacts the IPsec configuration of the gateways or the other endpoint it is communicating with. It is up to local policy that the "initial gateway" decides to keep the IPsec configuration of the endpoint or to remove it once the endpoint has moved to the "alternative gateway that is closer". In the case the "initial gateway" does not remove the SAs associated to the endpoint, the endpoint is considered attached simultaneously to two gateways.

The use of ADVPN with an endpoint that is added, removed or changed results in local IPsec configuration modifications. Only gateways that the endpoint is attached to are modified. Other gateways, spokes and hub are not impacted.

Gateways may accept traffic from another gateway. The traffic may be the one associated to an endpoint or to a gateway. In the first case, the gateway is considered as the "closer alternative gateway" as discussed above. The second case occurs if the "initial gateway" tunnels traffic from an "alternative gateway" to a "closer alternative gateway". It may then use ADVPN so traffic directly goes from the "alternative gateway" to the "closer alternative gateway". The IPsec configuration is then



updated on both the "alternative gateways" and the "closer alternative gateway".

Similarly, when the "closer alternative gateway" is removed, only gateways and endpoints attached to these gateways are impacted.

The use of ADVPN with a gateway that is added or removed results in local IPsec configuration modifications. Only gateways attached to are modified. Others gateways, spokes and hub are not impacted.

Spokes are between endpoints and gateways. Unlike end points, they have a complete network, and they are attached to a hub. If a spoke-to-spoke communication is set with ADVPN, then IPsec policies of the two spokes are updated. The hub may not modify its IPsec policies. Similarly, when a spoke is removed, the IPsec policies of the other spokes are updated.

The use of ADVPN with a spoke that is added or removed results in local IPsec configuration modifications. Only spokes attached to the one being removed are modified. Other gateways, spokes and hubs are not impacted.

Anytime a shortcut is established, new security policies are created on the shortcut initiator and the shortcut responders. ADVPN avoids these security policies to be created manually. In addition, it uses PSK authentication, which is, reduces latency and round trip times over other authentication methods like EAP-SIM.

Additionally, PROTECTED\_DOMAIN capability can provide the initial protected domain subnet information to all its endpoints, from a trusted suggester. The trusted suggester provides periodic update on protected domain subnet information its endpoints. This periodic update, avoids requirement of any manual configuration change required, whenever new endpoint is added or existing endpoint is removed/updated, within given ADVPN protected domain.

#### Requirement #2 :

The solution specified in this document does not require any manual intervention for establishing a direct tunnel between endpoints. As described in Requirement #1 above and in Section 4. , SPD and SAD entries get automatically updated without any manual intervention. If an IP address of a shortcut partner has changed, MOBIKE can help in updating SPD entries automatically. If an IP address change happens after a reboot of a shortcut partner, then the peer shortcut partner will detect this condition using IKEv2 keep-alive and can divert the traffic back to the "initial-gateway". Once rebooted, the shortcut

partner will establish IPsec tunnel with the "initial-gateway". At this stage, the "initial-gateway" will send SHORTCUT exchange to the shortcut partners, to establish shortcut tunnel with new IP address of shortcut partners.

Requirement #3 :

This draft enables shortcut partners to establish a secure channel between them automatically. This will allow other tunneling and routing protocols to establish direct tunnels or exchange route updates. However, how a routing protocol module is aware of this new shortcut tunnel (or how it exchanges route updates), with shortcut partners, using shortcut tunnel or how other tunneling protocols establish direct tunnel between shortcut partners, is specific to the vendor implementation. Thus it is out of scope of this specification.

Requirement #4 :

While this document describes the syntax of SHORTCUT messages, it makes no mandates about the policy for initiating shortcuts, nor about the policy for accepting or rejecting shortcuts. Some endpoints may agree to accept shortcuts from any peer, as long as the traffic selectors are a subset of those that the SPD says should go to that peer. Others may filter the shortcuts based on IKE ID, so that they do not open tunnels to endpoints outside their administrative domain. Future documents may profile such behavior.

Requirement #5 :

When a spoke becomes compromised it may compromise inbound/outbound communications associated with it. A compromised spoke may want to use ADVPN in order to corrupt additional traffic that go through other gateways and spokes. The ADVPN protocol provides facilities to create shortcuts, however the shortcuts for given traffic is always triggered by an endpoint dealing with that traffic. As a result, a compromised host does not affect the security of other unrelated peers.

Requirement #6 :

This document addresses seamless session handoffs when endpoints roam around different policy boundaries. A detailed explanation about this is given in Section A.3.

Requirement #7 :

When a shortcut between different gateways is created for a given endpoint-to-endpoint session, the endpoint-to-endpoint communication is not impacted by the shortcut. In other words, this is transparent to the endpoints. More precisely, a new shortcut partner is created on the two alternate gateways, spokes or hubs. This modifies the communication path, but not the session itself.

Requirement #8 :

This document does not explicitly detail all NAT scenarios, in this version at least, but does provide two mechanisms that address this.

When the suggester proposes a shortcut to the shortcut peers, the suggester has performed IKE AUTH and can detect the shortcut peers are behind a NAT. This can be done with multiple ways including the NAT\_DETECTION\_SOURCE\_IP / NAT\_DETECTION\_DESTINATION\_IP Notify Payload exchange, the UDP encapsulation and use of port 4500. In most cases, when the shortcut peer is behind a NAT, inbound IKEv2 and IPsec traffic are sent through a specific port.

The ADVPN protocol described in this document enables the suggester to specify each shortcut peer whether the other peer is behind a NAT or not by setting the NAT bit in the ADVPN\_INFO Notification. When this bit is set, it indicates UDP encapsulation MUST be used for IKEv2. In addition, the ADVPN\_INFO Notification also specifies the UDP Port on which the shortcut peer is reachable.

Another advantage of the ADVPN protocol is that the SHORTCUT exchange are sent to the shortcut peer by the suggester, and the suggester can determine whether a shortcut can be established or not. If the shortcut cannot be established, for example if the shortcut peers are both behind a NAT, then the suggester MAY forgo the establishment of the shortcut and thus avoid communication disruption due to NATs.

To address scenario where both the shortcut partners are behind NAT device, SHORTCUT exchange includes each other's peer UDP port number, that the shortcut suggester is receiving IKE and IPSec traffic. This information should help the shortcut partner to reach each other in certain types of NAT deployments.

Similarly the ADVPN protocol has designed optional exchanges that MAY in the future be designed to address specific NAT issues. For example, [MEDIATION] MAY be added for double NAT and hole punching.

Note that ADVPN is essentially based on the use of tunnel mode which makes TS selectors independent from the IP addresses of the outer header. Thus, the use of the tunnel mode makes ADVPN more resilient to NAT compared to transport mode.

Requirement #9 :

This document does not create a MIB. However, it does define several events that can be reportable:

- \* The gateway suggests a shortcut
- \* The peer accepts or rejects a shortcut (the former involves a change in policy)
- \* A shortcut times out (again involves a change in policy)

Requirement #10 :

The document is independent of administrative domains. One of the properties that may be associated with administrative domains is a set of one or more trust anchors used to issue certificates for VPN gateways and endpoints. To avoid the need for cross-trusting these anchors, this document offers the option of using dynamically-generated PSKs.

Requirement #11 :

While this document describes the syntax of SHORTCUT messages, it makes no mandates about the policy for initiating shortcuts, or the policy for accepting and rejecting shortcuts. Some endpoints may agree to accept shortcuts from any peer, as long as the Traffic Selectors are a subset of those that the SPD says should go to that peer. Others may filter the shortcuts based on IKE ID, so that they do not open tunnels to endpoints outside their administrative domain. Future documents may profile such behavior.

Requirement #12 :

The Traffic Selectors in the SHORTCUT message can be used to specify both multicast routing protocols, such as IGMP, and multicast traffic through the use of multicast addresses in selectors. With this, the SHORTCUT tunnels can be used to pass multicast and multicast routing traffic.

Requirement #13 :

This document defines several events that can be logged and monitored:

- \* The gateway suggests a shortcut
- \* The peer accepts or rejects a shortcut (the former involves a change in policy)
- \* A shortcut times out (again involving a change in policy)

A status report listing active shortcuts for a particular gateway is also possible and recommended for implementations.

Requirement #14 :

L3VPNs all use some kind of transport-layer protocol. GRE uses protocol number 43, IP-in-IP uses 4, and so on. Selectors for these protocols can easily be specified using the TS payloads included in SHORTCUTs. The additional information that may be needed to set up a tunnel for each of these protocols is outside the scope of this document.

## Requirement #15 :

QoS policy is outside the scope of this document. However, the mandate of RFC 5996 to allow multiple parallel SAs for different classes of QoS applies to peers that a VPN box learns about through SHORTCUT messages. This means that QoS policy can still be enforced. If there are any additional requirements to be addressed with respect to QoS, the SHORTCUT message structure can be extended to support identified QoS attributes that should be exchanged.

## Requirements #16

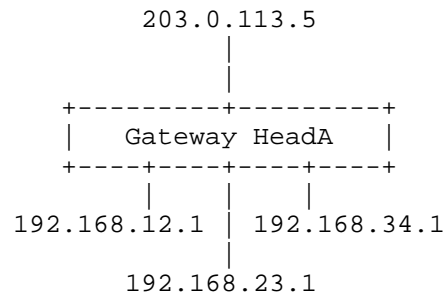
ADVPN does not make spokes, hubs or gateway single points of failure. By design, ADVPN provides two types of resiliency: Topological resiliency by creating shortcuts. These shortcuts provide alternate path, and make communications resilient in case a hub or spoke fails. In addition, the use of tunnel mode between gateways makes possible the use of MOBIKE that provides end point resiliency.

## Appendix C.

## PROTECTED\_DOMAIN Example

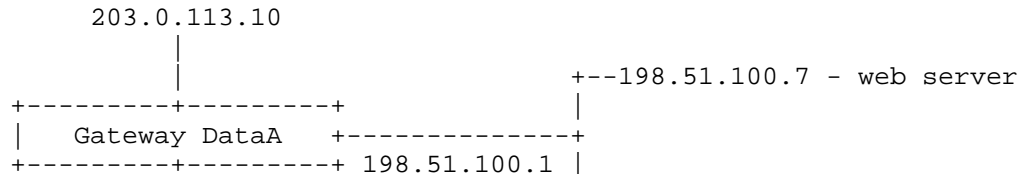
This appendix contains an example of how the PROTECTED\_DOMAIN response is created. As this example requires multiple subnets, we will use the non-routable addresses from RFC 1918 in addition to the documentation subnets from RFC 5737.

Assume that Company A has two major locations. First, at the company headquarters there are three non-routable subnets: 192.168.12.0/24, 192.168.23.0/24, and 192.168.34.0/24. At this location, the VPN gateway has four interfaces: one towards each of the three internal subnets, and one external interface that connects to the Internet with IP address 203.0.113.5, as illustrated below.



Traffic to the Internet and to partner sites gets NAT-ted, but non-routable addresses are allowed within the internal VPN.

Company A has also a datacenter, at a location different from the headquarters, which is implemented as a non-routable subnet: 192.168.45.0/24. In addition, there is also a routable subnet with some front-end servers: 198.51.100.0/24, known as the DMZ. The gateway has an external IP address 203.0.113.10. We would like access to the DMZ to go through the VPN for communications from within the company, but it can go either through the VPN or outside the VPN for traffic from partners.

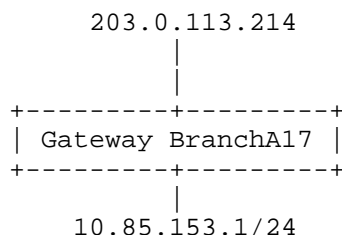


```

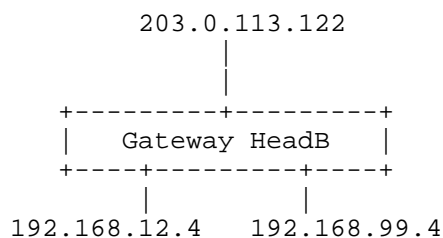
 |
192.168.45.1 +---192.51.100.5 - SMTP server

```

In addition to the two main locations introduced above, Company A also has many smaller locations. Each of those has one /24 non-routable subnet. The initial configuration of each of those small gateways is such that the internal subnet is different from that in all the other small gateways.



Company A also has a supplier, Company B, that has their own gateway with some routable and some non-routable addresses. They have a VPN tunnel configured with Gateway DataA. Although there is some overlap in the protected domains, the only non-routable addresses that go through this VPN are the 192.168.23.0/24 subnet that is behind HeadA, and the 192.168.99.0/24 that is behind HeadB. Others are either blocked, or NATted behind the address of HeadB



Only the branch office gateways and Gateway HeadA need the PROTECTED\_DOMAIN (see Section 3.9. ) configuration attribute. Gateway HeadB has a static policy, and Gateway DataA will not send to it a PROTECTED\_DOMAIN according to the established policy. The field contains the union of all the sets of addresses for which the DataA server is willing to forward traffic.

For the BranchA17 gateway, the initial configuration is very simple:



- . One peer is defined: Gateway DataA (203.0.113.10)
- . Either a CA certificate and a DN for DataA, or a PSK
- . An internal network: 10.85.153.0/24
- . Gateway DataA is a trusted suggester

In total, there are nine other branch office gateways other than BranchA17, and of course there is Gateway HeadA and Gateway HeadB. Only Gateway DataA knows about all of them. So the PROTECTED\_DOMAIN it sends to other gateways from the same administrative domain is as follows:

- . 192.168.12.0/24 (from behind HeadA)
- . 192.168.23.0/24 (from behind HeadA)
- . 192.168.34.0/24 (from behind HeadA)
- . 192.168.45.0/24 (from behind DataA itself)
- . 198.51.100.0/24 (DMZ behind DataA itself)
- . 192.168.99.0/24 (from behind HeadB)
- . 10.85.101.0/24 (from behind BranchA01)
- . 10.85.104.0/24 (from behind BranchA02)
- . 10.85.125.0/24 (from behind BranchA03)
- . 10.85.131.0/24 (from behind BranchA04)
- . 10.85.139.0/24 (from behind BranchA11)
- . 10.85.143.0/24 (from behind BranchA12)
- . 10.85.150.0/24 (from behind BranchA16)
- . 10.85.153.0/24 (from behind BranchA17)
- . 10.85.159.0/24 (from behind BranchA18)
- . 10.85.162.0/24 (from behind BranchA19)

Every time a new branch gateway is added, its protected domain is added to the SPD of DataA, and this also updates the contents of the PROTECTED\_DOMAIN that it sends. As described in Section 4.2. , the content expires in BranchA17 after a while, so the cache expiry time is also the time it takes for such a change to propagate to the other branch offices.

When BranchA17 receives this PROTECTED\_DOMAIN, it removes its own protected domain and anything else for which it has a static configuration, and adds the rest into the SPD as traffic that is protected and tunneled to the trusted suggester (Gateway DataA).

A more complex scenario would send a different PROTECTED\_DOMAIN also to the partner gateway HeadB. For example, it could send the following PROTECTED\_DOMAIN:

- . 192.168.23.0/24 (This is a network behind Gateway HeadA)
- . 192.168.45.1/32 (This is a single address at Gateway DataA)

The reason for having this configuration is that the IP addresses behind branch office gateways are not guaranteed to be unique outside of the administrative domain. So Gateway DataA NATs these addresses behind its own IP address, which then has to be part of the protected domain. Note that under this specification, such traffic cannot be "shortcutted", because we don't have a way to tell the BranchA17 gateway to NAT these packets when sending them through the shortcut tunnel (TBD)

#### Authors' Addresses

Praveen Sathyanarayan  
Juniper Networks, Inc.  
1194 N. Mathilda Ave.  
Sunnyvale, CA 94089  
USA

Email: praveenys@juniper.net

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

Email: shanna@juniper.net

Suresh Nagavenkata Melam  
Juniper Networks, Inc.  
1194 N. Mathilda Ave.  
Sunnyvale, CA 94089  
USA

Email: nmelam@juniper.net

Yoav Nir  
Check Point Software Technologies Ltd.  
5 Hasolelim st.

Tel Aviv 6789735  
Israel

Email: [ynir@checkpoint.com](mailto:ynir@checkpoint.com)

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

Email: [mglt.ietf@gmail.com](mailto:mglt.ietf@gmail.com)

Kostas Pentikousis  
EICT GmbH  
Torgauer Strasse 12-15  
10829 Berlin  
Germany

Email: [k.pentikousis@eict.de](mailto:k.pentikousis@eict.de)



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].



## 2. Motivation

There are two major topologies - Star topology and full-mesh topology - to communicate securely on over-lay network by using IPsec.

Figure.1 shows star topology. The number of IPsec connection is the same as the number of CPEs (CPE). Authentication, Authorization and Accounting (AAA) of each CPE can be achieved on the gateway.

The problem of the star topology is all the traffic go through the gateway, then it causes high load and latency.

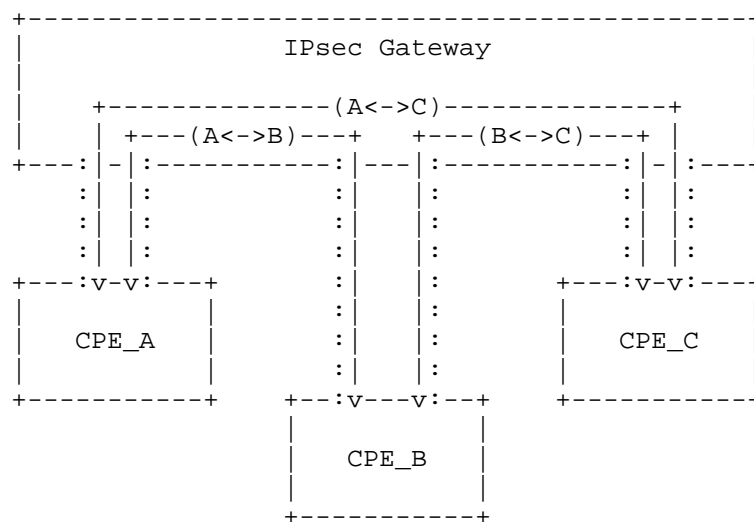


Figure 1

Figure.2 shows Full-mesh topology. There is no gateways. Each CPE establishes IPsec connection independently. The latency on this topology is relatively low compared to star topology.

In large system, there are huge number ( $(N^2-N)/2$ ) of IPsec connections. AAA of each CPE is hard to manage in this topology. Moreover, when a CPE is added, removed or changed, reconfiguration is needed for all rest of the CPEs.

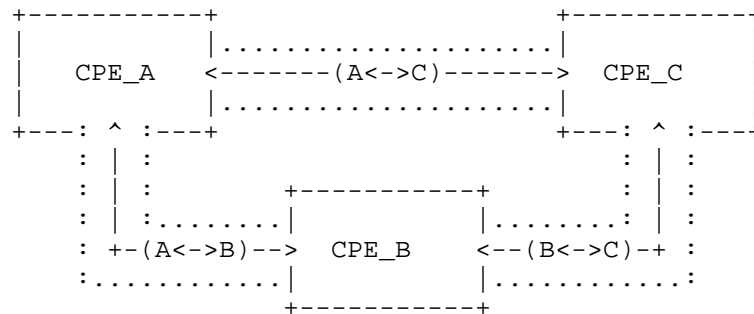


Figure 2

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.

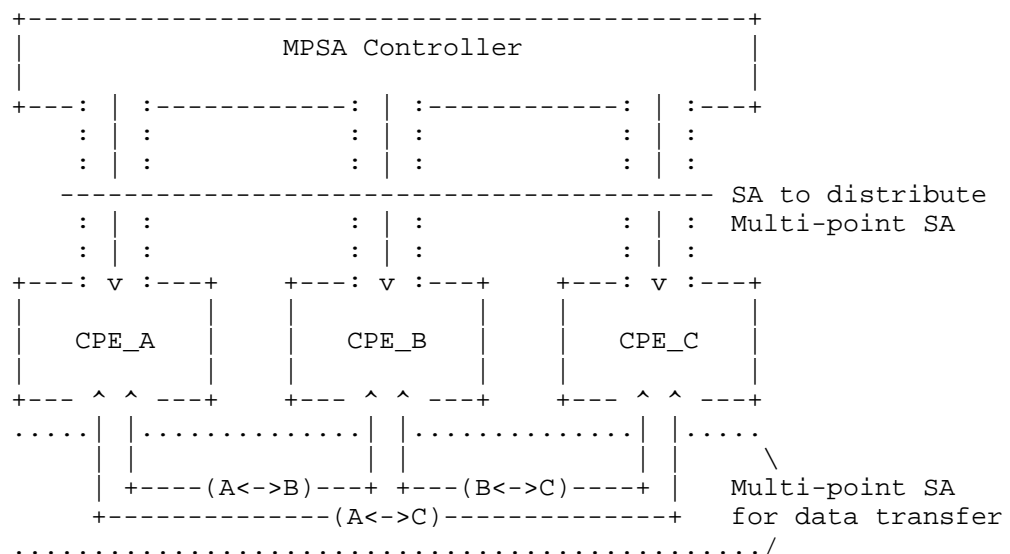


Figure 3

### 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.

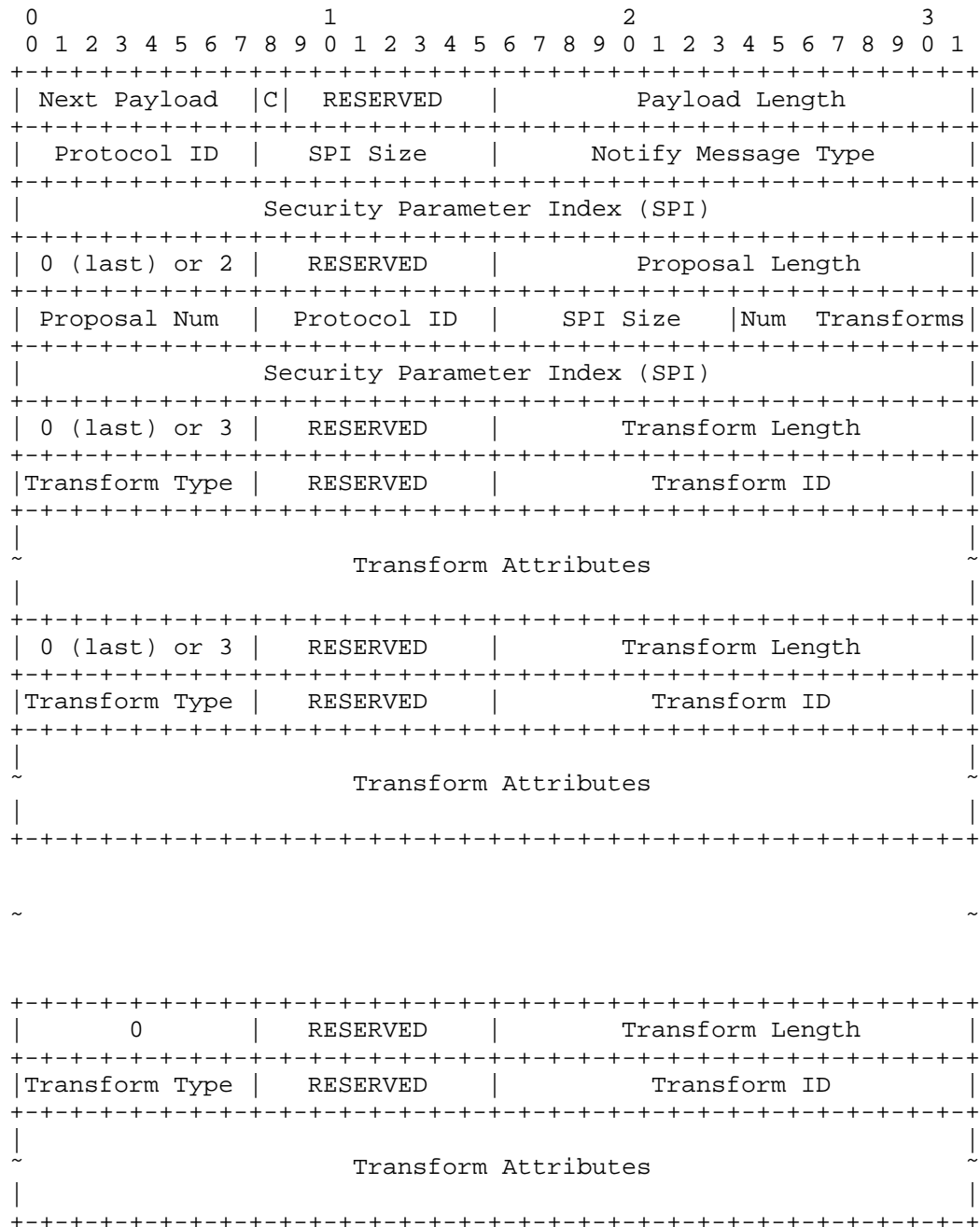
| Name            | Number |                  |  |
|-----------------|--------|------------------|--|
| -----           |        |                  |  |
| ROLL1_ROLLOVER1 | 1      |                  |  |
|                 |        |                  |  |
| Attribute Type  | Value  | Attribute Format |  |
| -----           |        |                  |  |
| 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.

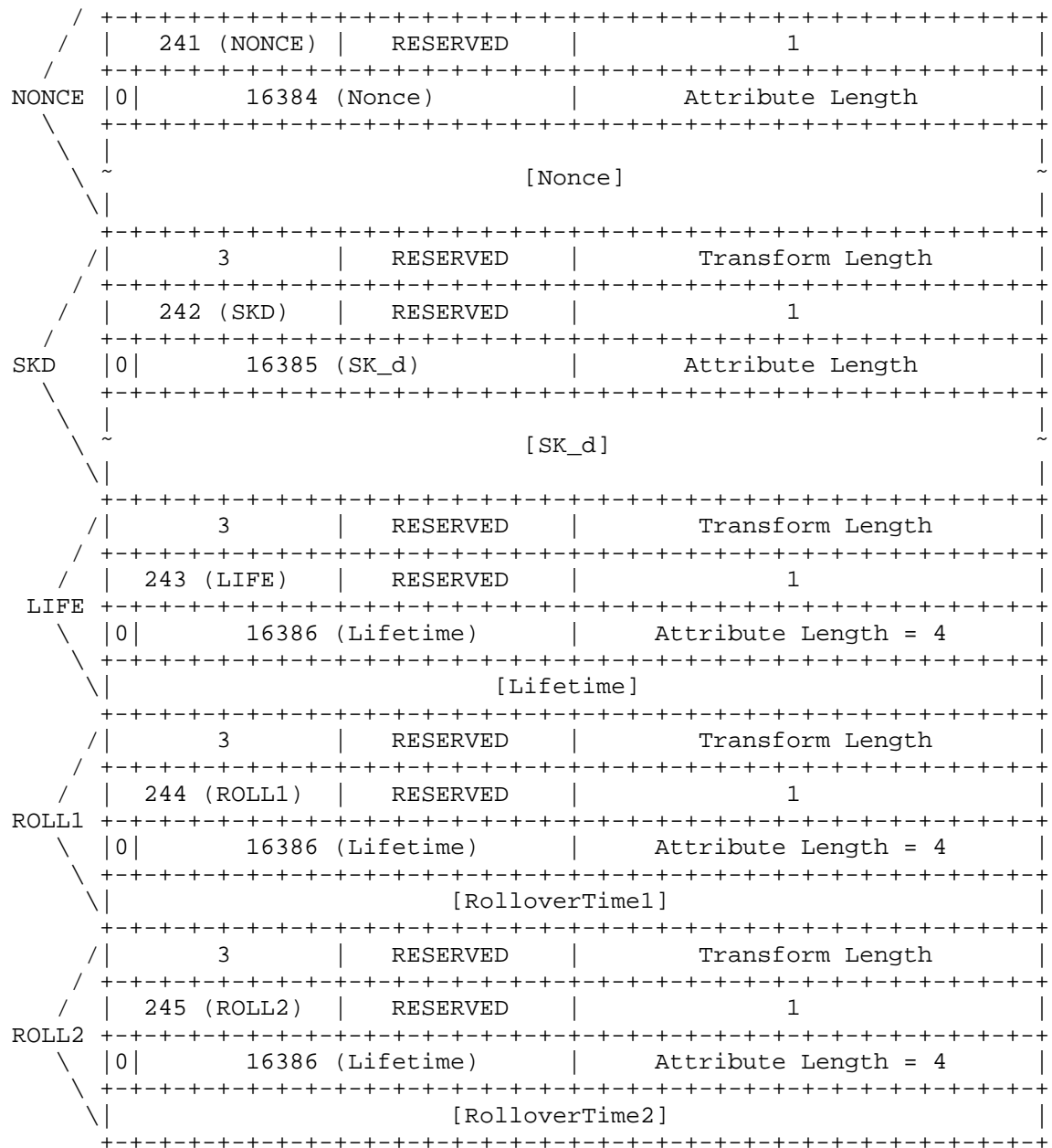
```

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 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |  |
|        | +++++                          |                 |   |   |   |   |   |   |   |   |              |          |   |   |   |   |   |   |   |   |                       |                |   |   |   |   |   |   |   |   |                |   |   |   |   |   |   |   |   |   |  |
| /      | 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