Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: October 21, 2017

S. Fluhrer
D. McGrew
P. Kampanakis
Cisco Systems
April 19, 2017

Postquantum Preshared Keys for IKEv2
draft-fluhrer-qr-ikev2-04

Abstract

The possibility of quantum computers pose a serious challenge to cryptography algorithms widely today. IKEv2 is one example of a cryptosystem that could be broken; someone storing VPN communications today could decrypt them at a later time when a quantum computer is available. It is anticipated that IKEv2 will be extended to support quantum secure key exchange algorithms; however that is not likely to happen in the near term. To address this problem before then, this document describes an extension of IKEv2 to allow it to be resistant to a Quantum Computer, by using preshared keys.

Status of This Memo

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

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

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

This Internet-Draft will expire on October 21, 2017.

Copyright Notice

Table of Contents

1. Introduction

It is an open question whether or not it is feasible to build a quantum computer (and if so, when might one be implemented), but if it is, many of the cryptographic algorithms and protocols currently in use would be insecure. A quantum computer would be able to solve DH and ECDH problems, and this would imply that the security of existing IKEv2 systems would be compromised. IKEv1 when used with strong preshared keys is not vulnerable to quantum attacks, because those keys are one of the inputs to the key derivation function. If the preshared key has sufficient entropy and the PRF, encryption and authentication transforms are postquantum secure, then the resulting system is believed to be quantum resistant, that is, believed to be invulnerable to an attacker with a Quantum Computer.

This document describes a way to extend IKEv2 to have a similar property; assuming that the two end systems share a long secret key, then the resulting exchange is quantum resistant. By bringing postquantum security to IKEv2, this note removes the need to use an obsolete version of the Internet Key Exchange in order to achieve that security goal.

The general idea is that we add an additional secret that is shared between the initiator and the responder; this secret is in addition

to the authentication method that is already provided within IKEv2. We stir in this secret into the SK_d value, which is used to generate the key material (KEYMAT) keys and the SKEYSEED for the child SAs; this secret provides quantum resistance to the IPsec SAs (and any child IKE SAs). We also stir in the secret into the SK_pi, SK_pr values; this allows both sides to detect a secret mismatch cleanly.

It was considered important to minimize the changes to IKEv2. The existing mechanisms to do authentication and key exchange remain in place (that is, we continue to do (EC)DH, and potentially a PKI authentication if configured). This does not replace the authentication checks that the protocol does; instead, it is done as a parallel check.

1.1. Changes

Changes in this draft from the previous versions

draft-03

- Modified how we stir the PPK into the IKEv2 secret state

- Modified how the use of PPKs is negotiated

draft-02

- Simplified the protocol by stirring in the preshared key into the child SAs; this avoids the problem of having the responder decide which preshared key to use (as it knows the initiator identity at that point); it does mean that someone with a Quantum Computer can recover the initial IKE negotation.

- Removed positive endorsements of various algorithms. Retained warnings about algorithms known to be weak against a Quantum Computer

draft-01

- Added explicit guidance as to what IKE and IPsec algorithms are Quantum Resistant

draft-00

- We switched from using vendor ID’s to transmit the additional data to notifications

- We added a mandatory cookie exchange to allow the server to communicate to the client before the initial exchange

- We added algorithm agility by having the server tell the client what algorithm to use in the cookie exchange

- We have the server specify the PPK Indicator Input, which allows the server to make a trade-off between the efficiency for the search of the clients PPK, and the anonymity of the client.

- We now use the negotiated PRF (rather than a fixed HMAC-SHA256) to transform the nonces during the KDF

1.2. 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].

2. Assumptions

We assume that each IKE peer has a list of Postquantum Preshared Keys (PPK) along with their identifiers (PPK_id), and any potential IKE initiator has a selection of which PPK to use with with any specific responder. In addition, the implementation has a configurable flag that determines whether this postquantum preshared key is mandatory. This PPK is independent of the preshared key (if any) that the IKEv2 protocol uses to perform authentication.

3. Exchanges

If the initiator is configured to use a postquantum preshared key with the responder (whether or not the use of the PPK is optional), then it will include a notify payload in the initial exchange as follows:

```
Initiator                       Responder
-------------------------------------------------------------------
HDR, SAi1, KEi, Ni, N(PPK_SUPPORT)  --->
```

N(PPK_SUPPORT) is a status notification payload with the type [TBA]; it has a protocol ID of 0, and no SPI and no notification data associated with it.

If the initiator needs to resend this initial message with a cookie (because the responder response included a cookie notification), then the resend would include the PPK_SUPPORT notification if the original message did.

When the responder receives this initial exchange with the notify, then it MUST check if has a PPK configured. If it does, it MUST

reply with the IKE initial exchange including a notification in response.

```
Initiator                          Responder
-------------------------------------------------------------------
                <--- HDR, SAr1, KEr, Nr, [CERTREQ], N(PPK_SUPPORT)
```

If the responder does not have a PPK configured, then it continues with the IKE protocol as normal, not including the notify.

When the initiator receives this reply, it checks whether the responder included the PPK_SUPPORT notify. If the responder did not, then the initiator MUST either proceed with the standard IKE negotiation (without using a PPK), or abort the exchange (for example, because the initiator has the PPK marked as mandatory). If the responder did include the PPK_SUPPORT notify, then it selects a PPK, along with its identifier PPK_id. Then, it computes this modification of the standard IKE key derivation:

```
 SKEYSEED = prf(Ni | Nr, g^ir)
 {SK_d' | SK_ai | SK_ar | SK_ei | SK_er | SK_pi' | SK_pr' )
                 = prf+ (SKEYSEED, Ni | Nr | SPIi | SPIr }
 SK_d = prf(PPK, SK_d')
 SK_pi = prf(PPK, SK_pi')
 SK_pr = prf(PPK, SK_pr')
```

That is, we use the standard IKE key derivation process except that the three subkeys SK_d, SK_pi, SK_pr are run through the prf again, this time using the PPK as the key.

The initiator then sends the initial encrypted message, including the PPK_id value as follows:

```
Initiator                          Responder
-------------------------------------------------------------------
HDR, SK {IDi, [CERT,] [CERTREQ,]
    [IDr,] AUTH, SAi2,
    TSi, TSr, N(PPK_IDENTITY)(PPK_id)}  --->
```

N(PPK_IDENITY) is a status notification payload with the type [TBA]; it has a protocol ID of 0, and no SPI and has a notification data that consists of the identifier PPK_id.

When the responder receives this encrypted exchange, it first computes the values:

```
SKEYSEED = prf(Ni | Nr, g^ir)
{SK_d' | SK_ai | SK_ar | SK_ei | SK_er | SK_pi' | SK_pr' }
                = prf+ (SKEYSEED, Ni | Nr | SPIi | SPIr )
```

It then uses the SK_ei value to decrypt the message; and then finds the PPK_id value attached to the notify. It then scans through the payload for the PPK_id attached to the N(PPK_IDENTITY); if it has no such PPK, it fails the negotiation. If it does have a PPK with that identity, it further computes:

```
SK_d = prf(PPK, SK_d')
SK_pi = prf(PPK, SK_pi')
SK_pr = prf(PPK, SK_pr')
```

And computes the enchange (validating the AUTH payload that the initiator included) as standard.

This table summarizes the above logic by the responder

Received PPK_SUPPORT	Have PPK	PPK Mandatory	Action
No	No	*	Standard IKE protocol
No	Yes	No	Standard IKE protocol
No	Yes	Yes	Abort negotiation
Yes	No	*	Standard IKE protocol
Yes	Yes	*	Include PPK_SUPPORT

When the initiator receives the response, then (if it is configured to use a PPK with the responder), then it checks for the presense of the notification. If it receives one, it marks the SA as using the configured PPK to generate SK_d, SK_pi, SK_pr (as shown above); if it does not receive one, it MUST either abort the exchange (if the PPK was configured as mandatory), or it MUST continue without using the PPK (if the PPK was configured as optional).

If the initial exchange had PPK_SUPPORT sent by both the initiator and the responder, and the initiator does not include a PPK_NOTIFY notification, then the responder SHOULD fail the exchange.

With this protocol, the computed SK_d is a function of the PPK, and assuming that the PPK has sufficient entropy (for example, at least 2**256 possible values), then even if an attacker were able to recover the rest of the inputs to the prf function, it would be infeasible to use Grover's algorithm with a Quantum Computer to recover the SK_d value. Similarly, every child SA key is a function of SK_d, hence all the keys for all the child SAs are also quantum resistant (assuming that the PPK was high entropy and secret, and that all the subkeys are sufficiently long). However, this quantum

resistance does not extend to the initial SK_ei, SK_er keys; an implementation MAY rekey the initial IKE SA immediately after negotiating it; this would reduce the amount of data available to an attacker with a Quantum Computer.

4. PPK ID format

This standard requires that both the initiator and the responder have a secret PPK value, with the responder selecting the PPK based on the PPK_ID that the initiator sends. In this initial standard, both the initator and the responder are configured with fixed PPK and PPK_ID values, and do the look up based on that. It is anticipated that later standards will extend this technique to allow dynamically changing PPK values. To facilitate such an extension, we specify that the PPK_ID that the initiator sends will have its first octet be the PPK ID Type value, which is encoded as follows:

PPK ID Type	Value
PPK_ID_OPAQUE	0
PPK_ID_FIXED	1
RESERVED TO IANA	2-127
Reserved for private use	128-255

For PPK_ID_OPAQUE, the format of the PPK ID (and the PPK itself) is not specified by this document; it is assumed to be mutually intelligible by both by initiator and the responder. This PPK ID type is intended for those implementations that choose not to disclose the type of PPK to active attackers.

For PPK_ID_FIXED, the format of the PPK ID and the PPK are fixed octet strings; the remaining bytes of the PPK_ID are a configured value. We assume that there is a fixed mapping between PPK_ID and PPK, which is configured locally to both the initiator and the responder. The responder can use to do a look up the passed PPK_id value to determine the corresponding PPK value. Not all implementations are able to configure arbitrary octet strings; to improve the potential interoperability, it is recommended that, in the PPK_ID_FIXED case, both the PPK and the PPK_ID strings be limited to the base64 character set, namely the 64 characters 0-9, A-Z, a-z, + and /.

The PPK ID type values 2-127 are reserved for IANA; values 128-255 are for private use among mutually consenting parties.

5. PPK Distribution

PPK_id's of the type PPK_ID_FIXED (and the corresponding PPKs) are assumed to be configured within the IKE device in an out-of-band fashion. While the method of distribution is a local matter, one suggestion would be to reuse the format within [RFC6030], with the Key Id field being the PPK_ID (without the 0x01 prefix for a PPK_ID_FIXED), and with the PPK being the secret, and the algorithm as PIN ("Algorithm=urn:ietf:params:xml:ns:keyprov:pskc:pin").

6. Upgrade procedure

This algorithm was designed so that someone can introduce PPKs into an existing IKE network without causing network disruption.

In the initial phase of the network upgrade, the network administrator would visit each IKE node, and configure:

- The set of PPKs (and corresponding PPK_id's) that this node would need to know

- For each peer that this node would initiate to, which PPK that we would use

- That the use of PPK is currently optional

With this configuration, the node will continue to operate with nodes that have not yet been upgraded. This is due to the PPK_SUPPORT notify; if the initiator has not been upgraded, it will not send the PPK_SUPPORT notify (and so the responder will know that we will not use a PPK); if the responder has not been upgraded, it will not send the PPK_SUPPORT notify (and so the initiator will know not to use a PPK). And, if both peers have been upgraded, they will both realize it, and in that case, the link will be quantum secure

As an optional second step, after all nodes have been upgraded, then the administrator may then go back through the nodes, and mark the use of PPK as mandatory. This will not affect the strength against a passive attacker; it would mean that an attacker with a Quantum Computer (which is sufficiently fast to be able to break the (EC)DH in real time would not be able to perform a downgrade attack).

7. Security Considerations

Quantum computers are able to perform Grover's algorithm; that effectively halves the size of a symmetric key. Because of this, the user SHOULD ensure that the postquantum preshared key used has at

least 256 bits of entropy, in order to provide a 128 bit security level.

Although this protocol preserves all the security properties of IKE against adversaries with conventional computers, this protocol allows an adversary with a Quantum Computer to decrypt all traffic encrypted with the initial IKE SA. In particular, it allows the adversary to recover the identities of both sides. If there is IKE traffic other than the identities that need to be protected against such an adversary, one suggestion would be to form an initial IKE SA (which is used to exchange identities), perhaps by using the protocol documented in RFC6023. Then, you would immediately create a child IKE SA (which is used to exchange everything else). Because the child IKE SA keys are a function of SK_d, which is a function of the PPK (among other things), traffic protected by that SA is secure against Quantum capable adversaries.

In addition, the policy SHOULD be set to negotiate only quantum-resistant symmetric algorithms; while this RFC doesn't claim to give advise as to what algorithms are secure (as that may change based on future cryptographical results), here is a list of defined IKEv2 and IPsec algorithms that should NOT be used, as they are known not to be Quantum Resistant

Any IKE Encryption algorithm, PRF or Integrity algorithm with key size <256 bits

Any ESP Transform with key size <256 bits

PRF_AES128_XCBC and PRF_AES128_CBC; even though they are defined to be able to use an arbitrary key size, they convert it into a 128 bit key internally

8. References

8.1. Normative References

[RFC2104] Krawczyk, H., Bellare, M., and R. Canetti, "HMAC: Keyed-Hashing for Message Authentication", RFC 2104, DOI 10.17487/RFC2104, February 1997, <http://www.rfc-editor.org/info/rfc2104>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <http://www.rfc-editor.org/info/rfc7296>.

8.2. Informational References

[RFC6023] Nir, Y., Tschofenig, H., Deng, H., and R. Singh, "A Childless Initiation of the Internet Key Exchange Version 2 (IKEv2) Security Association (SA)", RFC 6023, DOI 10.17487/RFC6023, October 2010, <http://www.rfc-editor.org/info/rfc6023>.

[RFC6030] Hoyer, P., Pei, M., and S. Machani, "Portable Symmetric Key Container (PSKC)", RFC 6030, DOI 10.17487/RFC6030, October 2010, <http://www.rfc-editor.org/info/rfc6030>.

[SPDP] McGrew, D., "A Secure Peer Discovery Protocol (SPDP)", 2001, <http://www.mindspring.com/~dmcgrew/spdp.txt>.

Appendix A. Discussion and Rationale

The idea behind this is that while a Quantum Computer can easily reconstruct the shared secret of an (EC)DH exchange, they cannot as easily recover a secret from a symmetric exchange this makes the SK_d, and hence the IPsec KEYMAT and any child SA's SKEYSEED, depend on both the symmetric PPK, and also the Diffie-Hellman exchange. If we assume that the attacker knows everything except the PPK during the key exchange, and there are 2**n plausible PPK's, then a Quantum Computer (using Grover's algorithm) would take O(2**(n/2)) time to recover the PPK. So, even if the (EC)DH can be trivially solved, the attacker still can't recover any key material (except for the SK_ei, SK_er, SK_ai, SK_ar values for the initial IKE exchange) unless they can find the PPK, and that's too difficult if the PPK has enough entropy (for example, 256 bits). Note that we do allow an attacker with a Quantum Computer to rederive the keying material for the initial IKE SA; this was a compromise to allow the responder to select the correct PPK quickly.

Another goal of this protocol is to minimize the number of changes within the IKEv2 protocol, and in particular, within the cryptography of IKEv2. By limiting our changes to notifications, and translating the nonces, it is hoped that this would be implementable, even on systems that perform much of the IKEv2 processing is in hardware.

A third goal was to be friendly to incremental deployment in operational networks, for which we might not want to have a global shared key, and also if we're rolling this out incrementally. This

is why we specifically try to allow the PPK to be dependent on the peer, and why we allow the PPK to be configured as optional.

A fourth goal was to avoid violating any of the security goals of IKEv2.

Appendix B. Acknowledgement

We would like to thank Tero Kivine, Valery Smyslov, Paul Wouters and the rest of the ipsecme working group for their feedback and suggestions for the scheme

Authors' Addresses

Scott Fluhrer
Cisco Systems

Email: sfluhrer@cisco.com

David McGrew
Cisco Systems

Email: mcgrew@cisco.com

Panos Kampanakis
Cisco Systems

Email: pkampana@cisco.com

IPSecME Working Group
Internet-Draft
Intended status: Standards Track
Expires: October 17, 2017

Y. Nir
Check Point

April 15, 2017

Using Edwards-curve Digital Signature Algorithm (EdDSA) in the Internet Key Exchange (IKEv2)
draft-ietf-ipsecme-eddsa-03

Abstract

This document describes the use of the Edwards-curve digital signature algorithm in the IKEv2 protocol.

Status of This Memo

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

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

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

This Internet-Draft will expire on October 17, 2017.

Copyright Notice

Table of Contents

1. Introduction

The Internet Key Exchange protocol [RFC7296] can use arbitrary signature algorithms as described in [RFC7427]. The latter RFC defines the SIGNATURE_HASH_ALGORITHMS notification where each side of the IKE negotiation lists its supported hash algorithms. This assumes that all signature schemes involve a hashing phase followed by a signature phase. This made sense because most signature algorithms either cannot sign messages bigger than their key or truncate messages bigger than their key.

EdDSA ([RFC8032]) defines signature methods that do not require pre-hashing of the message. Unlike other methods, these accept arbitrary-sized messages, so no pre-hashing is required. These methods are called Ed25519 and Ed448, which respectively use the Edwards 25519 and the Edwards 448 ("Goldilocks") curves. Although that document also defines pre-hashed versions of these algorithm, those versions are not recommended for protocols where the entire to-be-signed message is available at once. See section 8.5 or RFC 8032 for that recommendation.

EdDSA defines the binary format of the signatures that should be used in the "Signature Value" field of the Authentication Data Format in section 3. The CURDLE PKIX document ([I.D-curdle-pkix]) defines the object identifiers (OIDs) for these signature methods. For convenience, these OIDs are repeated in Appendix A.

In order to signal within IKE that no hashing needs to be done, we define a new value has in the SIGNATURE_HASH_ALGORITHMS notification, one that indicates that no hashing is performed.

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. The "Identity" Hash Identifier

This document defines a new value called "Identity" (value is 5) in the hash algorithm registry for use in the SIGNATURE_HASH_ALGORITHMS notification. Inserting this new value into the notification indicates that the receiver supports at least one signature algorithm that accepts arbitrary-sized messages such as Ed25519 and Ed448.

Ed25519 and Ed448 are only defined with the Identity hash, and MUST NOT be sent to a receiver that has not indicated support for the "Identity" hash.

The pre-hashed versions of Ed25519 and Ed448 (Ed25519ph and Ed448ph respectively) SHOULD NOT be used in IKE.

3. Security Considerations

The new "Identity" value is needed only for signature algorithms that accept an arbitrary-sized input. It MUST NOT be used if none of the supported and configured algorithms have this property. On the other hand there is no good reason to pre-hash the inputs where the signature algorithm has that property. For this reason implementations MUST have the "Identity" value in the SIGNATURE_HASH_ALGORITHMS notification when EdDSA is supported and configured. Implementations SHOULD NOT have other hash algorithms in the notification if all supported and configured signature algorithms have this property.

4. IANA Considerations

IANA has assigned the value 5 for the algorithm with the name "Identity" in the "IKEv2 Hash Algorithms" registry with this draft as reference.

Upon publication of this document IANA is requested to update the entry with this document as reference.

5. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <http://www.rfc-editor.org/info/rfc7296>.

[RFC7427] Kivinen, T. and J. Snyder, "Signature Authentication in the Internet Key Exchange Version 2 (IKEv2)", RFC 7427, DOI 10.17487/RFC7427, January 2015, <http://www.rfc-editor.org/info/rfc7427>.

[RFC8032] Josefsson, S. and I. Liusvaara, "Edwards-Curve Digital Signature Algorithm (EdDSA)", RFC 8032, DOI 10.17487/RFC8032, January 2017, <http://www.rfc-editor.org/info/rfc8032>.

[I.D-curdle-pkix] Josefsson, S. and J. Schaad, "Algorithm Identifiers for Ed25519, Ed25519ph, Ed448, Ed448ph, X25519 and X448 for use in the Internet X.509 Public Key Infrastructure", March 2017, <https://tools.ietf.org/html/draft-ietf-curdle-pkix-04>.

Appendix A. ASN.1 Objects

The normative reference for the ASN.1 objects for Ed25519 and Ed448 is in [I.D-curdle-pkix]. They are repeated below for convenience.

A.1. ASN.1 Object for Ed25519

```
id-Ed25519 OBJECT IDENTIFIER ::= { 1.3.101.112 }
```

Parameters are absent. Length is 7 bytes.

Binary encoding: 3005 0603 2B65 70

A.2. ASN.1 Object for Ed448

```
id-Ed448 OBJECT IDENTIFIER ::= { 1.3.101.113 }
```

Parameters are absent. Length is 7 bytes.

Binary encoding: 3005 0603 2B65 71

Author's Address

Yoav Nir
Check Point Software Technologies Ltd.
5 Hasolelim st.
Tel Aviv 6789735
Israel

EMail: ynir.ietf@gmail.com

Network Working Group
Internet-Draft
Obsoletes: 4307 (if approved)
Updates: 7296 (if approved)
Intended status: Standards Track
Expires: September 30, 2017

Y. Nir
Check Point
T. Kivinen
INSIDE Secure
P. Wouters
Red Hat
D. Migault
Ericsson

March 29, 2017

Algorithm Implementation Requirements and Usage Guidance for IKEv2
draft-ietf-ipsecme-rfc4307bis-18

Abstract

The IPsec series of protocols makes use of various cryptographic algorithms in order to provide security services. The Internet Key Exchange (IKE) protocol is used to negotiate the IPsec Security Association (IPsec SA) parameters, such as which algorithms should be used. To ensure interoperability between different implementations, it is necessary to specify a set of algorithm implementation requirements and usage guidance to ensure that there is at least one algorithm that all implementations support. This document updates RFC 7296 and obsoletes RFC 4307 in defining the current algorithm implementation requirements and usage guidance for IKEv2, and does minor cleaning up of the IKEv2 IANA registry. This document does not update the algorithms used for packet encryption using IPsec Encapsulated Security Payload (ESP).

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 30, 2017.

Table of Contents

1. Introduction

The Internet Key Exchange (IKE) protocol [RFC7296] is used to negotiate the parameters of the IPsec SA, such as the encryption and authentication algorithms and the keys for the protected communications between the two endpoints. The IKE protocol itself is

also protected by cryptographic algorithms which are negotiated between the two endpoints using IKE. Different implementations of IKE may negotiate different algorithms based on their individual local policy. To ensure interoperability, a set of "mandatory-to-implement" IKE cryptographic algorithms is defined.

This document describes the parameters of the IKE protocol and updates the IKEv2 specification. It changes the mandatory to implement authentication algorithms of Section 4 of [RFC7296] by saying RSA key lengths of less than 2048 SHOULD NOT be used. It does not describe the cryptographic parameters of the AH or ESP protocols.

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].

When used in the tables in this document, these terms indicate that the listed algorithm MUST, MUST NOT, SHOULD, SHOULD NOT or MAY be implemented as part of an IKEv2 implementation. Additional terms used in this document are:

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, an algorithm marked as SHOULD- may be deprecated to a MAY in a future version of this document.
MUST-	This term means the same as MUST. However, it is expected at some point that this algorithm will no longer be a MUST in a future document. Although its status will be determined at a later time, it is reasonable to expect that if a future revision of a document alters the status of a MUST- algorithm, it will remain at least a SHOULD or a SHOULD- level.
IoT	stands for Internet of Things.

1.2. Updating Algorithm Implementation Requirements and Usage Guidance

The field of cryptography evolves continuously. New stronger algorithms appear and existing algorithms are found to be less secure then originally thought. Therefore, algorithm implementation requirements and usage guidance need to be updated from time to time to reflect the new realityI The choices for algorithms must be conservative to minimize the risk of algorithm compromise. Algorithms need to be suitable for a wide variety of CPU

architectures and device deployments ranging from high end bulk encryption devices to small low-power IoT devices.

The algorithm implementation requirements and usage guidance may need to change over time to adapt to the changing world. For this reason, the selection of mandatory-to-implement algorithms was removed from the main IKEv2 specification and placed in this separate document.

1.3. Updating Algorithm Requirement Levels

The mandatory-to-implement algorithm of tomorrow should already be available in most implementations of IKE by the time it is made mandatory. This document attempts to identify and introduce those algorithms for future mandatory-to-implement status. There is no guarantee that the algorithms in use today may become mandatory in the future. Published algorithms are continuously subjected to cryptographic attack and may become too weak or could become completely broken before this document is updated.

This document only provides recommendations for the mandatory-to-implement algorithms or algorithms too weak that are recommended not to be implemented. As a result, any algorithm listed at the IKEv2 IANA registry not mentioned in this document MAY be implemented. For clarification and consistency with [RFC4307] an algorithm will be denoted here as MAY only when it has been downgraded.

Although this document updates the algorithms to keep the IKEv2 communication secure over time, it also aims at providing recommendations so that IKEv2 implementations remain interoperable. IKEv2 interoperability is addressed by an incremental introduction or deprecation of algorithms. In addition, this document also considers the new use cases for IKEv2 deployment, such as Internet of Things (IoT).

It is expected that deprecation of an algorithm is performed gradually. This provides time for various implementations to update their implemented algorithms while remaining interoperable. Unless there are strong security reasons, an algorithm is expected to be downgraded from MUST to MUST- or SHOULD, instead of MUST NOT. Similarly, an algorithm that has not been mentioned as mandatory-to-implement is expected to be introduced with a SHOULD instead of a MUST.

The current trend toward Internet of Things and its adoption of IKEv2 requires this specific use case to be taken into account as well. IoT devices are resource constrained devices and their choice of algorithms are motivated by minimizing the footprint of the code, the computation effort and the size of the messages to send. This

document indicates "(IoT)" when a specified algorithm is specifically listed for IoT devices. Requirement levels that are marked as "IoT" apply to IoT devices and to server-side implementations that might presumably need to interoperate with them, including any general-purpose VPN gateways.

1.4. Document Audience

The recommendations of this document mostly target IKEv2 implementers who need to create implementations that meet both high security expectations as well as high interoperability between various vendors and with different versions. Interoperability requires a smooth move to more secure cipher suites. This may differ from a user point of view that may deploy and configure IKEv2 with only the safest cipher suite.

This document does not give any recommendations for the use of algorithms, it only gives implementation recommendations regarding implementations. The use of algorithms by users is dictated by the security policy requirements for that specific user, and are outside the scope of this document.

IKEv1 is out of scope of this document. IKEv1 is deprecated and the recommendations of this document must not be considered for IKEv1, as most IKEv1 implementations have been "frozen" and will not be able to update the list of mandatory-to-implement algorithms.

2. Algorithm Selection

2.1. Type 1 - IKEv2 Encryption Algorithm Transforms

The algorithms in the below table are negotiated in the SA payload and used for the Encrypted Payload. References to the specification defining these algorithms and the ones in the following subsections are in the IANA registry [IKEV2-IANA]. Some of these algorithms are Authenticated Encryption with Associated Data (AEAD - [RFC5282]). Algorithms that are not AEAD MUST be used in conjunction with one of the integrity algorithms in Section 2.3.

Name	Status	AEAD?	Comment
ENCR_AES_CBC	MUST	No	(1)
ENCR_CHACHA20_POLY1305	SHOULD	Yes	
ENCR_AES_GCM_16	SHOULD	Yes	(1)
ENCR_AES_CCM_8	SHOULD	Yes	(IoT)
ENCR_3DES	MAY	No	
ENCR_DES	MUST NOT	No	

(1) - This requirement level is for 128-bit and 256-bit keys. 192-bit keys remain at MAY level. (IoT) - This requirement is for interoperability with IoT. Only 128-bit keys are at SHOULD level. 192-bit and 256-bit remain at the MAY level.

ENCR_AES_CBC is raised from SHOULD+ for 128-bit keys and MAY for 256-bit keys in [RFC4307] to MUST. 192-bit keys remain at the MAY level. ENCR_AES_CBC is the only shared mandatory-to-implement algorithm with RFC4307 and as a result it is necessary for interoperability with IKEv2 implementation compatible with RFC4307.

ENCR_CHACHA20_POLY1305 was not ready to be considered at the time of RFC4307. It has been recommended by the Crypto Forum Research Group (CFRG) of the IRTF as an alternative to AES-CBC and AES-GCM. It is also being standardized for IPsec for the same reasons. At the time of writing, there were not enough IKEv2 implementations supporting ENCR_CHACHA20_POLY1305 to be able to introduce it at the SHOULD+ level.

ENCR_AES_GCM_16 was not considered in RFC4307. At the time RFC4307 was written, AES-GCM was not defined in an IETF document. AES-GCM was defined for ESP in [RFC4106] and later for IKEv2 in [RFC5282]. The main motivation for adopting AES-GCM for ESP is encryption performance compared to AES-CBC. This resulted in AES-GCM being widely implemented for ESP. As the computation load of IKEv2 is relatively small compared to ESP, many IKEv2 implementations have not implemented AES-GCM. For this reason, AES-GCM is not promoted to a greater status than SHOULD. The reason for promotion from MAY to SHOULD is to promote the slightly more secure AEAD method over the traditional encrypt+auth method. Its status is expected to be raised once widely implemented. As the advantage of the shorter (and weaker) ICVs is minimal, the 8 and 12 octet ICV’s remain at the MAY level.

ENCR_AES_CCM_8 was not considered in RFC4307. This document considers it as SHOULD be implemented in order to be able to interact with Internet of Things devices. As this case is not a general use

case for non-IoT VPNs, its status is expected to remain as SHOULD. The 8 octet size of the ICV is expected to be sufficient for most use cases of IKEv2, as far less packets are exchanged in those cases, and IoT devices want to make packets as small as possible. The SHOULD level is for 128-bit keys, 256-bit keys remains at MAY level.

ENCR_3DES has been downgraded from RFC4307 MUST- to MAY. All IKEv2 implementations already implement ENCR_AES_CBC, so there is no need to keep support for the much slower ENCR_3DES. In addition, ENCR_CHACHA20_POLY1305 provides a more modern alternative to AES.

ENCR_DES can be brute-forced using off-the-shelf hardware. It provides no meaningful security whatsoever and therefore MUST NOT be implemented.

2.2. Type 2 - IKEv2 Pseudo-random Function Transforms

Transform Type 2 algorithms are pseudo-random functions used to generate pseudo-random values when needed.

Name	Status	Comment
PRF_HMAC_SHA2_256	MUST	
PRF_HMAC_SHA2_512	SHOULD+	
PRF_HMAC_SHA1	MUST-	
PRF_AES128_XCBC	SHOULD	(IoT)
PRF_HMAC_MD5	MUST NOT	

(IoT) - This requirement is for interoperability with IoT

As no SHA2 based transforms were referenced in RFC4307, PRF_HMAC_SHA2_256 was not mentioned in RFC4307. PRF_HMAC_SHA2_256 MUST be implemented in order to replace SHA1 and PRF_HMAC_SHA1.

PRF_HMAC_SHA2_512 SHOULD be implemented as a future replacement for PRF_HMAC_SHA2_256 or when stronger security is required. PRF_HMAC_SHA2_512 is preferred over PRF_HMAC_SHA2_384, as the additional overhead of PRF_HMAC_SHA2_512 is negligible.

PRF_HMAC_SHA1 has been downgraded from MUST in RFC4307 to MUST- as cryptographic attacks against SHA1 are increasing, resulting in an industry-wide trend to deprecate its usage

PRF_AES128_XCBC is only recommended in the scope of IoT, as Internet of Things deployments tend to prefer AES based pseudo-random functions in order to avoid implementing SHA2. For the non-IoT VPN

deployment it has been downgraded from SHOULD in RFC4307 to MAY as it has not seen wide adoption.

PRF_HMAC_MD5 has been downgraded from MAY in RFC4307 to MUST NOT. Cryptographic attacks against MD5, such as collision attacks mentioned in [TRANSCRIPTION], are resulting in an industry-wide trend to deprecate and remove MD5 (and thus HMAC-MD5) from cryptographic libraries.

2.3. Type 3 - IKEv2 Integrity Algorithm Transforms

The algorithms in the below table are negotiated in the SA payload and used for the Encrypted Payload. References to the specification defining these algorithms are in the IANA registry. When an AEAD algorithm (see Section 2.1) is proposed, this algorithm transform type is not in use.

Name	Status	Comment
AUTH_HMAC_SHA2_256_128	MUST	
AUTH_HMAC_SHA2_512_256	SHOULD	
AUTH_HMAC_SHA1_96	MUST-	
AUTH_AES_XCBC_96	SHOULD	(IoT)
AUTH_HMAC_MD5_96	MUST NOT	
AUTH_DES_MAC	MUST NOT	
AUTH_KPDK_MD5	MUST NOT	

(IoT) - This requirement is for interoperability with IoT

AUTH_HMAC_SHA2_256_128 was not mentioned in RFC4307, as no SHA2 based transforms were mentioned. AUTH_HMAC_SHA2_256_128 MUST be implemented in order to replace AUTH_HMAC_SHA1_96.

AUTH_HMAC_SHA2_512_256 SHOULD be implemented as a future replacement of AUTH_HMAC_SHA2_256_128 or when stronger security is required. This value has been preferred over AUTH_HMAC_SHA2_384, as the additional overhead of AUTH_HMAC_SHA2_512 is negligible.

AUTH_HMAC_SHA1_96 has been downgraded from MUST in RFC4307 to MUST- as cryptographic attacks against SHA1 are increasing, resulting in an industry-wide trend to deprecate its usage

AUTH_AES_XCBC_96 is only recommended in the scope of IoT, as Internet of Things deployments tend to prefer AES based pseudo-random functions in order to avoid implementing SHA2. For the non-IoT VPN

deployment, it has been downgraded from SHOULD in RFC4307 to MAY as it has not been widely adopted.

AUTH_DES_MAC, AUTH_HMAC_MD5_96, and AUTH_KPDK_MD5 were not mentioned in RFC4307 so their default statuses were MAY. They have been downgraded to MUST NOT. There is an industry-wide trend to deprecate DES and MD5. MD5 support is being removed from cryptographic libraries in general because its non-HMAC use is known to be subject to collision attacks, for example as mentioned in [TRANSCRIPTION].

2.4. Type 4 - IKEv2 Diffie-Hellman Group Transforms

There are several Modular Exponential (MODP) groups and several Elliptic Curve groups (ECC) that are defined for use in IKEv2. These groups are defined in both the [RFC7296] base document and in extensions documents and are identified by group number. Note that it is critical to enforce a secure Diffie-Hellman exchange as this exchange provides keys for the session. If an attacker can retrieve one of the private numbers (a or b) and the complementary public value (g**b or g**a), then the attacker can compute the secret and the keys used and decrypt the exchange and IPsec SA created inside the IKEv2 SA. Such an attack can be performed off-line on a previously recorded communication, years after the communication happened. This differs from attacks that need to be executed during the authentication which must be performed online and in near real-time.

Number	Description	Status
14	2048-bit MODP Group	MUST
19	256-bit random ECP group	SHOULD
5	1536-bit MODP Group	SHOULD NOT
2	1024-bit MODP Group	SHOULD NOT
1	768-bit MODP Group	MUST NOT
22	1024-bit MODP Group with 160-bit Prime Order Subgroup	MUST NOT
23	2048-bit MODP Group with 224-bit Prime Order Subgroup	SHOULD NOT
24	2048-bit MODP Group with 256-bit Prime Order Subgroup	SHOULD NOT

Group 14 or 2048-bit MODP Group is raised from SHOULD+ in RFC4307 to MUST as a replacement for 1024-bit MODP Group. Group 14 is widely implemented and considered secure.

Group 19 or 256-bit random ECP group was not specified in RFC4307, as this group was not defined at that time. Group 19 is widely implemented and considered secure and therefore has been promoted to the SHOULD level.

Group 5 or 1536-bit MODP Group has been downgraded from MAY in RFC4307 to SHOULD NOT. It was specified earlier, but is now considered to be vulnerable to being broken within the next few years by a nation state level attack, so its security margin is considered too narrow.

Group 2 or 1024-bit MODP Group has been downgraded from MUST- in RFC4307 to SHOULD NOT. It is known to be weak against sufficiently funded attackers using commercially available mass-computing resources, so its security margin is considered too narrow. It is expected in the near future to be downgraded to MUST NOT.

Group 1 or 768-bit MODP Group was not mentioned in RFC4307 and so its status was MAY. It can be broken within hours using cheap of-the-shelves hardware. It provides no security whatsoever. It has therefore been downgraded to MUST NOT.

Group 22, 23 and 24 are MODP Groups with Prime Order Subgroups that are not safe-primes. The seeds for these groups have not been publicly released, resulting in reduced trust in these groups. These groups were proposed as alternatives for group 2 and 14 but never saw wide deployment. It has been shown that Group 22 with 1024-bit MODP is too weak and academia have the resources to generate malicious values at this size. This has resulted in Group 22 to be demoted to MUST NOT. Group 23 and 24 have been demoted to SHOULD NOT and are expected to be further downgraded in the near future to MUST NOT. Since Group 23 and 24 have small subgroups, the checks specified in "Additional Diffie-Hellman Test for the IKEv2" [RFC6989] section 2.2 first bullet point MUST be done when these groups are used.

2.5. Summary of Changes from RFC 4307

The following table summarizes the changes from RFC 4307.

RFC EDITOR: PLEASE REMOVE THIS PARAGRAPH AND REPLACE XXXX IN THE TABLE BELOW WITH THE NUMBER OF THIS RFC

Algorithm	RFC 4307	RFC XXXX
ENCR_3DES	MUST-	MAY
ENCR_NULL	MUST NOT[errata]	MUST NOT
ENCR_AES_CBC	SHOULD+	MUST
ENCR_AES_CTR	SHOULD	(*)
PRF_HMAC_MD5	MAY	MUST NOT
PRF_HMAC_SHA1	MUST	MUST-
PRF_AES128_XCBC	SHOULD+	SHOULD
AUTH_HMAC_MD5_96	MAY	MUST NOT
AUTH_HMAC_SHA1_96	MUST	MUST-
AUTH_AES_XCBC_96	SHOULD+	SHOULD
Group 2 (1024-bit)	MUST-	SHOULD NOT
Group 14 (2048-bit)	SHOULD+	MUST

(*) This algorithm is not mentioned in the above sections, so it defaults to MAY.

3. IKEv2 Authentication

IKEv2 authentication may involve a signatures verification. Signatures may be used to validate a certificate or to check the signature of the AUTH value. Cryptographic recommendations regarding certificate validation are out of scope of this document. What is mandatory to implement is provided by the PKIX Community. This document is mostly concerned with signature verification and generation for the authentication.

3.1. IKEv2 Authentication Method

Number	Description	Status
1	RSA Digital Signature	MUST
2	Shared Key Message Integrity Code	MUST
3	DSS Digital Signature	SHOULD NOT
9	ECDSA with SHA-256 on the P-256 curve	SHOULD
10	ECDSA with SHA-384 on the P-384 curve	SHOULD
11	ECDSA with SHA-512 on the P-521 curve	SHOULD
14	Digital Signature	SHOULD

RSA Digital Signature is widely deployed and therefore kept for interoperability. It is expected to be downgraded in the future as its signatures are based on the older RSASSA-PKCS1-v1.5 which is no longer recommended. RSA authentication, as well as other specific

Authentication Methods, are expected to be replaced with the generic Digital Signature method of [RFC7427].

Shared Key Message Integrity Code is widely deployed and mandatory to implement in the IKEv2 in the RFC7296. The status remains MUST.

ECDSA based Authentication Methods are also expected to be downgraded as these do not provide hash function agility. Instead, ECDSA (like RSA) is expected to be performed using the generic Digital Signature method. It's status is SHOULD.

DSS Digital Signature is bound to SHA-1 and has the same level of security as 1024-bit RSA. It is currently at SHOULD NOT and is expected to be downgraded to MUST NOT in the future.

Digital Signature [RFC7427] is expected to be promoted as it provides hash function, signature format and algorithm agility.Its current status is SHOULD.

3.1.1. Recommendations for RSA key length

Description	Status
RSA with key length 2048	MUST
RSA with key length 3072 and 4096	SHOULD
RSA with key length between 2049 and 4095	MAY
RSA with key length smaller than 2048	SHOULD NOT

The IKEv2 RFC7296 mandates support for the RSA keys of size 1024 or 2048 bits, but key sizes less than 2048 are updated to SHOULD NOT as there is industry-wide trend to deprecate key lengths less than 2048 bits. Since these signatures only have value in real-time, and need no future protection, smaller keys were kept at SHOULD NOT instead of MUST NOT.

3.2. Digital Signature Recommendations

When a Digital Signature authentication method is implemented, the following recommendations are applied for hash functions:

Number	Description	Status	Comment
1	SHA1	MUST NOT	
2	SHA2-256	MUST	
3	SHA2-384	MAY	
4	SHA2-512	SHOULD	

When the Digital Signature authentication method is used with RSA signature algorithm, RSASSA-PSS MUST be supported and RSASSA-PKCS1-v1.5 MAY be supported.

The following table lists recommendations for authentication methods in RFC7427 [RFC7427] notation. These recommendations are applied only if Digital Signature authentication method is implemented.

Description	Status	Comment
RSASSA-PSS with SHA-256	MUST	
ecdsa-with-sha256	SHOULD	
sha1WithRSAEncryption	MUST NOT	
dsa-with-sha1	MUST NOT	
ecdsa-with-sha1	MUST NOT	
RSASSA-PSS with Empty Parameters	MUST NOT	(*)
RSASSA-PSS with Default Parameters	MUST NOT	(*)

(*) Empty or Default parameters means it is using SHA1, which is at level MUST NOT.

4. Algorithms for Internet of Things

Some algorithms in this document are marked for use with the Internet of Things (IoT). There are several reasons why IoT devices prefer a different set of algorithms from regular IKEv2 clients. IoT devices are usually very constrained, meaning the memory size and CPU power is so limited, that these clients only have resources to implement and run one set of algorithms. For example, instead of implementing AES and SHA, these devices typically use AES_XCBC as integrity algorithm so SHA does not need to be implemented.

For example, IEEE Std 802.15.4 [IEEE-802-15-4] devices have a mandatory to implement link level security using AES-CCM with 128 bit keys. The IEEE Recommended Practice for Transport of Key Management Protocol (KMP) Datagrams [IEEE-802-15-9] already provide a way to use

Minimal IKEv2 [RFC7815] over 802.15.4 to provide link keys for the 802.15.4 layer.

These devices might want to use AES-CCM as their IKEv2 algorithm, so they can reuse the hardware implementing it. They cannot use the AES-CBC algorithm, as the hardware quite often do not include support for AES decryption needed to support the CBC mode. So despite the AES-CCM algorithm requiring AEAD [RFC5282] support, the benefit of reusing the crypto hardware makes AES-CCM the preferred algorithm.

Another important aspect of IoT devices is that their transfer rates are usually quite low (in order of tens of kbits/s), and each bit they transmit has an energy consumption cost associated with it and shortens their battery life. Therefore, shorter packets are preferred. This is the reason for recommending the 8 octet ICV over the 16 octet ICV.

Because different IoT devices will have different constraints, this document cannot specify the one mandatory profile for IoT. Instead, this document points out commonly used algorithms with IoT devices.

5. Security Considerations

The security of cryptographic-based systems 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 of the protocol used by the system to ensure that there are no non-cryptographic ways to bypass the security of the overall system.

The Diffie-Hellman Group parameter is the most important one to choose conservatively. Any party capturing all IKE and ESP traffic that (even years later) can break the selected DH group in IKE, can gain access to the symmetric keys used to encrypt all the ESP traffic. Therefore, these groups must be chosen very conservatively. However, specifying an extremely large DH group also puts a considerable load on the device, especially when this is a large VPN gateway or an IoT constrained device.

This document concerns itself with the selection of cryptographic algorithms for the use of IKEv2, 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 isn’t necessarily forever and it is expected that new revisions of this document will be issued from time to time to reflect the current best practice in this area.

6. IANA Considerations

This document renames some of the names in the "Transform Type 1 - Encryption Algorithm Transform IDs" registry of the "Internet Key Exchange Version 2 (IKEv2) Parameters". All the other names have ENCR_ prefix except 3, and all other entries use names in format of uppercase words separated with underscores except 6. This document changes those names to match others.

This document requests IANA to rename following entries for the AES-GCM cipher [RFC4106] and the Camellia cipher [RFC5529]:

Old name	New name
AES-GCM with a 8 octet ICV	ENCR_AES_GCM_8
AES-GCM with a 12 octet ICV	ENCR_AES_GCM_12
AES-GCM with a 16 octet ICV	ENCR_AES_GCM_16
ENCR_CAMELLIA_CCM with an 8-octet ICV	ENCR_CAMELLIA_CCM_8
ENCR_CAMELLIA_CCM with a 12-octet ICV	ENCR_CAMELLIA_CCM_12
ENCR_CAMELLIA_CCM with a 16-octet ICV	ENCR_CAMELLIA_CCM_16

In addition to add this RFC as reference to both ESP Reference and IKEv2 Reference columns for ENCR_AES_GCM entries, keeping the current references there also, and also add this RFC as reference to the ESP Reference column for ENCR_CAMELLIA_CCM entries, keeping the current reference there also.

The final registry entries should be:

Number	Name	ESP Reference	IKEv2 Reference
...			
18	ENCR_AES_GCM_8	[RFC4106][RFCXXXX]	[RFC5282][RFCXXXX]
19	ENCR_AES_GCM_12	[RFC4106][RFCXXXX]	[RFC5282][RFCXXXX]
20	ENCR_AES_GCM_16	[RFC4106][RFCXXXX]	[RFC5282][RFCXXXX]
...			
25	ENCR_CAMELLIA_CCM_8	[RFC5529][RFCXXXX]	-
26	ENCR_CAMELLIA_CCM_12	[RFC5529][RFCXXXX]	-
27	ENCR_CAMELLIA_CCM_16	[RFC5529][RFCXXXX]	-

7. Acknowledgements

The first version of this document was RFC 4307 by Jeffrey I. Schiller of the Massachusetts Institute of Technology (MIT). Much of the original text has been copied verbatim.

We would like to thank Paul Hoffman, Yaron Sheffer, John Mattsson, Tommy Pauly, Eric Rescorla and Pete Resnick for their valuable feedback and reviews.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC4106] Viega, J. and D. McGrew, "The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)", RFC 4106, DOI 10.17487/RFC4106, June 2005, <http://www.rfc-editor.org/info/rfc4106>.

[RFC4307] Schiller, J., "Cryptographic Algorithms for Use in the Internet Key Exchange Version 2 (IKEv2)", RFC 4307, DOI 10.17487/RFC4307, December 2005, <http://www.rfc-editor.org/info/rfc4307>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <http://www.rfc-editor.org/info/rfc7296>.

[RFC5282] Black, D. and D. McGrew, "Using Authenticated Encryption Algorithms with the Encrypted Payload of the Internet Key Exchange version 2 (IKEv2) Protocol", RFC 5282, DOI 10.17487/RFC5282, August 2008, <http://www.rfc-editor.org/info/rfc5282>.

8.2. Informative References

[RFC7427] Kivinen, T. and J. Snyder, "Signature Authentication in the Internet Key Exchange Version 2 (IKEv2)", RFC 7427, DOI 10.17487/RFC7427, January 2015, <http://www.rfc-editor.org/info/rfc7427>.

[RFC6989] Sheffer, Y. and S. Fluhrer, "Additional Diffie-Hellman Tests for the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 6989, DOI 10.17487/RFC6989, July 2013, <http://www.rfc-editor.org/info/rfc6989>.

[RFC7815] Kivinen, T., "Minimal Internet Key Exchange Version 2 (IKEv2) Initiator Implementation", RFC 7815, DOI 10.17487/RFC7815, March 2016, <http://www.rfc-editor.org/info/rfc7815>.

[RFC5529] Kato, A., Kanda, M., and S. Kanno, "Modes of Operation for Camellia for Use with IPsec", RFC 5529, DOI 10.17487/RFC5529, April 2009, <http://www.rfc-editor.org/info/rfc5529>.

[IKEV2-IANA] "Internet Key Exchange Version 2 (IKEv2) Parameters", <http://www.iana.org/assignments/ikev2-parameters>.

[TRANSCRIPTION] Bhargavan, K. and G. Leurent, "Transcript Collision Attacks: Breaking Authentication in TLS, IKE, and SSH", NDSS , feb 2016.

[IEEE-802-15-4] "IEEE Standard for Low-Rate Wireless Personal Area Networks (WPANs)", IEEE Standard 802.15.4, 2015.

[IEEE-802-15-9] "IEEE Recommended Practice for Transport of Key Management Protocol (KMP) Datagrams", IEEE Standard 802.15.9, 2016.

Authors' Addresses

Yoav Nir
Check Point Software Technologies Ltd.
5 Hasolelim st.
Tel Aviv 6789735
Israel

EMail: ynir.ietf@gmail.com

Tero Kivinen
INSIDE Secure
Eerikinkatu 28
HELSINKI FI-00180
FI

EMail: kivinen@iki.fi

Paul Wouters
Red Hat

EMail: pwouters@redhat.com

Daniel Migault
Ericsson
8400 boulevard Decarie
Montreal, QC H4P 2N2
Canada

Phone: +1 514-452-2160
EMail: daniel.migault@ericsson.com

Network Working Group
Internet-Draft
Obsoletes: 7321 (if approved)
Intended status: Standards Track
Expires: December 21, 2017

P. Wouters
Red Hat
D. Migault
J. Mattsson
Ericsson
Y. Nir
Check Point
T. Kivinen
INSIDE Secure
June 19, 2017

Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)
draft-ietf-ipsecme-rfc7321bis-06

Abstract

This document updates the Cryptographic Algorithm Implementation Requirements for ESP and AH. The goal of these document is to enable ESP and AH to benefit from cryptography that is up to date while making IPsec interoperable.

This document obsoletes RFC 7321.

Status of This Memo

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

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

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

This Internet-Draft will expire on December 21, 2017.

Copyright Notice

Table of Contents

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].

This document provides guidance and recommendations so that ESP and AH can be used with a cryptographic algorithms that are up to date. The challenge of such document is to make sure that over the time IPsec implementations can use secure and up-to-date cryptographic algorithms while keeping IPsec interoperable.

1.1. Updating Algorithm Implementation Requirements and Usage Guidance

The field of cryptography evolves continuously. New stronger algorithms appear and existing algorithms are found to be less secure than originally thought. Therefore, algorithm implementation requirements and usage guidance need to be updated from time to time to reflect the new reality. The choices for algorithms must be conservative to minimize the risk of algorithm compromise. Algorithms need to be suitable for a wide variety of CPU architectures and device deployments ranging from high end bulk encryption devices to small low-power IoT devices.

The algorithm implementation requirements and usage guidance may need to change over time to adapt to the changing world. For this reason, the selection of mandatory-to-implement algorithms was removed from the main IKEv2 specification and placed in a separate document.

1.2. Updating Algorithm Requirement Levels

The mandatory-to-implement algorithm of tomorrow should already be available in most implementations of AH/ESP by the time it is made mandatory. This document attempts to identify and introduce those algorithms for future mandatory-to-implement status. There is no guarantee that the algorithms in use today may become mandatory in the future. Published algorithms are continuously subjected to cryptographic attack and may become too weak or could become completely broken before this document is updated.

This document only provides recommendations for the mandatory-to-implement algorithms and algorithms too weak that are recommended not to be implemented. As a result, any algorithm listed at the IPsec IANA registry not mentioned in this document MAY be implemented. It is expected that this document will be updated over time and next versions will only mention algorithms which status has evolved. For clarification when an algorithm has been mentioned in [RFC7321], this document states explicitly the update of the status.

Although this document updates the algorithms to keep the AH/ESP communication secure over time, it also aims at providing recommendations so that AH/ESP implementations remain interoperable. AH/ESP interoperability is addressed by an incremental introduction or deprecation of algorithms. In addition, this document also considers the new use cases for AH/ESP deployment, such as Internet of Things (IoT).

It is expected that deprecation of an algorithm is performed gradually. This provides time for various implementations to update their implemented algorithms while remaining interoperable. Unless

there are strong security reasons, an algorithm is expected to be downgraded from MUST to MUST- or SHOULD, instead of MUST NOT. Similarly, an algorithm that has not been mentioned as mandatory-to-implement is expected to be introduced with a SHOULD instead of a MUST.

The current trend toward Internet of Things and its adoption of AH/ESP requires this specific use case to be taken into account as well. IoT devices are resource constrained devices and their choice of algorithms are motivated by minimizing the footprint of the code, the computation effort and the size of the messages to send. This document indicates "(IoT)" when a specified algorithm is specifically listed for IoT devices. Requirement levels that are marked as "IoT" apply to IoT devices and to server-side implementations that might presumably need to interoperate with them, including any general-purpose VPN gateways.

1.3. Document Audience

The recommendations of this document mostly target AH/ESP implementers as implementations need to meet both high security expectations as well as high interoperability between various vendors and with different versions. Interoperability requires a smooth move to more secure cipher suites. This may differ from a user point of view that may deploy and configure AH/ESP with only the safest cipher suite.

This document does not give any recommendations for the use of algorithms, it only gives implementation recommendations for implementations. The use of algorithms by users is dictated by the security policy requirements for that specific user, and are outside the scope of this document.

The algorithms considered here are listed by the IANA as part of the IKEv2 parameters. IKEv1 is out of scope of this document. IKEv1 is deprecated and the recommendations of this document must not be considered for IKEv1, nor IKEv1 parameters be considered by this document.

The IANA registry for Internet Key Exchange Version 2 (IKEv2) Parameters contains some entries that are not for use with ESP or AH. This document does not modify the status of those algorithms.

2. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and

"OPTIONAL" in this document are to be interpreted as described in [RFC2119].

We define some additional terms here:

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, an algorithm marked as SHOULD- may be deprecated to a MAY in a future version of this document.
MUST-	This term means the same as MUST. However, we expect at some point that this algorithm will no longer be a MUST in a future document. Although its status will be determined at a later time, it is reasonable to expect that if a future revision of a document alters the status of a MUST- algorithm, it will remain at least a SHOULD or a SHOULD- level.
IoT	stands for Internet of Things.

3. Manual Keying

Manual Keying SHOULD NOT be used as it is inherently dangerous. Without any secure keying protocol such a IKE, IPsec does not offer Perfect Forward Secrecy ("PFS") protection and there is no entity to ensure refreshing of session keys, tracking SPI uniqueness and ensuring nonces, IVs and counters are never re-used. This document was written for deploying ESP/AH using IKE ([RFC7296]) and assumes that keying happens using IKE version 2 or higher.

If Manual Keying is used regardless, Counter Mode algorithms such as ENCR_AES_CTR, ENCR_AES_CCM, ENCR_AES_GCM and ENCR_CHACHA20_POLY1305 MUST NOT be used as it is incompatible with a secure and persistent handling of the counter, as explained in the Security Considerations Section of [RFC3686]. This particularly applies to IoT devices that have no state across reboots. As of publication date of this document, ENCR_AES_CBC is the only Mandatory-To-Implement encryption algorithm suitable for Manual Keying.

4. Encryption must be Authenticated

Encryption without authentication is not effective and MUST NOT be used. IPsec offers three ways to provide both encryption and authentication:

- ESP with an AEAD cipher
- ESP with a non-AEAD cipher + authentication
- ESP with a non-AEAD cipher + AH with authentication

The fastest and most modern method is to use ESP with a combined mode cipher such as an AEAD cipher that handles encryption/decryption and authentication in a single step. In this case, the AEAD cipher is set as the encryption algorithm and the authentication algorithm is set to none. Examples of this are ENCR_AES_GCM_16 and ENCR_CHACHA20_POLY1305.

A more traditional approach is to use ESP with an encryption and an authentication algorithm. This approach is slower, as the data has to be processed twice, once for encryption/decryption and once for authentication. An example of this is ENCR_AES_CBC combined with AUTH_HMAC_SHA2_512_256.

The last method that can be used is ESP+AH. This method is NOT RECOMMENDED. It is the slowest method and also takes up more octets due to the double header of ESP+AH, resulting in a smaller effective MTU for the encrypted data. With this method, ESP is only used for confidentiality without an authentication algorithm and a second IPsec protocol of type AH is used for authentication. An example of this is ESP with ENCR_AES_CBC with AH with AUTH_HMAC_SHA2_512_256.

5. ESP Encryption Algorithms

Name	Status	AEAD	Comment
ENCR_DES_IV64	MUST NOT	No	UNSPECIFIED
ENCR_DES	MUST NOT	No	[RFC2405]
ENCR_3DES	SHOULD NOT	No	[RFC2451]
ENCR_BLOWFISH	MUST NOT	No	[RFC2451]
ENCR_3IDEA	MUST NOT	No	UNSPECIFIED
ENCR_DES_IV32	MUST NOT	No	UNSPECIFIED
ENCR_NULL	MUST	No	[RFC2410]
ENCR_AES_CBC	MUST	No	[RFC3602][1]
ENCR_AES_CCM_8	SHOULD(IoT)	Yes	[RFC4309]
ENCR_AES_GCM_16	MUST	Yes	[RFC4106][1]
ENCR_CHACHA20_POLY1305	SHOULD	Yes	[RFC7634]

[1] - This requirement level is for 128-bit and 256-bit keys. 192-bit keys remain at MAY level. (IoT) - This requirement is for interoperability with IoT. Only 128-bit keys are at the given level.

IPsec sessions may have very long life time, and carry multiple packets, so there is a need to move to 256-bit keys in the long term. For that purpose the requirement level for 128 bit keys and 256 bit keys are at MUST (when applicable). In that sense 256 bit keys status has been raised from MAY in RFC7321 to MUST.

IANA has allocated codes for cryptographic algorithms that have not been specified by the IETF. Such algorithms are noted as UNSPECIFIED. Usually, the use of theses algorithms is limited to specific cases, and the absence of specification makes interoperability difficult for IPsec communications. These algorithms were not been mentioned in [RFC7321] and this document clarify that such algorithms MUST NOT be implemented for IPsec communications.

Similarly IANA also allocated code points for algorithms that are not expected to be used to secure IPsec communications. Such algorithms are noted as Non IPsec. As a result, these algorithms MUST NOT be implemented.

Various older and not well tested and never widely implemented ciphers have been changed to MUST NOT.

ENCR_3DES status has been downgraded from MAY in RFC7321 to SHOULD NOT. ENCR_CHACHA20_POLY1305 is a more modern approach alternative for ENCR_3DES than ENCR_AES_CBC and so it expected to be favored to replace ENCR_3DES.

ENCR_BLOWFISH has been downgraded to MUST NOT as it has been deprecated for years by TWOFISH, which is not standarized for ESP and therefore not listed in this document. Some implementations support TWOFISH using a private range number.

ENCR_NULL status was set to MUST in [RFC7321] and remains a MUST to enable the use of ESP with only authentication which is preferred over AH due to NAT traversal. ENCR_NULL is expected to remain MUST by protocol requirements.

ENCR_AES_CBC status remains at MUST. ENCR_AES_CBC MUST be implemented in order to enable interoperability between implementations that followed RFC7321. However, there is a trend for the industry to move to AEAD encryption, and the overhead of ENCR_AES_CBC remains quite large so it is expected to be replaced by AEAD algorithms in the long term.

ENCR_AES_CCM_8 status was set to MAY in [RFC7321] and has been raised from MAY to SHOULD in order to interact with Internet of Things devices. As this case is not a general use case for VPNs, its status is expected to remain as SHOULD.

ENCR_AES_GCM_16 status has been updated from SHOULD+ to MUST in order to favor the use of authenticated encryption and AEAD algorithms. ENCR_AES_GCM_16 has been widely implemented for ESP due to its increased performance and key longevity compared to ENCR_AES_CBC.

ENCR_CHACHA20_POLY1305 was not ready to be considered at the time of RFC7321. It has been recommended by the CRFG and others as an alternative to AES-CBC and AES-GCM. It is also being standardized for ESP for the same reasons. At the time of writing, there are not enough ESP implementations of ENCR_CHACHA20_POLY1305 to be able to introduce it at the SHOULD+ level. Its status has been set to SHOULD and is expected to become MUST in the long term.

6. ESP and AH Authentication Algorithms

Authentication algorithm recommendations in this section are targeting two types of communications:

- Authenticated only communications without encryption, such as ESP with NULL encryption or AH communications.
- Communications that are encrypted with non-AEAD algorithm which MUST be combined with an authentication algorithm.

Name	Status	Comment
AUTH_NONE	MUST / MUST NOT	[RFC7296] AEAD
AUTH_HMAC_MD5_96	MUST NOT	[RFC2403][RFC7296]
AUTH_HMAC_SHA1_96	MUST-	[RFC2404][RFC7296]
AUTH_DES_MAC	MUST NOT	[UNSPECIFIED]
AUTH_KPDK_MD5	MUST NOT	[UNSPECIFIED]
AUTH_AES_XCBC_96	SHOULD	[RFC3566][RFC7296]
		(IoT)
AUTH_AES_128_GMAC	MAY	[RFC4543]
AUTH_AES_256_GMAC	MAY	[RFC4543]
AUTH_HMAC_SHA2_256_128	MUST	[RFC4868]
AUTH_HMAC_SHA2_512_256	SHOULD	[RFC4868]

(IoT) - This requirement is for interoperability with IoT

AUTH_NONE has been downgraded from MAY in RFC7321 to MUST NOT. The only case where AUTH_NONE is acceptable is when authenticated encryption algorithms are selected from Section 5. In all other cases, AUTH_NONE MUST NOT be selected. As ESP and AH both provide authentication, one may be tempted to combine these protocols to provide authentication. As mentioned by RFC7321, it is NOT RECOMMENDED to use ESP with NULL authentication - with non authenticated encryption - in conjunction with AH; some configurations of this combination of services have been shown to be insecure [PD10]. In addition, AUTH_NONE authentication cannot be combined with ESP NULL encryption.

AUTH_HMAC_MD5_96 and AUTH_KPDK_MD5 were not mentioned in RFC7321. As MD5 is known to be vulnerable to collisions, these algorithms MUST NOT be used.

AUTH_HMAC_SHA1_96 has been downgraded from MUST in RFC7321 to MUST- as there is an industry-wide trend to deprecate its usage.

AUTH_DES_MAC was not mentioned in RFC7321. As DES is known to be vulnerable, it MUST NOT be used.

AUTH_AES_XCBC_96 is set as SHOULD only in the scope of IoT, as Internet of Things deployments tend to prefer AES based HMAC functions in order to avoid implementing SHA2. For the wide VPN deployment, as it has not been widely adopted, it has been downgraded from SHOULD to MAY.

AUTH_AES_128_GMAC status has been downgraded from SHOULD+ to MAY. Along with AUTH_AES_192_GMAC and AUTH_AES_256_GMAC, these algorithms should only be used for AH and not for ESP. If using ENCR_NULL, AUTH_HMAC_SHA2_256_128 is recommended for integrity. If using AES-GMAC in ESP without authentication, ENCR_NULL_AUTH_AES_GMAC is recommended. Therefore, these ciphers are kept at MAY.

AUTH_HMAC_SHA2_256_128 was not mentioned in RFC7321, as no SHA2 based authentication was mentioned. AUTH_HMAC_SHA2_256_128 MUST be implemented in order to replace AUTH_HMAC_SHA1_96. Note that due to a long standing common implementation bug of this algorithm that truncates the hash at 96-bits instead of 128-bits, it is recommended that implementations prefer AUTH_HMAC_SHA2_512_256 over AUTH_HMAC_SHA2_256_128 if they implement AUTH_HMAC_SHA2_512_256.

AUTH_HMAC_SHA2_512_256 SHOULD be implemented as a future replacement of AUTH_HMAC_SHA2_256_128 or when stronger security is required. This value has been preferred to AUTH_HMAC_SHA2_384, as the additional overhead of AUTH_HMAC_SHA2_512 is negligible.

7. ESP and AH Compression Algorithms

Name	Status	Comment
IPCOMP_OUI	MUST NOT	UNSPECIFIED
IPCOMP_DEFLATE	MAY	[RFC2393]
IPCOMP_LZS	MAY	[RFC2395]
IPCOMP_LZJH	MAY	[RFC3051]

(IoT) - This requirement is for interoperability with IoT

Compression was not mentioned in RFC7321. As it is not widely deployed, it remains optional and at the MAY-level.

8. Summary of Changes from RFC 7321

The following table summarizes the changes from RFC 7321.

RFC EDITOR: PLEASE REMOVE THIS PARAGRAPH AND REPLACE XXXX IN THE TABLE BELOW WITH THE NUMBER OF THIS RFC

Algorithm	RFC 7321	RFC XXXX
ENCR_AES_GCM_16	SHOULD+	MUST
ENCR_AES_CCM_8	MAY	SHOULD
ENCR_AES_CTR	MAY	(*)
ENCR_3DES	MAY	SHOULD NOT
AUTH_HMAC_SHA1_96	MUST	MUST-
AUTH_AES_128_GMAC	SHOULD+	MAY
AUTH_NONE	MAY	MUST / MUST NOT

(*) This algorithm is not mentioned in the above sections, so it defaults to MAY.

9. Acknowledgements

Some of the wording in this document was adapted from [RFC7321], the document that this one obsoletes, which was written by D. McGrew and P. Hoffman.

10. IANA Considerations

This document has no IANA actions.

11. 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 to date leads us to believe that they will likely remain secure into the foreseeable future. However, this is not necessarily forever. Therefore, we expect that revisions of that document will be issued from time to time to reflect the current best practice in this area.

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <http://www.rfc-editor.org/info/rfc4301>.

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, DOI 10.17487/RFC4302, December 2005, <http://www.rfc-editor.org/info/rfc4302>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <http://www.rfc-editor.org/info/rfc4303>.

[RFC7321] McGrew, D. and P. Hoffman, "Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)", RFC 7321, DOI 10.17487/RFC7321, August 2014, <http://www.rfc-editor.org/info/rfc7321>.

12.2. Informative References

[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.

[RFC2393] Shacham, A., Monsour, R., Pereira, R., and M. Thomas, "IP Payload Compression Protocol (IPComp)", RFC 2393, DOI 10.17487/RFC2393, December 1998, <http://www.rfc-editor.org/info/rfc2393>.

[RFC2395] Friend, R. and R. Monsour, "IP Payload Compression Using LZS", RFC 2395, DOI 10.17487/RFC2395, December 1998, <http://www.rfc-editor.org/info/rfc2395>.

[RFC2403] Madson, C. and R. Glenn, "The Use of HMAC-MD5-96 within ESP and AH", RFC 2403, DOI 10.17487/RFC2403, November 1998, <http://www.rfc-editor.org/info/rfc2403>.

[RFC2404] Madson, C. and R. Glenn, "The Use of HMAC-SHA-1-96 within ESP and AH", RFC 2404, DOI 10.17487/RFC2404, November 1998, <http://www.rfc-editor.org/info/rfc2404>.

[RFC2405] Madson, C. and N. Doraswamy, "The ESP DES-CBC Cipher Algorithm With Explicit IV", RFC 2405, DOI 10.17487/RFC2405, November 1998, <http://www.rfc-editor.org/info/rfc2405>.

[RFC2410] Glenn, R. and S. Kent, "The NULL Encryption Algorithm and Its Use With IPsec", RFC 2410, DOI 10.17487/RFC2410, November 1998, <http://www.rfc-editor.org/info/rfc2410>.

[RFC2451] Pereira, R. and R. Adams, "The ESP CBC-Mode Cipher Algorithms", RFC 2451, DOI 10.17487/RFC2451, November 1998, <http://www.rfc-editor.org/info/rfc2451>.

[RFC3051] Heath, J. and J. Border, "IP Payload Compression Using ITU-T V.44 Packet Method", RFC 3051, DOI 10.17487/RFC3051, January 2001, <http://www.rfc-editor.org/info/rfc3051>.

[RFC3566] Frankel, S. and H. Herbert, "The AES-XCBC-MAC-96 Algorithm and Its Use With IPsec", RFC 3566, DOI 10.17487/RFC3566, September 2003, <http://www.rfc-editor.org/info/rfc3566>.

[RFC3602] Frankel, S., Glenn, R., and S. Kelly, "The AES-CBC Cipher Algorithm and Its Use with IPsec", RFC 3602, DOI 10.17487/RFC3602, September 2003, <http://www.rfc-editor.org/info/rfc3602>.

[RFC3686] Housley, R., "Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)", RFC 3686, DOI 10.17487/RFC3686, January 2004, <http://www.rfc-editor.org/info/rfc3686>.

[RFC4106] Viega, J. and D. McGrew, "The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)", RFC 4106, DOI 10.17487/RFC4106, June 2005, <http://www.rfc-editor.org/info/rfc4106>.

[RFC4309] Housley, R., "Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)", RFC 4309, DOI 10.17487/RFC4309, December 2005, <http://www.rfc-editor.org/info/rfc4309>.

[RFC4543] McGrew, D. and J. Viega, "The Use of Galois Message Authentication Code (GMAC) in IPsec ESP and AH", RFC 4543, DOI 10.17487/RFC4543, May 2006, <http://www.rfc-editor.org/info/rfc4543>.

[RFC4868] Kelly, S. and S. Frankel, "Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec", RFC 4868, DOI 10.17487/RFC4868, May 2007, <http://www.rfc-editor.org/info/rfc4868>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <http://www.rfc-editor.org/info/rfc7296>.

[RFC7634] Nir, Y., "ChaCha20, Poly1305, and Their Use in the Internet Key Exchange Protocol (IKE) and IPsec", RFC 7634, DOI 10.17487/RFC7634, August 2015, <http://www.rfc-editor.org/info/rfc7634>.

Authors' Addresses

Paul Wouters
Red Hat

Email: pwouters@redhat.com

Daniel Migault
Ericsson
8400 boulevard Decarie
Montreal, QC H4P 2N2
Canada

Phone: +1 514-452-2160
Email: daniel.migault@ericsson.com

John Mattsson
Ericsson AB
SE-164 80 Stockholm
Sweden

Email: john.mattsson@ericsson.com

Yoav Nir
Check Point Software Technologies Ltd.
5 Hasolelim st.
Tel Aviv 6789735
Israel

Email: ynir.ietf@gmail.com

Tero Kivinen
INSIDE Secure
Eerikinkatu 28
HELSINKI FI-00180
FI

Email: kivinen@iki.fi

Network
Internet-Draft
Intended status: Standards Track
Expires: December 1, 2017

T. Pauly
Apple Inc.
S. Touati
Ericsson
R. Mantha
Cisco Systems
May 30, 2017

TCP Encapsulation of IKE and IPsec Packets
draft-ietf-ipsecme-tcp-encaps-10

Abstract

This document describes a method to transport IKE and IPsec packets over a TCP connection for traversing network middleboxes that may block IKE negotiation over UDP. This method, referred to as TCP encapsulation, involves sending both IKE packets for Security Association establishment and ESP packets over a TCP connection. This method is intended to be used as a fallback option when IKE cannot be negotiated over UDP.

Status of This Memo

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

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

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

This Internet-Draft will expire on December 1, 2017.

Copyright Notice

Table of Contents

1. Introduction

IKEv2 [RFC7296] is a protocol for establishing IPsec Security Associations (SAs), using IKE messages over UDP for control traffic, and using Encapsulating Security Payload (ESP) messages for encrypted data traffic. Many network middleboxes that filter traffic on public hotspots block all UDP traffic, including IKE and IPsec, but allow TCP connections through since they appear to be web traffic. Devices on these networks that need to use IPsec (to access private enterprise networks, to route voice-over-IP calls to carrier networks, or because of security policies) are unable to establish IPsec SAs. This document defines a method for encapsulating both the IKE control messages as well as the IPsec data messages within a TCP connection.

Using TCP as a transport for IPsec packets adds a third option to the list of traditional IPsec transports:

1. Direct. Currently, IKE negotiations begin over UDP port 500. If no NAT is detected between the Initiator and the Responder, then subsequent IKE packets are sent over UDP port 500 and IPsec data packets are sent using ESP [RFC4303].

2. UDP Encapsulation [RFC3948]. If a NAT is detected between the Initiator and the Responder, then subsequent IKE packets are sent over UDP port 4500 with four bytes of zero at the start of the UDP payload and ESP packets are sent out over UDP port 4500. Some peers default to using UDP encapsulation even when no NAT are detected on the path as some middleboxes do not support IP protocols other than TCP and UDP.

3. TCP Encapsulation. If both of the other two methods are not available or appropriate, both IKE negotiation packets as well as ESP packets can be sent over a single TCP connection to the peer.

Direct use of ESP or UDP Encapsulation should be preferred by IKE implementations due to performance concerns when using TCP Encapsulation Section 12. Most implementations should use TCP Encapsulation only on networks where negotiation over UDP has been attempted without receiving responses from the peer, or if a network is known to not support UDP.

1.1. Prior Work and Motivation

Encapsulating IKE connections within TCP streams is a common approach to solve the problem of UDP packets being blocked by network middleboxes. The goal of this document is to promote

interoperability by providing a standard method of framing IKE and ESP message within streams, and to provide guidelines for how to configure and use TCP encapsulation.

Some previous alternatives include:

Cellular Network Access Interworking Wireless LAN (IWLAN) uses IKEv2 to create secure connections to cellular carrier networks for making voice calls and accessing other network services over Wi-Fi networks. 3GPP has recommended that IKEv2 and ESP packets be sent within a TLS connection to be able to establish connections on restrictive networks.

ISAKMP over TCP Various non-standard extensions to ISAKMP have been deployed that send IPsec traffic over TCP or TCP-like packets.

SSL VPNs Many proprietary VPN solutions use a combination of TLS and IPsec in order to provide reliability. These often run on TCP port 443.

IKEv2 over TCP IKEv2 over TCP as described in [I-D.nir-ipsecme-ike-tcp] is used to avoid UDP fragmentation.

The goal of this specification is to provide a standardized method for using TCP streams to transport IPsec that is compatible with the current IKE standard, and avoids the overhead of other alternatives that always rely on TCP or TLS.

1.2. Terminology and Notation

This document distinguishes between the IKE peer that initiates TCP connections to be used for TCP encapsulation and the roles of Initiator and Responder for particular IKE messages. During the course of IKE exchanges, the role of IKE Initiator and Responder may swap for a given SA (as with IKE SA Rekeys), while the initiator of the TCP connection is still responsible for tearing down the TCP connection and re-establishing it if necessary. For this reason, this document will use the term "TCP Originator" to indicate the IKE peer that initiates TCP connections. The peer that receives TCP connections will be referred to as the "TCP Responder". If an IKE SA is rekeyed one or more times, the TCP Originator MUST remain the peer that originally initiated the first IKE SA.

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. Configuration

One of the main reasons to use TCP encapsulation is that UDP traffic may be entirely blocked on a network. Because of this, support for TCP encapsulation is not specifically negotiated in the IKE exchange. Instead, support for TCP encapsulation must be pre-configured on both the TCP Originator and the TCP Responder.

Implementations MUST support TCP encapsulation on TCP port 4500, which is reserved for IPsec NAT Traversal.

Beyond a flag indicating support for TCP encapsulation, the configuration for each peer can include the following optional parameters:

- Alternate TCP ports on which the specific TCP Responder listens for incoming connections. Note that the TCP Originator may initiate TCP connections to the TCP Responder from any local port.

- An extra framing protocol to use on top of TCP to further encapsulate the stream of IKE and IPsec packets. See Appendix A for a detailed discussion.

Since TCP encapsulation of IKE and IPsec packets adds overhead and has potential performance trade-offs compared to direct or UDP-encapsulated SAs (as described in Performance Considerations, Section 12), implementations SHOULD prefer ESP direct or UDP encapsulated SAs over TCP encapsulated SAs when possible.

3. TCP-Encapsulated Header Formats

Like UDP encapsulation, TCP encapsulation uses the first four bytes of a message to differentiate IKE and ESP messages. TCP encapsulation also adds a length field to define the boundaries of messages within a stream. The message length is sent in a 16-bit field that precedes every message. If the first 32-bits of the message are zeros (a Non-ESP Marker), then the contents comprise an IKE message. Otherwise, the contents comprise an ESP message. Authentication Header (AH) messages are not supported for TCP encapsulation.

Although a TCP stream may be able to send very long messages, implementations SHOULD limit message lengths to typical UDP datagram ESP payload lengths. The maximum message length is used as the effective MTU for connections that are being encrypted using ESP, so the maximum message length will influence characteristics of inner connections, such as the TCP Maximum Segment Size (MSS).

Note that this method of encapsulation will also work for placing IKE and ESP messages within any protocol that presents a stream abstraction, beyond TCP.

3.1. TCP-Encapsulated IKE Header Format

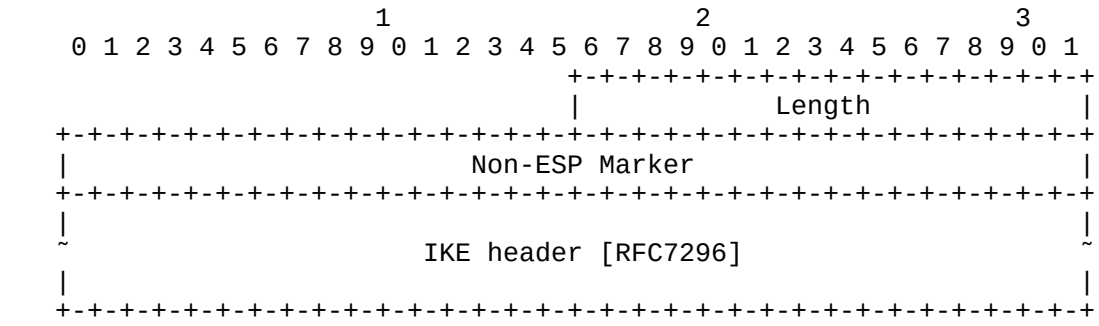


Figure 1

The IKE header is preceded by a 16-bit length field in network byte order that specifies the length of the IKE message (including the Non-ESP marker) within the TCP stream. As with IKE over UDP port 4500, a zeroed 32-bit Non-ESP Marker is inserted before the start of the IKE header in order to differentiate the traffic from ESP traffic between the same addresses and ports.

- Length (2 octets, unsigned integer) - Length of the IKE packet including the Length Field and Non-ESP Marker.

3.2. TCP-Encapsulated ESP Header Format

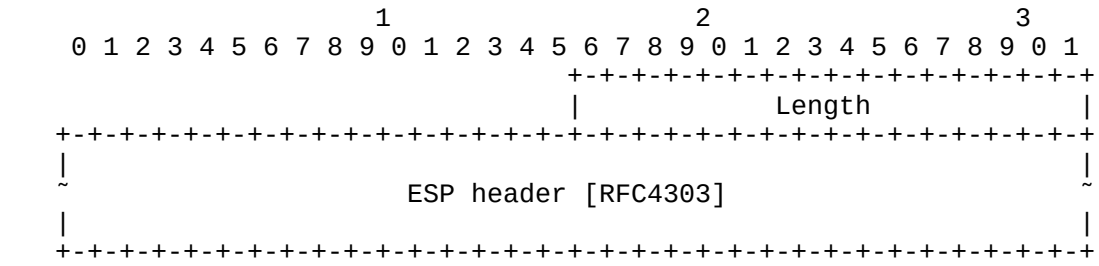


Figure 2

The ESP header is preceded by a 16-bit length field in network byte order that specifies the length of the ESP packet within the TCP stream.

The SPI field in the ESP header MUST NOT be a zero value.

- o Length (2 octets, unsigned integer) - Length of the ESP packet including the Length Field.

4. TCP-Encapsulated Stream Prefix

Each stream of bytes used for IKE and IPsec encapsulation MUST begin with a fixed sequence of six bytes as a magic value, containing the characters "IKETCP" as ASCII values. This value is intended to identify and validate that the TCP connection is being used for TCP encapsulation as defined in this document, to avoid conflicts with the prevalence of previous non-standard protocols that used TCP port 4500. This value is only sent once, by the TCP Originator only, at the beginning of any stream of IKE and ESP messages.

If other framing protocols are used within TCP to further encapsulate or encrypt the stream of IKE and ESP messages, the Stream Prefix must be at the start of the TCP Originator’s IKE and ESP message stream within the added protocol layer [Appendix A]. Although some framing protocols do support negotiating inner protocols, the stream prefix should always be used in order for implementations to be as generic as possible and not rely on other framing protocols on top of TCP.

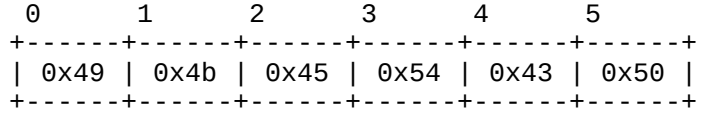


Figure 3

5. Applicability

TCP encapsulation is applicable only when it has been configured to be used with specific IKE peers. If a Responder is configured to use TCP encapsulation, it MUST listen on the configured port(s) in case any peers will initiate new IKE sessions. Initiators MAY use TCP encapsulation for any IKE session to a peer that is configured to support TCP encapsulation, although it is recommended that Initiators should only use TCP encapsulation when traffic over UDP is blocked.

Since the support of TCP encapsulation is a configured property, not a negotiated one, it is recommended that if there are multiple IKE endpoints representing a single peer (such as multiple machines with different IP addresses when connecting by Fully-Qualified Domain Name, or endpoints used with IKE redirection), all of the endpoints equally support TCP encapsulation.

If TCP encapsulation is being used for a specific IKE SA, all messages for that IKE SA and its Child SAs MUST be sent over a TCP connection until the SA is deleted or MOBIKE is used to change the SA endpoints and/or encapsulation protocol. See Section 8 for more details on using MOBIKE to transition between encapsulation modes.

5.1. Recommended Fallback from UDP

Since UDP is the preferred method of transport for IKE messages, implementations that use TCP encapsulation should have an algorithm for deciding when to use TCP after determining that UDP is unusable. If an Initiator implementation has no prior knowledge about the network it is on and the status of UDP on that network, it SHOULD always attempt negotiate IKE over UDP first. IKEv2 defines how to use retransmission timers with IKE messages, and IKE_SA_INIT messages specifically [RFC7296]. Generally, this means that the implementation will define a frequency of retransmission, and the maximum number of retransmissions allowed before marking the IKE SA as failed. An implementation can attempt negotiation over TCP once it has hit the maximum retransmissions over UDP, or slightly before to reduce connection setup delays. It is recommended that the initial message over UDP is retransmitted at least once before falling back to TCP, unless the Initiator knows beforehand that the network is likely to block UDP.

6. Connection Establishment and Teardown

When the IKE Initiator uses TCP encapsulation, it will initiate a TCP connection to the Responder using the configured TCP port. The first bytes sent on the stream MUST be the stream prefix value [Section 4]. After this prefix, encapsulated IKE messages will negotiate the IKE SA and initial Child SA [RFC7296]. After this point, both encapsulated IKE Figure 1 and ESP Figure 2 messages will be sent over the TCP connection. The TCP Responder MUST wait for the entire stream prefix to be received on the stream before trying to parse out any IKE or ESP messages. The stream prefix is sent only once, and only by the TCP Originator.

In order to close an IKE session, either the Initiator or Responder SHOULD gracefully tear down IKE SAs with DELETE payloads. Once the SA has been deleted, the TCP Originator SHOULD close the TCP connection if it does not intend to use the connection for another IKE session to the TCP Responder. If the connection is left idle, and the TCP Responder needs to clean up resources, the TCP Responder MAY close the TCP connection.

An unexpected FIN or a RST on the TCP connection may indicate either a loss of connectivity, an attack, or some other error. If a DELETE

payload has not been sent, both sides SHOULD maintain the state for their SAs for the standard lifetime or time-out period. The TCP Originator is responsible for re-establishing the TCP connection if it is torn down for any unexpected reason. Since new TCP connections may use different ports due to NAT mappings or local port allocations changing, the TCP Responder MUST allow packets for existing SAs to be received from new source ports.

A peer MUST discard a partially received message due to a broken connection.

Whenever the TCP Originator opens a new TCP connection to be used for an existing IKE SA, it MUST send the stream prefix first, before any IKE or ESP messages. This follows the same behavior as the initial TCP connection.

If a TCP connection is being used to resume a previous IKE session, the TCP Responder can recognize the session using either the IKE SPI from an encapsulated IKE message or the ESP SPI from an encapsulated ESP message. If the session had been fully established previously, it is suggested that the TCP Originator send an UPDATE_SA_ADDRESSES message if MOBIKE is supported, or an INFORMATIONAL message (a keepalive) otherwise.

The TCP Responder MUST NOT accept any messages for the existing IKE session on a new incoming connection unless that connection begins with the stream prefix. If either the TCP Originator or TCP Responder detects corruption on a connection that was started with a valid stream prefix, it SHOULD close the TCP connection. The connection can be determined as corrupted if there are too many subsequent messages that cannot be parsed as valid IKE messages or ESP messages with known SPIs, or if the authentication check for an ESP message with a known SPI fails. Implementations SHOULD NOT tear down a connection if only a single ESP message has an unknown SPI, since the SPI databases may be momentarily out of sync. If there is instead a syntax issue within an IKE message, an implementation MUST send the INVALID_SYNTAX notify payload and tear down the IKE SA as usual, rather than tearing down the TCP connection directly.

An TCP Originator SHOULD only open one TCP connection per IKE SA, over which it sends all of the corresponding IKE and ESP messages. This helps ensure that any firewall or NAT mappings allocated for the TCP connection apply to all of the traffic associated with the IKE SA equally.

Similarly, a TCP Responder SHOULD at any given time send packets for an IKE SA and its Child SAs over only one TCP connection. It SHOULD choose the TCP connection on which it last received a valid and

decryptable IKE or ESP message. In order to be considered valid for choosing a TCP connection, an IKE message must be successfully decrypted and authenticated, not be a retransmission of a previously received message, and be within the expected window for IKE message IDs. Similarly, an ESP message must pass authentication checks and be decrypted, not be a replay of a previous message.

Since a connection may be broken and a new connection re-established by the TCP Originator without the TCP Responder being aware, a TCP Responder SHOULD accept receiving IKE and ESP messages on both old and new connections until the old connection is closed by the TCP Originator. A TCP Responder MAY close a TCP connection that it perceives as idle and extraneous (one previously used for IKE and ESP messages that has been replaced by a new connection).

Multiple IKE SAs MUST NOT share a single TCP connection, unless one is a rekey of an existing IKE SA, in which case there will temporarily be two IKE SAs on the same TCP connection.

7. Interaction with NAT Detection Payloads

When negotiating over UDP port 500, IKE_SA_INIT packets include NAT_DETECTION_SOURCE_IP and NAT_DETECTION_DESTINATION_IP payloads to determine if UDP encapsulation of IPsec packets should be used. These payloads contain SHA-1 digests of the SPIs, IP addresses, and ports as defined in [RFC7296]. IKE_SA_INIT packets sent on a TCP connection SHOULD include these payloads with the same content as when sending over UDP, and SHOULD use the applicable TCP ports when creating and checking the SHA-1 digests.

If a NAT is detected due to the SHA-1 digests not matching the expected values, no change should be made for encapsulation of subsequent IKE or ESP packets, since TCP encapsulation inherently supports NAT traversal. Implementations MAY use the information that a NAT is present to influence keep-alive timer values.

If a NAT is detected, implementations need to handle transport mode TCP and UDP packet checksum fixup as defined for UDP encapsulation in [RFC3948].

8. Using MOBIKE with TCP encapsulation

When an IKE session that has negotiated MOBIKE [RFC4555] is transitioning between networks, the Initiator of the transition may switch between using TCP encapsulation, UDP encapsulation, or no encapsulation. Implementations that implement both MOBIKE and TCP encapsulation MUST support dynamically enabling and disabling TCP encapsulation as interfaces change.

When a MOBIKE-enabled Initiator changes networks, the UPDATE_SA_ADDRESSES notification SHOULD be sent out first over UDP before attempting over TCP. If there is a response to the UPDATE_SA_ADDRESSES notification sent over UDP, then the ESP packets should be sent directly over IP or over UDP port 4500 (depending on if a NAT was detected), regardless of if a connection on a previous network was using TCP encapsulation. Similarly, if the Responder only responds to the UPDATE_SA_ADDRESSES notification over TCP, then the ESP packets should be sent over the TCP connection, regardless of if a connection on a previous network did not use TCP encapsulation.

9. Using IKE Message Fragmentation with TCP encapsulation

IKE Message Fragmentation [RFC7383] is not required when using TCP encapsulation, since a TCP stream already handles the fragmentation of its contents across packets. Since fragmentation is redundant in this case, implementations might choose to not negotiate IKE fragmentation. Even if fragmentation is negotiated, an implementation SHOULD NOT send fragments when going over a TCP connection, although it MUST support receiving fragments.

If an implementation supports both MOBIKE and IKE fragmentation, it SHOULD negotiate IKE fragmentation over a TCP encapsulated session in case the session switches to UDP encapsulation on another network.

10. Considerations for Keep-alives and DPD

Encapsulating IKE and IPsec inside of a TCP connection can impact the strategy that implementations use to detect peer liveness and to maintain middlebox port mappings. Peer liveness should be checked using IKE Informational packets [RFC7296].

In general, TCP port mappings are maintained by NATs longer than UDP port mappings, so IPsec ESP NAT keep-alives [RFC3948] SHOULD NOT be sent when using TCP encapsulation. Any implementation using TCP encapsulation MUST silently drop incoming NAT keep-alive packets, and not treat them as errors. NAT keep-alive packets over a TCP encapsulated IPsec connection will be sent with a length value of 1 byte, whose value is 0xFF Figure 2.

Note that depending on the configuration of TCP and TLS on the connection, TCP keep-alives [RFC1122] and TLS keep-alives [RFC6520] may be used. These MUST NOT be used as indications of IKE peer liveness.

11. Middlebox Considerations

Many security networking devices such as Firewalls or Intrusion Prevention Systems, network optimization/acceleration devices and Network Address Translation (NAT) devices keep the state of sessions that traverse through them.

These devices commonly track the transport layer and/or the application layer data to drop traffic that is anomalous or malicious in nature. While many of these devices will be more likely to pass TCP-encapsulated traffic as opposed to UDP-encapsulated traffic, some may still block or interfere with TCP-encapsulated IKE and IPsec.

A network device that monitors the transport layer will track the state of TCP sessions, such as TCP sequence numbers. TCP encapsulation of IKE should therefore use standard TCP behaviors to avoid being dropped by middleboxes.

12. Performance Considerations

Several aspects of TCP encapsulation for IKE and IPsec packets may negatively impact the performance of connections within a tunnel-mode IPsec SA. Implementations should be aware of these performance impacts and take these into consideration when determining when to use TCP encapsulation. Implementations SHOULD favor using direct ESP or UDP encapsulation over TCP encapsulation whenever possible.

12.1. TCP-in-TCP

If the outer connection between IKE peers is over TCP, inner TCP connections may suffer effects from using TCP within TCP. Running TCP within TCP is discouraged, since the TCP algorithms generally assume that they are running over an unreliable datagram layer.

If the outer (tunnel) TCP connection experiences packet loss, this loss will be hidden from any inner TCP connections, since the outer connection will retransmit to account for the losses. Since the outer TCP connection will deliver the inner messages in order, any messages after a lost packet may have to wait until the loss is recovered. This means that loss on the outer connection will be interpreted only as delay by inner connections. The burstiness of inner traffic can increase, since a large number of inner packets may be delivered across the tunnel at once. The inner TCP connection may interpret a long period of delay as a transmission problem, triggering a retransmission timeout, which will cause spurious retransmissions. The sending rate of the inner connection may be unnecessarily reduced if the retransmissions are not detected as spurious in time.

The inner TCP connection's round-trip-time estimation will be affected by the burstiness of the outer TCP connection if there are long delays when packets are retransmitted by the outer TCP connection. This will make the congestion control loop of the inner TCP traffic less reactive, potentially permanently leading to a lower sending rate than the outer TCP would allow for.

TCP-in-TCP can also lead to increased buffering, or bufferbloat. This can occur when the window size of the outer TCP connection is reduced, and becomes smaller than the window sizes of the inner TCP connections. This can lead to packets backing up in the outer TCP connection's send buffers. In order to limit this effect, the outer TCP connection should have limits on its send buffer size, and on the rate at which it reduces its window size.

Note that any negative effects will be shared between all flows going through the outer TCP connection. This is of particular concern for any latency-sensitive or real-time applications using the tunnel. If such traffic is using a TCP encapsulated IPsec connection, it is recommended that the number of inner connections sharing the tunnel be limited as much as possible.

12.2. Added Reliability for Unreliable Protocols

Since ESP is an unreliable protocol, transmitting ESP packets over a TCP connection will change the fundamental behavior of the packets. Some application-level protocols that prefer packet loss to delay (such as Voice over IP or other real-time protocols) may be negatively impacted if their packets are retransmitted by the TCP connection due to packet loss.

12.3. Quality of Service Markings

Quality of Service (QoS) markings, such as DSCP and Traffic Class, should be used with care on TCP connections used for encapsulation. Individual packets SHOULD NOT use different markings than the rest of the connection, since packets with different priorities may be routed differently and cause unnecessary delays in the connection.

12.4. Maximum Segment Size

A TCP connection used for IKE encapsulation SHOULD negotiate its maximum segment size (MSS) in order to avoid unnecessary fragmentation of packets.

12.5. Tunnelling ECN in TCP

Since there is not a one-to-one relationship between outer IP packets and inner ESP/IP messages when using TCP encapsulation, the markings for Explicit Congestion Notification (ECN) [RFC3168] cannot be simply mapped. However, any ECN Congestion Experienced (CE) marking on inner messages should be preserved through the tunnel.

Implementations SHOULD follow the ECN compatibility mode as described in [RFC6040]. In compatibility mode, the outer TCP connection SHOULD mark its packets as not ECN-capable, and MUST NOT clear any ECN markings on inner packets. Note that outer packets may be ECN marked even though the outer connection did not negotiate support for ECN. If an implementation receives such an outer packet, it MAY propagate the markings as described in the Default Tunnel Egress Behaviour [RFC6040] for any inner packet contained within a single outer TCP packet, or simply apply the rules as if the outer packet were Not-ECT if the inner packet spans multiple outer packets.

13. Security Considerations

IKE Responders that support TCP encapsulation may become vulnerable to new Denial-of-Service (DoS) attacks that are specific to TCP, such as SYN-flooding attacks. TCP Responders should be aware of this additional attack-surface.

TCP Responders should be careful to ensure that the stream prefix "IKETCP" uniquely identifies incoming streams as ones that use the TCP encapsulation protocol, and they are not running any other protocols on the same listening port that could conflict with this.

Attackers may be able to disrupt the TCP connection by sending spurious RST packets. Due to this, implementations SHOULD make sure that IKE session state persists even if the underlying TCP connection is torn down.

If MOBIKE is being used, all of the security considerations outlined for MOBIKE apply [RFC4555].

Similarly to MOBIKE, TCP encapsulation requires a TCP Responder to handle changing of source address and port due to network or connection disruption. The successful delivery of valid IKE or ESP messages over a new TCP connection is used by the TCP Responder to determine where to send subsequent responses. If an attacker is able to send packets on a new TCP connection that pass the validation checks of the TCP Responder, it can influence which path future packets take. For this reason, the validation of messages on the TCP Responder must include decryption, authentication, and replay checks.

Since TCP provides a reliable, in-order delivery of ESP messages, the ESP Anti-Replay Window size SHOULD be set to 1. See [RFC4303] for a complete description of the ESP Anti-Replay Window. This increases the protection of implementations against replay attacks.

14. IANA Considerations

TCP port 4500 is already allocated to IPsec for NAT Traversal. This port SHOULD be used for TCP encapsulated IKE and ESP as described in this document.

This document updates the reference for TCP port 4500:

Keyword	Decimal	Description	Reference
ipsec-nat-t	4500/tcp	IPsec NAT-Traversal	[RFC-this-rfc]

Figure 4

15. Acknowledgments

The authors would like to acknowledge the input and advice of Stuart Cheshire, Delziel Fernandes, Yoav Nir, Christoph Paasch, Yaron Sheffer, David Schinazi, Graham Bartlett, Byju Pularikkal, March Wu, Kingwel Xie, Valery Smyslov, Jun Hu, and Tero Kivinen. Special thanks to Eric Kinnear for his implementation work.

16. References

16.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3948] Huttunen, A., Swander, B., Volpe, V., DiBurro, L., and M. Stenberg, "UDP Encapsulation of IPsec ESP Packets", RFC 3948, DOI 10.17487/RFC3948, January 2005, <http://www.rfc-editor.org/info/rfc3948>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <http://www.rfc-editor.org/info/rfc4303>.

[RFC6040] Briscoe, B., "Tunnelling of Explicit Congestion Notification", RFC 6040, DOI 10.17487/RFC6040, November 2010, <http://www.rfc-editor.org/info/rfc6040>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <http://www.rfc-editor.org/info/rfc7296>.

16.2. Informative References

[I-D.nir-ipsecme-ike-tcp]
Nir, Y., "A TCP transport for the Internet Key Exchange", draft-nir-ipsecme-ike-tcp-01 (work in progress), July 2012.

[RFC1122] Braden, R., Ed., "Requirements for Internet Hosts - Communication Layers", STD 3, RFC 1122, DOI 10.17487/RFC1122, October 1989, <http://www.rfc-editor.org/info/rfc1122>.

[RFC2817] Khare, R. and S. Lawrence, "Upgrading to TLS Within HTTP/1.1", RFC 2817, DOI 10.17487/RFC2817, May 2000, <http://www.rfc-editor.org/info/rfc2817>.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, DOI 10.17487/RFC3168, September 2001, <http://www.rfc-editor.org/info/rfc3168>.

[RFC4555] Eronen, P., "IKEv2 Mobility and Multihoming Protocol (MOBIKE)", RFC 4555, DOI 10.17487/RFC4555, June 2006, <http://www.rfc-editor.org/info/rfc4555>.

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

[RFC6520] Seggelmann, R., Tuexen, M., and M. Williams, "Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS) Heartbeat Extension", RFC 6520, DOI 10.17487/RFC6520, February 2012, <http://www.rfc-editor.org/info/rfc6520>.

[RFC7383] Smyslov, V., "Internet Key Exchange Protocol Version 2 (IKEv2) Message Fragmentation", RFC 7383, DOI 10.17487/RFC7383, November 2014, <http://www.rfc-editor.org/info/rfc7383>.

Appendix A. Using TCP encapsulation with TLS

This section provides recommendations on how to use TLS in addition to TCP encapsulation.

When using TCP encapsulation, implementations may choose to use TLS [RFC5246] on the TCP connection to be able to traverse middle-boxes, which may otherwise block the traffic.

If a web proxy is applied to the ports used for the TCP connection, and TLS is being used, the TCP Originator can send an HTTP CONNECT message to establish an SA through the proxy [RFC2817].

The use of TLS should be configurable on the peers, and may be used as the default when using TCP encapsulation, or else be a fallback when basic TCP encapsulation fails. The TCP Responder may expect to read encapsulated IKE and ESP packets directly from the TCP connection, or it may expect to read them from a stream of TLS data packets. The TCP Originator should be pre-configured to use TLS or not when communicating with a given port on the TCP Responder.

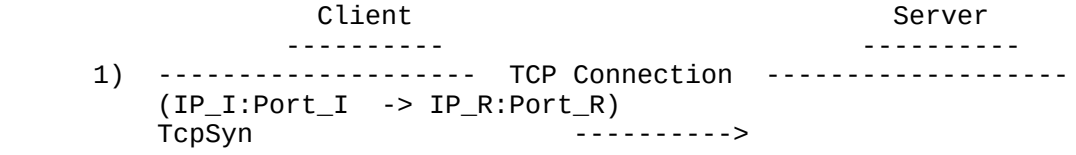
When new TCP connections are re-established due to a broken connection, TLS must be re-negotiated. TLS Session Resumption is recommended to improve efficiency in this case.

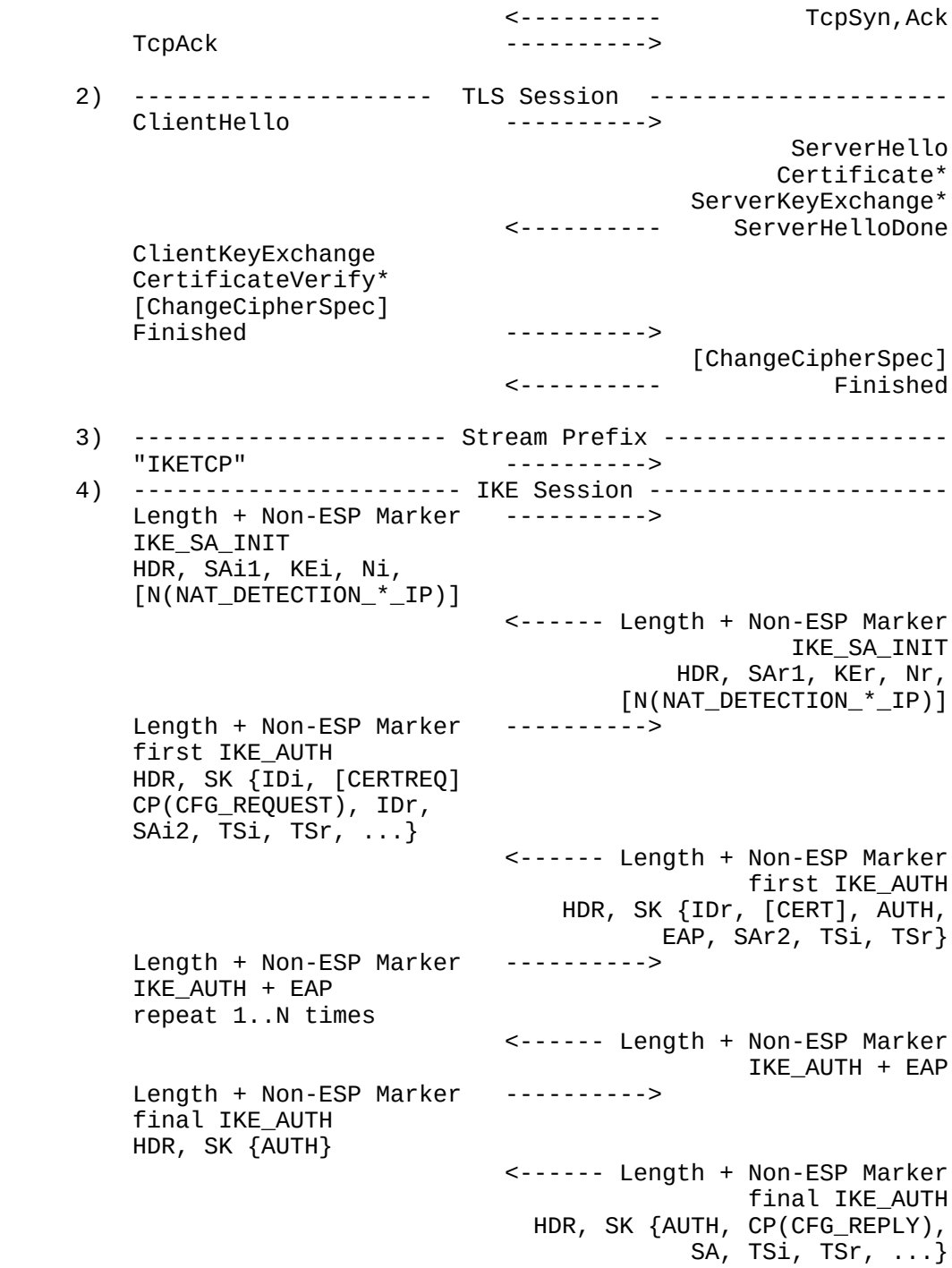
The security of the IKE session is entirely derived from the IKE negotiation and key establishment and not from the TLS session (which in this context is only used for encapsulation purposes), therefore when TLS is used on the TCP connection, both the TCP Originator and TCP Responder SHOULD allow the NULL cipher to be selected for performance reasons.

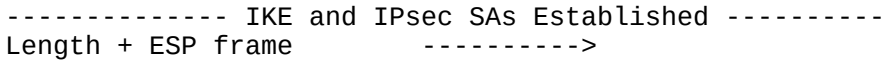
Implementations should be aware that the use of TLS introduces another layer of overhead requiring more bytes to transmit a given IKE and IPsec packet. For this reason, direct ESP, UDP encapsulation, or TCP encapsulation without TLS should be preferred in situations in which TLS is not required in order to traverse middle-boxes.

Appendix B. Example exchanges of TCP Encapsulation with TLS

B.1. Establishing an IKE session




TcpSyn,Ack
TcpAck
2) TLS Session
ClientHello
ServerHello
Certificate*
ServerKeyExchange*
ServerHelloDone
ClientKeyExchange
CertificateVerify*
[ChangeCipherSpec]
Finished
[ChangeCipherSpec]
Finished
3) Stream Prefix
"IKETCP"
4) IKE Session
Length + Non-ESP Marker
IKE_SA_INIT
HDR, SAi1, KEi, Ni,
[N(NAT_DETECTION_*_IP)]
Length + Non-ESP Marker
IKE_SA_INIT
HDR, SAr1, KEr, Nr,
[N(NAT_DETECTION_*_IP)]
Length + Non-ESP Marker
first IKE_AUTH
HDR, SK {IDi, [CERTREQ]
CP(CFG_REQUEST), IDr,
SAi2, TSi, TSr, ...}
Length + Non-ESP Marker
first IKE_AUTH
HDR, SK {IDr, [CERT], AUTH,
EAP, SAr2, TSi, TSr}
Length + Non-ESP Marker
IKE_AUTH + EAP
repeat 1..N times
Length + Non-ESP Marker
IKE_AUTH + EAP
Length + Non-ESP Marker
final IKE_AUTH
HDR, SK {AUTH}
Length + Non-ESP Marker
final IKE_AUTH
HDR, SK {AUTH, CP(CFG_REPLY),
SA, TSi, TSr, ...}



1. Client establishes a TCP connection with the server on port 4500, or an alternate pre-configured port that the server is listening on.

2. If configured to use TLS, the client initiates a TLS handshake. During the TLS handshake, the server SHOULD NOT request the client's certificate, since authentication is handled as part of IKE negotiation.

3. Client send the Stream Prefix for TCP encapsulated IKE Section 4 traffic to signal the beginning of IKE negotiation.

4. Client and server establish an IKE connection. This example shows EAP-based authentication, although any authentication type may be used.

B.2. Deleting an IKE session

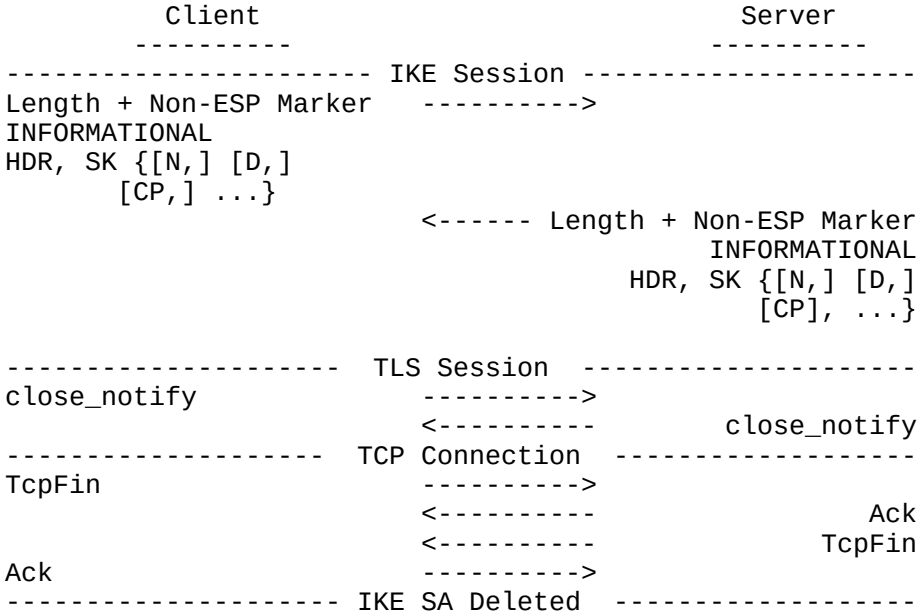


Figure 6

1. Client and server exchange INFORMATIONAL messages to notify IKE SA deletion.

2. Client and server negotiate TLS session deletion using TLS CLOSE_NOTIFY.

3. The TCP connection is torn down.

The deletion of the IKE SA should lead to the disposal of the underlying TLS and TCP state.

B.3. Re-establishing an IKE session

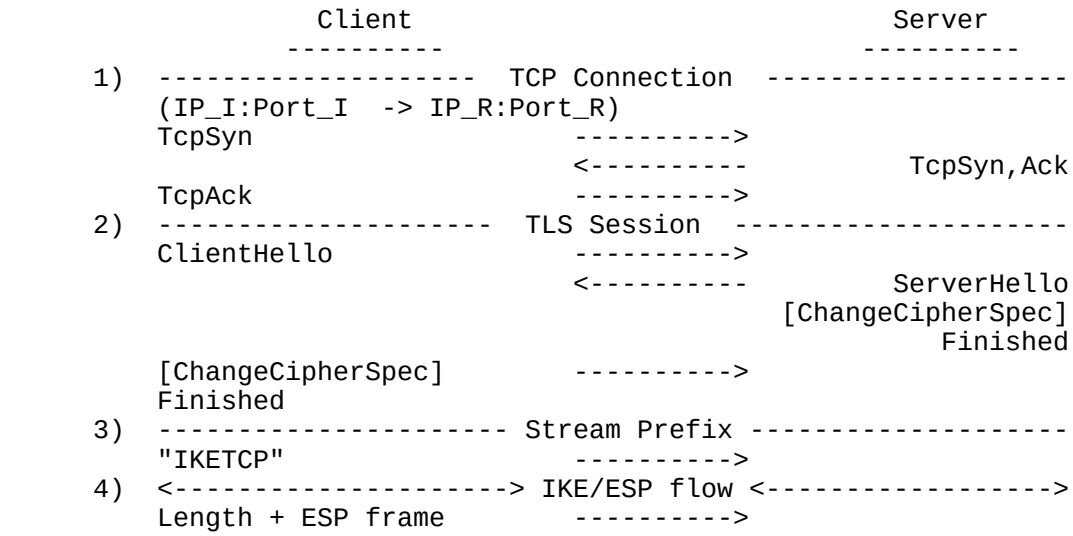


Figure 7

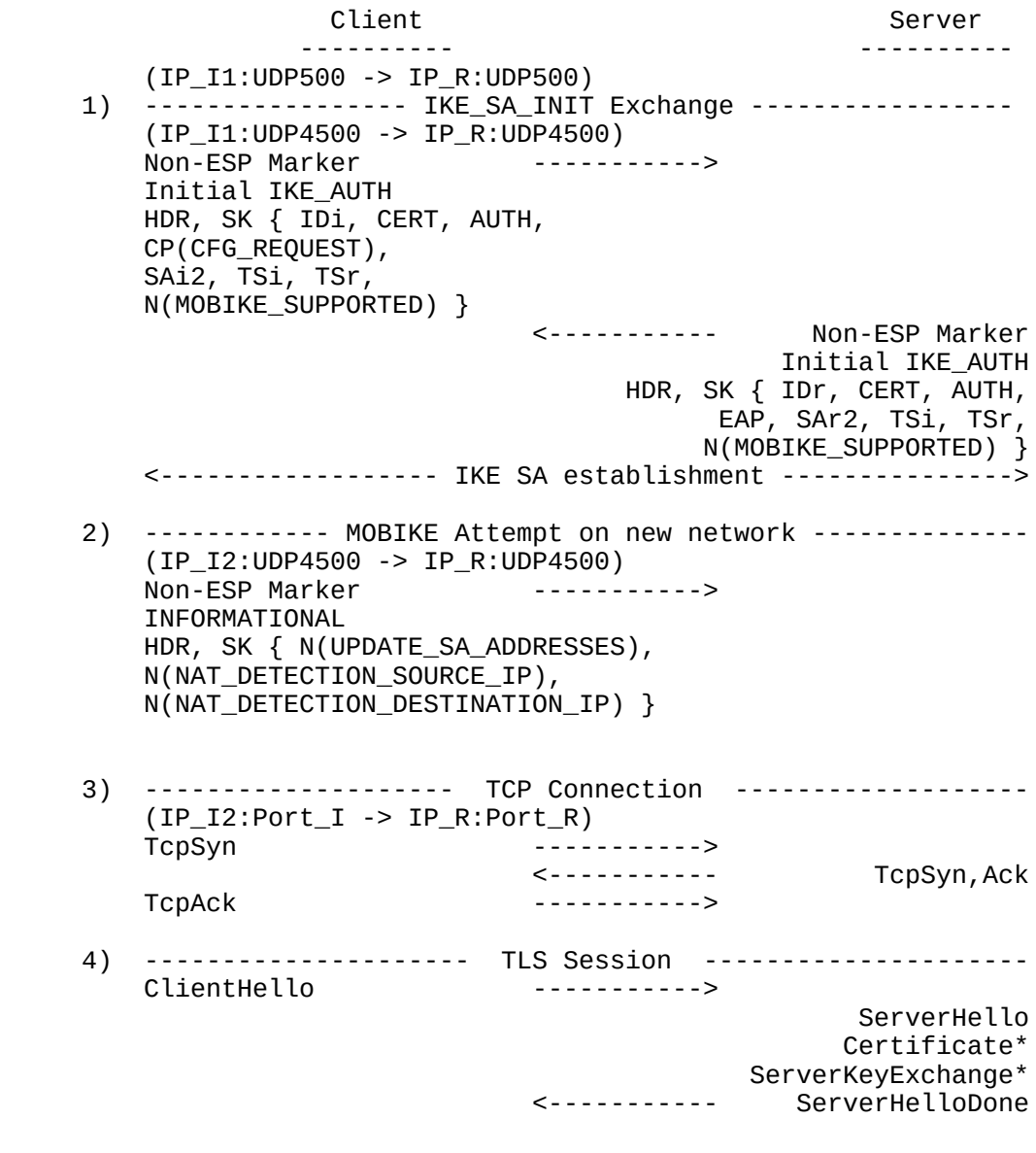
1. If a previous TCP connection was broken (for example, due to a RST), the client is responsible for re-initiating the TCP connection. The TCP Originator's address and port (IP_I and Port_I) may be different from the previous connection's address and port.

2. In ClientHello TLS message, the client SHOULD send the Session ID it received in the previous TLS handshake if available. It is up to the server to perform either an abbreviated handshake or full handshake based on the session ID match.

3. After TCP and TLS are complete, the client sends the Stream Prefix for TCP encapsulated IKE traffic Section 4.

4. The IKE and ESP packet flow can resume. If MOBIKE is being used, the Initiator SHOULD send UPDATE_SA_ADDRESSES.

B.4. Using MOBIKE between UDP and TCP Encapsulation



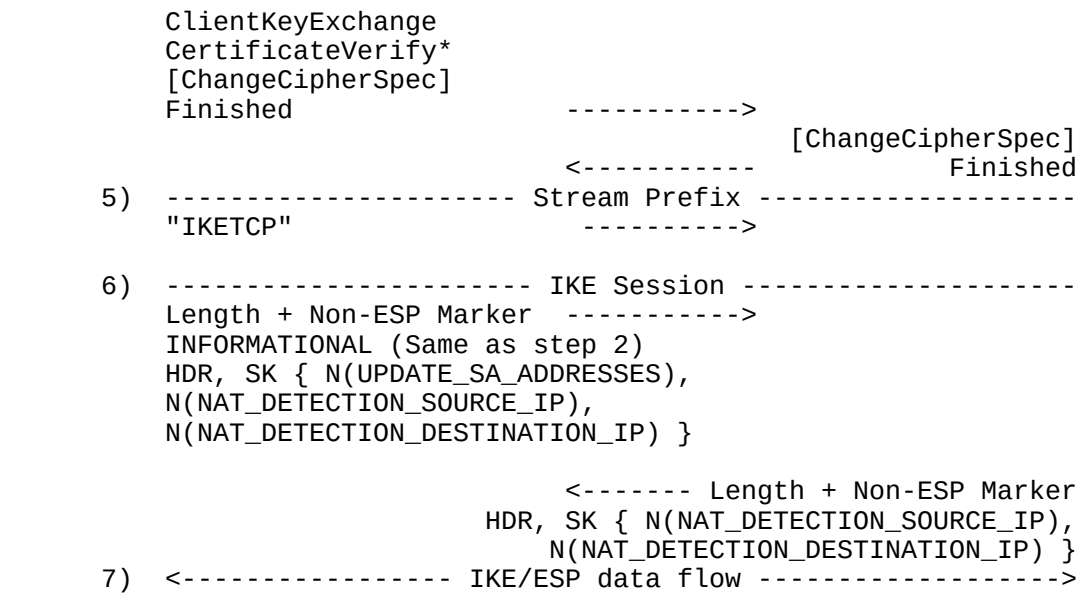


Figure 8

1. During the IKE_SA_INIT exchange, the client and server exchange MOBIKE_SUPPORTED notify payloads to indicate support for MOBIKE.

2. The client changes its point of attachment to the network, and receives a new IP address. The client attempts to re-establish the IKE session using the UPDATE_SA_ADDRESSES notify payload, but the server does not respond because the network blocks UDP traffic.

3. The client brings up a TCP connection to the server in order to use TCP encapsulation.

4. The client initiates and TLS handshake with the server.

5. The client sends the Stream Prefix for TCP encapsulated IKE traffic Section 4.

6. The client sends the UPDATE_SA_ADDRESSES notify payload on the TCP encapsulated connection. Note that this IKE message is the same as the one sent over UDP in step 2, and should have the same message ID and contents.

7. The IKE and ESP packet flow can resume.

Authors' Addresses

Tommy Pauly
Apple Inc.
1 Infinite Loop
Cupertino, California 95014
US

Email: tpauly@apple.com

Samy Touati
Ericsson
2755 Augustine
Santa Clara, California 95054
US

Email: samy.touati@ericsson.com

Ravi Mantha
Cisco Systems
SEZ, Embassy Tech Village
Panathur, Bangalore 560 037
India

Email: ramantha@cisco.com

IPSECME
Internet-Draft
Intended status: Standards Track
Expires: December 22, 2017

D. Migault, Ed.
Ericsson
T. Guggemos, Ed.
LMU Munich
Y. Nir
Dell EMC

June 20, 2017

Implicit IV for Counter-based Ciphers in IPsec
draft-mglt-ipsecme-implicit-iv-04

Abstract

IPsec ESP sends an initialization vector (IV) or nonce in each packet, adding 8 or 16 octets. Some algorithms such as AES-GCM, AES-CCM, AES-CTR and ChaCha20-Poly1305 require a unique nonce but do not require an unpredictable nonce. When using such algorithms the packet counter value can be used to generate a nonce, saving 8 octets per packet. This document describes how to do this.

Status of This Memo

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

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

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

This Internet-Draft will expire on December 22, 2017.

Copyright Notice

Table of Contents

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

Counter-based AES modes of operation such as AES-CTR ([RFC3686]), AES-CCM ([RFC4309]), and AES-GCM ([RFC4106]) require the specification of an nonce for each ESP packet. The same applies for ChaCha20-Poly1305 ([RFC7634]. Currently this nonce is sent in each ESP packet ([RFC4303]). This practice is designated in this document as "explicit nonce".

In some context, such as IoT, it may be preferable to avoid carrying the extra bytes associated to the IV and instead generate it locally on each peer. The local generation of the nonce is designated in this document as "implicit IV".

The size of this nonce depends on the specific algorithm, but all of the algorithms mentioned above take an 8-octet nonce.

This document defines how to compute the nonce locally when it is implicit. It also specifies how peers agree with the Internet Key Exchange version 2 (IKEv2 - [RFC7296]) on using an implicit IV versus an explicit IV.

This document limits its scope to the algorithms mentioned above. Other algorithms with similar properties may later be defined to use this extension.

This document does not consider AES-CBC ([RFC3602]) as AES-CBC requires the IV to be unpredictable. Deriving it directly from the packet counter as described below is insecure as mentioned in Security Consideration of [RFC3602] and has led to real world chosen plain-text attack such as BEAST [BEAST].

3. Terminology

- o IoT: Internet of Things.
- o IV: Initialization Vector.
- o Nonce: a fixed-size octet string used only once. This is similar to IV, except that in common usage there is no implication of non-predictability.

4. Implicit IV

With the algorithms listed in Section 2, the 8 byte nonce MUST NOT repeat. The binding between a ESP packet and its nonce is provided using the Sequence Number or the Extended Sequence Number. Figure 1 and Figure 2 represent the IV with a regular 4-byte Sequence Number and with an 8-byte Extended Sequence Number respectively.

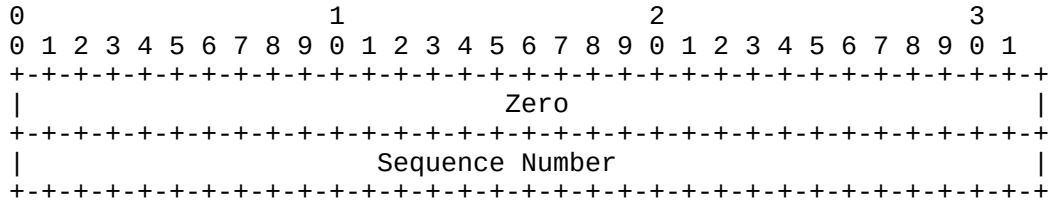


Figure 1: Implicit IV with a 4 byte Sequence Number

- o Sequence Number: the 4 byte Sequence Number carried in the ESP packet.
- o Zero: a 4 byte array with all bits set to zero.

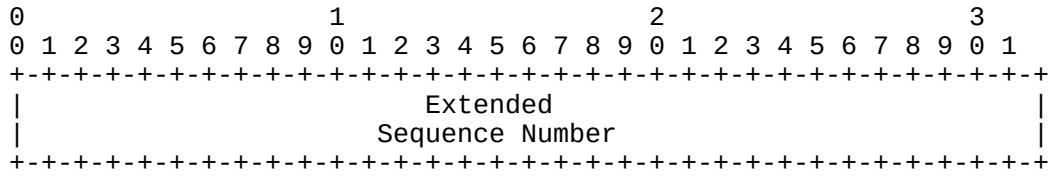


Figure 2: Implicit IV with an 8 byte Extended Sequence Number

- o Extended Sequence Number: the 8 byte Extended Sequence Number of the Security Association. The 4 byte low order bytes are carried in the ESP packet.

5. Initiator Behavior

An initiator supporting this feature SHOULD propose implicit IV for all relevant algorithms. To facilitate backward compatibility with non-supporting peers the initiator SHOULD also include those same algorithms without IIV. This may require extra transforms.

6. Responder Behavior

The rules of SA payload processing ensure that the responder will never send an SA payload containing the IIV indicator to an initiator that does not support IIV.

7. Security Consideration

Nonce generation for these algorithms has not been explicitly defined. It has been left to the implementation as long as certain security requirements are met. This document provides an explicit and normative way to generate IVs. The mechanism described in this document meets the IV security requirements of all relevant algorithms.

As the IV MUST NOT repeat for one SPI when Counter-Mode ciphers are used, Implicit IV as described in this document MUST NOT be used in setups with the chance that the Sequence Number overlaps for one SPI. Multicast as described in [RFC5374], [RFC6407] and [I-D.yeung-g-ikev2] is a prominent example, where many senders share one secret and thus one SPI. Section 3.5 of [RFC6407] explains how repetition MAY BE prevented by using a prefix for each group member, which could be prefixed to the Sequence Number. Otherwise, Implicit IV MUST NOT be used in multicast scenarios.

8. IANA Considerations

AES-CTR, AES-CCM, AES-GCM and ChaCha20-Poly1305 are likely to implement the implicit IV described in this document. This section limits assignment of new code points to the recommended suites provided in [I-D.ietf-ipsecme-rfc4307bis] and [I-D.ietf-ipsecme-rfc7321bis], thus the new Transform Type 1 - Encryption Algorithm Transform IDs are as defined below:

- ENCR_AES-CCM_8_IIV

- ENCR_AES-GCM_16_IIV

- ENCR_CHACHA20-POLY1305_IIV

9. References

9.1. Normative References

[I-D.ietf-ipsecme-rfc4307bis]
Nir, Y., Kivinen, T., Wouters, P., and D. Migault, "Algorithm Implementation Requirements and Usage Guidance for IKEv2", draft-ietf-ipsecme-rfc4307bis-18 (work in progress), March 2017.

[I-D.ietf-ipsecme-rfc7321bis]
Wouters, P., Migault, D., Mattsson, J., Nir, Y., and T. Kivinen, "Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)", draft-ietf-ipsecme-rfc7321bis-06 (work in progress), June 2017.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3602] Frankel, S., Glenn, R., and S. Kelly, "The AES-CBC Cipher Algorithm and Its Use with IPsec", RFC 3602, DOI 10.17487/RFC3602, September 2003, <http://www.rfc-editor.org/info/rfc3602>.

[RFC3686] Housley, R., "Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)", RFC 3686, DOI 10.17487/RFC3686, January 2004, <http://www.rfc-editor.org/info/rfc3686>.

[RFC4106] Viega, J. and D. McGrew, "The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)", RFC 4106, DOI 10.17487/RFC4106, June 2005, <http://www.rfc-editor.org/info/rfc4106>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <http://www.rfc-editor.org/info/rfc4303>.

[RFC4309] Housley, R., "Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)", RFC 4309, DOI 10.17487/RFC4309, December 2005, <http://www.rfc-editor.org/info/rfc4309>.

[RFC5374] Weis, B., Gross, G., and D. Ignjatic, "Multicast Extensions to the Security Architecture for the Internet Protocol", RFC 5374, DOI 10.17487/RFC5374, November 2008, <http://www.rfc-editor.org/info/rfc5374>.

[RFC6407] Weis, B., Rowles, S., and T. Hardjono, "The Group Domain of Interpretation", RFC 6407, DOI 10.17487/RFC6407, October 2011, <http://www.rfc-editor.org/info/rfc6407>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <http://www.rfc-editor.org/info/rfc7296>.

[RFC7634] Nir, Y., "ChaCha20, Poly1305, and Their Use in the Internet Key Exchange Protocol (IKE) and IPsec", RFC 7634, DOI 10.17487/RFC7634, August 2015, <http://www.rfc-editor.org/info/rfc7634>.

9.2. Informational References

[BEAST] Thai, T. and J. Juliano, "Here Come The xor Ninjas", , May 2011, <https://www.researchgate.net/publication/266529975_Here_Come_The_Ninjas>.

[I-D.yeung-g-ikev2] Weis, B., Nir, Y., and V. Smyslov, "Group Key Management using IKEv2", draft-yeung-g-ikev2-11 (work in progress), March 2017.

Authors' Addresses

Daniel Migault (editor)
Ericsson
8400 boulevard Decarie
Montreal, QC H4P 2N2
Canada

Email: daniel.migault@ericsson.com

Tobias Guggemos (editor)
LMU Munich
Oettingenstr. 67
80538 Munich, Bavaria
Germany

Email: guggemos@mnm-team.org
URI: http://mnm-team.org/~guggemos

Yoav Nir
Dell EMC
9 Andrei Sakharov St
Haifa 3190500
Israel

Email: ynir.ietf@gmail.com

Light-Weight Implementation Guidance (lwig)
Internet-Draft
Intended status: Informational
Expires: November 16, 2017

D. Migault
Ericsson
T. Guggemos
LMU Munich

May 15, 2017

Minimal ESP
draft-mglt-lwig-minimal-esp-05

Abstract

This document describes a minimal implementation of the IP Encapsulation Security Payload (ESP) described in RFC 4303. Its purpose is to enable implementation of ESP with a minimal set of options that makes the minimal implementation compatible with ESP as described in RFC 4303. A minimal version of ESP is not intended to become a replacement of the RFC 4303 ESP, but instead to enable a limited implementation to interoperate with implementations of RFC 4303 ESP.

This document describes what is required from RFC 4303 ESP as well as various ways to optimize compliance with RFC 4303 ESP.

This document does not update or modify RFC 4303, but provides a compact description of how to implement the minimal version of the protocol. If this document and RFC 4303 conflicts then RFC 4303 is the authoritative description.

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 November 16, 2017.

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

ESP [RFC4303] is part of the IPsec suite protocol [RFC4301] . It is used to provide confidentiality, data origin authentication, connectionless integrity, an anti-replay service (a form of partial sequence integrity) and limited traffic flow confidentiality.

Figure 1 describes an ESP Packet. Currently ESP is implemented in the kernel of major multi purpose Operating Systems (OS). The ESP and IPsec stack implemented is usually complete to fit multiple purpose usage of these OS. Completeness of the IPsec stack as well as multi purpose of these OS is often performed at the expense of resources, or a lack of performance, and so devices especially constraint devices like sensors have developed their own specific and task specific OS. This document provides a minimal ESP implementation guideline so these devices can implement ESP and benefit from IPsec.

For each field of the ESP packet represented in Figure 1 this document provides recommendations and guidance for minimal implementations. The primary purpose of Minimal ESP is to remain interoperable with other nodes implementing RFC 4303 ESP, while limiting the standard complexity of the implementation.

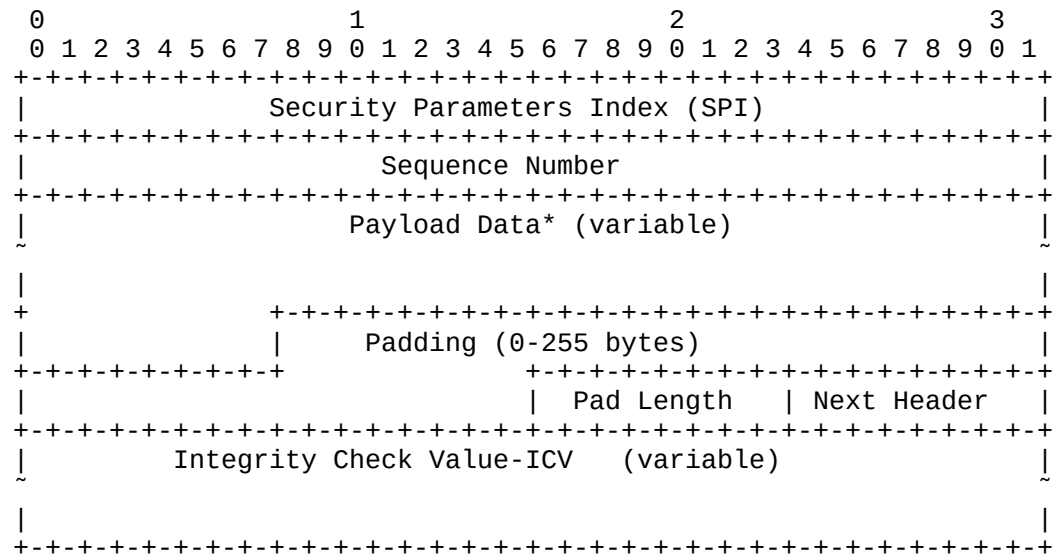


Figure 1: ESP Packet Description

3. Security Parameter Index (SPI) (32 bit)

According to the [RFC4303], the SPI is a mandatory 32 bits field and is not allowed to be removed.

The SPI has a local significance to index the Security Association (SA). From [RFC4301] section 4.1, nodes supporting only unicast communications can index their SA only using the SPI. On the other hand, nodes supporting multicast communications must also use the IP addresses and thus SA lookup needs to be performed using the longest match.

For nodes supporting only unicast communications, it is RECOMMENDED to index SA with the SPI only. Some other local constraints on the node may require a combination of the SPI as well as other parameters to index the SA.

It is RECOMMENDED to randomly generate the SPI indexing each inbound session. A random generation provides a stateless way to generate the SPIs, while keeping the probability of collision between SPIs relatively low. In case of collision, the SPI is simply re-generated.

However, for some constraint nodes, generating a random SPI may consume to much resource, in which case SPI can be generated using

predictable functions or even a fix value. In fact, the SPI does not need to the SPI does not need to be random.

When a constraint node uses fix value for SPIs, it imposes some limitations on the number of inbound SA. This limitation can be alleviate by how the SA look up is performed. When fix SPI are used, it is RECOMMENDED the constraint node has as many SPI values as ESP session per host IP address, and that lookup includes the IP addresses.

Note that SPI value is used only for inbound traffic, as such the SPI negotiated with IKEv2 [RFC7296] or [RFC7815] by a peer, is the value used by the remote peer when its sends traffic.

The use of fix SPI should not be considered as a way to avoid strong random generators. Such generator will be required in order to provide strong cryptographic protection. Instead, the use of a fix SPI should only considered as a way to overcome the resource limitations of the node, when this is feasible.

The use of a limited number of fix SPI also come with security or privacy drawbacks. Typically, a passive attacker may derive information such as the number of constraint devices connecting the remote peer, and in conjunction with data rate, the attacker may eventually determine the application the constraint device is associated to. In addition, if the fix value SPI is fixed by a manufacturer or by some software application, the SPI may leak in an obvious way the type of sensor, the application involved or the model of the constraint device. As a result, the use of a unpredictable SPI is preferred to provide better privacy.

As far as security is concerned, revealing the type of application or model of the constraint device could be used to identify the vulnerabilities the constraint device is subject to. This is especially sensitive for constraint device where patches or software updates will be challenging to operate. As a result, these devices may remain vulnerable for relatively long period. In addition, predictable SPI enable an attacker to forge packets with a valid SPI. Such packet will not be rejected due to an SPI mismatch, but instead after the signature check which requires more resource and thus make DoS more efficient, especially for devices powered by batteries.

Values 0-255 SHOULD NOT be used. Values 1-255 are reserved and 0 is only allowed to be used internal and it MUST NOT be send on the wire.

[RFC4303] mentions :

:: "The SPI is an arbitrary 32-bit value that is used by a receiver to identify the SA to which an incoming packet is bound. The SPI field is mandatory. [...]"

:: "For a unicast SA, the SPI can be used by itself to specify an SA, or it may be used in conjunction with the IPsec protocol type (in this case ESP). Because the SPI value is generated by the receiver for a unicast SA, whether the value is sufficient to identify an SA by itself or whether it must be used in conjunction with the IPsec protocol value is a local matter. This mechanism for mapping inbound traffic to unicast SAs MUST be supported by all ESP implementations."

4. Sequence Number(SN) (32 bit)

According to [RFC4303], the sequence number is a mandatory 32 bits field in the packet.

The SN is set by the sender so the receiver can implement anti-replay protection. The SN is derived from any strictly increasing function that guarantees: if packet B is sent after packet A, then SN of packet B is strictly greater then the SN of packet A.

In IoT, constraint devices are expected to establish communication with specific devices, like a specific gateway, or nodes similar to them. As a result, the sender may know whereas the receiver implements anti-replay protection or not. Even though the sender may know the receiver does not implement anti replay protection, the sender MUST implement a always increasing function to generate the SN.

Usually, SN is generated by incrementing a counter for each packet sent. A constraint device may avoid maintaining this context. If the device has a clock, it may use the time indicated by the clock has a SN. This guarantees a strictly increasing function, and avoid storing any additional values or context related to the SN. When the use of a clock is considered, one should take care that packets associated to a given SA are not sent with the same time value.

[RFC4303] mentions :

:: "This unsigned 32-bit field contains a counter value that increases by one for each packet sent, i.e., a per-SA packet sequence number. For a unicast SA or a single-sender multicast SA, the sender MUST increment this field for every transmitted packet. Sharing an SA among multiple senders is permitted, though generally not recommended. [...] The field is mandatory and MUST

always be present even if the receiver does not elect to enable the anti-replay service for a specific SA."

5. Padding

The purpose of padding is to respect the 32 bit alignment of ESP. ESP MUST have at least one padding byte Pad Length that indicates the padding length. ESP padding bytes are generated by a succession of unsigned bytes starting with 1, 2, 3 with the last byte set to Pad Length, where Pad Length designates the length of the padding bytes.

Checking the padding structure is not mandatory, so the constraint device may not proceed to such checks, however, in order to interoperate with existing ESP implementations, it MUST build the padding bytes as recommended by ESP.

In some situation the padding bytes may take a fix value. This would typically be the case when the Data Payload is of fix size.

[RFC4303] mentions :

:: "If Padding bytes are needed but the encryption algorithm does not specify the padding contents, then the following default processing MUST be used. The Padding bytes are initialized with a series of (unsigned, 1-byte) integer values. The first padding byte appended to the plaintext is numbered 1, with subsequent padding bytes making up a monotonically increasing sequence: 1, 2, 3, ... When this padding scheme is employed, the receiver SHOULD inspect the Padding field. (This scheme was selected because of its relative simplicity, ease of implementation in hardware, and because it offers limited protection against certain forms of "cut and paste" attacks in the absence of other integrity measures, if the receiver checks the padding values upon decryption.)"

ESP [RFC4303] also provides Traffic Flow Confidentiality (TFC) as a way to perform padding to hide traffic characteristics, which differs from respecting a 32 bit alignment. TFC is not mandatory and MUST be negotiated with the SA management protocol. As a result, TFC is not expected to be supported by a minimal ESP implementation. On the other hand, disabling TFC should be carefully measured and understood as it exposes the node to traffic shaping. This could expose the application as well as the devices used to a passive monitoring attacker. Such information could be used by the attacker in case a vulnerability is disclosed on the specific device. In addition, some application use - such as health applications - may also reveal important privacy oriented informations.

Some constraint nodes that have limited battery life time may also prefer avoiding sending extra padding bytes. However the same nodes may also be very specific to an application and device. As a result, they are also likely to be the main target for traffic shaping. In most cases, the payload carried by these nodes is quite small, and the standard padding mechanism may also be used as an alternative to TFC, with a sufficient trade off between the require energy to send additional payload and the exposure to traffic shaping attacks.

6. Next Header (8 bit)

According to [RFC4303], the Next Header is a mandatory 8 bits field in the packet. Next header is intended to specify the data contained in the payload as well as dummy packet. In addition, the Next Header may also carry an indication on how to process the packet [I-D.nikander-esp-beet-mode].

The ability to generate and receive dummy packet is required by [RFC4303]. For interoperability, it is RECOMMENDED a minimal ESP implementation discards dummy packets. Note that such recommendation only applies for nodes receiving packets, and that nodes designed to only send data may not implement this capability.

As the generation of dummy packets is subject to local management and based on a per-SA basis, a minimal ESP implementation may not generate such dummy packet. More especially, in constraint environment sending dummy packets may have too much impact on the device life time, and so may be avoided. On the other hand, constraint nodes may be dedicated to specific applications, in which case, traffic pattern may expose the application or the type of node. For these nodes, not sending dummy packet may have some privacy implication that needs to be measured.

In some cases, devices are dedicated to a single application or a single transport protocol, in which case, the Next Header has a fix value.

Specific processing indications have not been standardized yet [I-D.nikander-esp-beet-mode] and is expected to result from an agreement between the peers. As a result, it is not expected to be part of a minimal implementation of ESP.

[RFC4303] mentions :

> :: "The Next Header is a mandatory, 8-bit field that identifies the type of data contained in the Payload Data field, e.g., an IPv4 or IPv6 packet, or a next layer header and data. [...] the protocol value 59 (which means "no next header") MUST be used to

designate a "dummy" packet. A transmitter MUST be capable of generating dummy packets marked with this value in the next protocol field, and a receiver MUST be prepared to discard such packets, without indicating an error."

7. ICV

The ICV depends on the crypto-suite used. Currently recommended [I-D.ietf-ipsecme-rfc7321bis] only recommend crypto-suites with an ICV which makes the ICV a mandatory field.

As detailed in Section 8 we recommend to use authentication, the ICV field is expected to be present that is to say with a size different from zero. This makes it a mandatory field which size is defined by the security recommendations only.

[RFC4303] mentions :

:: "The Integrity Check Value is a variable-length field computed over the ESP header, Payload, and ESP trailer fields. Implicit ESP trailer fields (integrity padding and high-order ESN bits, if applicable) are included in the ICV computation. The ICV field is optional. It is present only if the integrity service is selected and is provided by either a separate integrity algorithm or a combined mode algorithm that uses an ICV. The length of the field is specified by the integrity algorithm selected and associated with the SA. The integrity algorithm specification MUST specify the length of the ICV and the comparison rules and processing steps for validation."

8. Cryptographic Suites

The cryptographic suites implemented are an important component of ESP. The recommended suites to use are expect to evolve over time and implementer SHOULD follow the recommendations provided by [I-D.ietf-ipsecme-rfc7321bis] and updates. Recommendations are provided for standard nodes as well as constraint nodes.

This section lists some of the criteria that may be considered. The list is not expected to be exhaustive and may also evolve overtime. As a result, the list is provided as indicative:

1. Security: Security is the criteria that should be considered first when a selection of cipher suites is performed. The security of cipher suites is expected to evolve over time, and it is of primary importance to follow up-to-date security guidances and recommendations. The chosen cipher suites MUST NOT be known vulnerable or weak (see [I-D.ietf-ipsecme-rfc7321bis] for

outdated ciphers). ESP can be used to authenticate only or to encrypt the communication. In the later case, authenticated encryption must always be considered [I-D.ietf-ipsecme-rfc7321bis].

2. Interoperability: Interoperability considers the cipher suites shared with the other nodes. Note that it is not because a cipher suite is widely deployed that is secured. As a result, security SHOULD NOT be weaken for interoperability. [I-D.ietf-ipsecme-rfc7321bis] and successors consider the life cycle of cipher suites sufficiently long to provide interoperability. Constraint devices may have limited interoperability requirements which makes possible to reduces the number of cipher suites to implement.

3. Power Consumption and Cipher Suite Complexity: Complexity of the cipher suite or the energy associated to it are especially considered when devices have limited resources or are using some batteries, in which case the battery determines the life of the device. The choice of a cryptographic function may consider re-using specific libraries or to take advantage of hardware acceleration provided by the device. For example if the device benefits from AES hardware modules and uses AES-CTR, it may prefer AUTH_AES-XCBC for its authentication. In addition, some devices may also embed radio modules with hardware acceleration for AES-CCM, in which case, this mode may be preferred.

4. Power Consumption and Bandwidth Consumption: Similarly to the cipher suite complexity, reducing the payload sent, may significantly reduce the energy consumption of the device. As a result, cipher suites with low overhead may be considered. To reduce the overall payload size one may for example, one MAY consider:

    1. Use of counter-based ciphers without fixed block length (e.g. AES-CTR, or ChaCha20-Poly1305).

    2. Use of ciphers with capability of using implicit IVs [I-D.mglt-ipsecme-implicit-iv].

    3. Use of ciphers recommended for IoT [I-D.ietf-ipsecme-rfc7321bis].

    4. Avoid Padding by sending payload data which are aligned to the cipher block length -2 for the ESP trailer.

9. IANA Considerations

There are no IANA consideration for this document.

10. Security Considerations

Security considerations are those of [RFC4303]. In addition, this document provided security recommendations an guidances over the implementation choices for each fields.

11. Acknowledgment

The authors would like to thank Scott Fluhrer, Tero Kivinen, Valery Smyslov, Yoav Nir, Michael Richardson for their valuable comments.

12. References

12.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC3602] Frankel, S., Glenn, R., and S. Kelly, "The AES-CBC Cipher Algorithm and Its Use with IPsec", RFC 3602, DOI 10.17487/RFC3602, September 2003, <http://www.rfc-editor.org/info/rfc3602>.

[RFC3686] Housley, R., "Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)", RFC 3686, DOI 10.17487/RFC3686, January 2004, <http://www.rfc-editor.org/info/rfc3686>.

[RFC4106] Viega, J. and D. McGrew, "The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)", RFC 4106, DOI 10.17487/RFC4106, June 2005, <http://www.rfc-editor.org/info/rfc4106>.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <http://www.rfc-editor.org/info/rfc4301>.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, DOI 10.17487/RFC4303, December 2005, <http://www.rfc-editor.org/info/rfc4303>.

[RFC4309] Housley, R., "Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)", RFC 4309, DOI 10.17487/RFC4309, December 2005, <http://www.rfc-editor.org/info/rfc4309>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <http://www.rfc-editor.org/info/rfc7296>.

[RFC7815] Kivinen, T., "Minimal Internet Key Exchange Version 2 (IKEv2) Initiator Implementation", RFC 7815, DOI 10.17487/RFC7815, March 2016, <http://www.rfc-editor.org/info/rfc7815>.

12.2. Informative References

[I-D.ietf-ipsecme-rfc7321bis]
Migault, D., Mattsson, J., Wouters, P., Nir, Y., and T. Kivinen, "Cryptographic Algorithm Implementation Requirements and Usage Guidance for Encapsulating Security Payload (ESP) and Authentication Header (AH)", draft-ietf-ipsecme-rfc7321bis-05 (work in progress), February 2017.

[I-D.mglt-ipsecme-implicit-iv]
Migault, D., Guggemos, T., and Y. Nir, "Implicit IV for Counter-based Ciphers in IPsec", draft-mglt-ipsecme-implicit-iv-02 (work in progress), November 2016.

[I-D.nikander-esp-beet-mode]
NikanderMelen, PJ., "A Bound End-to-End Tunnel (BEET) mode for ESP", draft-nikander-esp-beet-mode-09 (work in progress), February 2017.

Appendix A. Document Change Log

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

-00: First version published.

-01: Clarified description

-02: Clarified description

Authors' Addresses

Daniel Migault
Ericsson
8400 boulevard Decarie
Montreal, QC H4P 2N2
Canada

Email: daniel.migault@ericsson.com

Tobias Guggemos
LMU Munich
MNM-Team
Oettingenstr. 67
80538 Munich, Bavaria
Germany

Email: guggemos@mnm-team.org

Network
Internet-Draft
Intended status: Standards Track
Expires: March 25, 2017

T. Pauly
Apple Inc.
P. Wouters
Red Hat

September 21, 2016

Split DNS Configuration for IKEv2
draft-pauly-ipsecme-split-dns-02

Abstract

This document defines two Configuration Payload Attribute Types for the IKEv2 protocol that add support for private DNS domains. These domains should be resolved using DNS servers reachable through an IPsec connection, while leaving all other DNS resolution unchanged. This approach of resolving a subset of domains using non-public DNS servers is referred to as "Split DNS".

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 March 25, 2017.

Copyright Notice

Table of Contents

1. Introduction

Split DNS is a common configuration for secure tunnels, such as Virtual Private Networks in which host machines private to an organization can only be resolved using internal DNS resolvers [RFC2775]. In such configurations, it is often desirable to only resolve hosts within a set of private domains using the tunnel, while letting resolutions for public hosts be handled by a device's default DNS configuration.

The Internet Key Exchange protocol version 2 [RFC7296] negotiates configuration parameters using Configuration Payload Attribute Types. This document defines two Configuration Payload Attribute Types that add support for trusted Split DNS domains.

The INTERNAL_DNS_DOMAIN attribute type is used to convey one or more DNS domains that should be resolved only using the provided DNS nameserver IP addresses, causing these requests to use the IPsec connection.

The INTERNAL_DNSSEC_TA attribute type is used to convey DNSSEC trust anchors for those domains.

When only a subset of traffic is routed into a private network using an IPsec SA, these Configuration Payload options can be used to define which private domains should be resolved through the IPsec connection without affecting the client's global DNS resolution.

For the purposes of this document, DNS resolution servers accessible through an IPsec connection will be referred to as "internal DNS servers", and other DNS servers will be referred to as "external DNS servers".

A client using these configuration payloads will be able to request and receive Split DNS configurations using the INTERNAL_DNS_DOMAIN and INTERNAL_DNSSEC_TA configuration attributes. The client device can use the internal DNS server(s) for any DNS queries within the assigned domains, while routing other DNS queries to its regular external DNS server.

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].

2. Background

Split DNS is a common configuration for enterprise VPN deployments, in which only one or a few private DNS domains are accessible and resolvable via an IPsec based VPN connection.

Other tunnel-establishment protocols already support the assignment of Split DNS domains. For example, there are proprietary extensions to IKEv1 that allow a server to assign Split DNS domains to a client. However, the IKEv2 standard does not include a method to configure this option. This document defines a standard way to negotiate this option for IKEv2.

3. Protocol Exchange

In order to negotiate which domains are considered internal to an IKEv2 tunnel, initiators indicate support for Split DNS in their CFG_REQUEST payloads, and responders assign internal domains (and DNSSEC trust anchors) in their CFG_REPLY payloads. When Split DNS has been negotiated, the existing DNS server configuration attributes will be interpreted as internal DNS servers that can resolve hostnames within the internal domains.

3.1. Configuration Request

To indicate support for Split DNS, an initiator sending a CFG_REQUEST payload MAY include one or more INTERNAL_DNS_DOMAIN attributes as defined in Section 4. If an INTERNAL_DNS_DOMAIN attribute is included in the CFG_REQUEST, the initiator SHOULD also include one or both of the INTERNAL_IP4_DNS and INTERNAL_IP6_DNS attributes in its CFG_REQUEST.

If the length of the INTERNAL_DNS_DOMAIN attribute is zero, then the initiator is requesting that the attribute be assigned without restricting the subdomains that it will accept.

If the length of the INTERNAL_DNS_DOMAIN is greater than zero, the value is a single DNS domain. The initiator is indicating that it will only allow this domain and any sub-domains within this domain to be resolved using the internal DNS servers. The list of INTERNAL_DNS_DOMAIN attributes in the CFG_REQUEST defines the full set of domains the intiator is willing to resolve using the internal DNS servers.

The absence of INTERNAL_DNS_DOMAIN attributes in the CFG_REQUEST payload indicates that the initiator does not support or is unwilling to accept Split DNS configuration.

To indicate support for DNSSEC, an initiator sending a CFG_REQUEST payload MAY include one or more INTERNAL_DNS_TA attributes as defined in Section 4. These payloads MUST immediately follow a INTERNAL_DNS_DOMAIN attribute, which binds the DNSSEC trust anchor request to the domain.

An initiator MAY convey its current DNSSEC trust anchors for the domain specified in the INTERNAL_DNS_DOMAIN attribute. If it does not wish to convey this information, it MUST use a length of 0.

The absence of INTERNAL_DNS_TA attributes in the CFG_REQUEST payload indicates that the initiator does not support or is unwilling to accept DNSSEC trust anchor configuration.

3.2. Configuration Reply

Responders MAY send one or more INTERNAL_DNS_DOMAIN attributes in their CFG_REPLY payload if the CFG_REQUEST contained at least one INTERNAL_DNS_DOMAIN attribute. If the CFG_REQUEST did not contain an INTERNAL_DNS_DOMAIN attribute, the responder MUST NOT include an INTERNAL_DNS_DOMAIN attribute in the CFG_REPLY. If an INTERNAL_DNS_DOMAIN attribute is included in the CFG_REPLY, the responder SHOULD also include one or both of the INTERNAL_IP4_DNS and

INTERNAL_IP6_DNS attributes in its CFG_REPLY. These DNS server configurations are necessary to define which servers should receive queries for hostnames in internal domains. If the CFG_REQUEST included an INTERNAL_DNS_DOMAIN attribute, but the CFG_REPLY does not include an INTERNAL_DNS_DOMAIN attribute, the initiator should behave as if Split DNS configurations are not supported by the server.

Each INTERNAL_DNS_DOMAIN represents a domain that the DNS servers address listed in INTERNAL_IP4_DNS and INTERNAL_IP6_DNS can resolve.

If the CFG_REQUEST included INTERNAL_DNS_DOMAIN attributes with non-zero lengths, the CFG_REPLY MUST NOT assign any domains in its INTERNAL_DNS_DOMAIN attributes that are not contained within the requested domains. The initiator SHOULD ignore any domains beyond its requested list.

For each DNS domain specified in an INTERNAL_DNS_DOMAIN attribute, one or more INTERNAL_DNSSEC_TA attributes MAY be included by the responder. This attribute lists the corresponding DSSNEC trust anchor in the DNS wire format of a DS record as specified in [RFC4034]. The INTERNAL_DNSSEC_TA attribute MUST immediately follow the INTERNAL_DNS_DOMAIN attribute that it applies to.

3.3. Mapping DNS Servers to Domains

All DNS servers provided in the CFG_REPLY MUST support resolving hostnames within all INTERNAL_DNS_DOMAIN domains. In other words, the INTERNAL_DNS_DOMAIN attributes in a CFG_REPLY payload form a single list of Split DNS domains that applies to the entire list of INTERNAL_IP4_DNS and INTERNAL_IP6_DNS attributes.

3.4. Example Exchanges

3.4.1. Simple Case

In this example exchange, the initiator requests INTERNAL_IP4_DNS and INTERNAL_DNS_DOMAIN attributes in its CFG_REQUEST, but does not specify any value for either. This indicates that it supports Split DNS, but has no preference for which DNS requests should be routed through the tunnel.

The responder replies with two DNS server addresses, and one internal domain, "example.com".

Any subsequent DNS queries from the initiator for domains such as "www.example.com" should use 198.51.100.2 or 198.51.100.4 to resolve.

```
CP(CFG_REQUEST) =
  INTERNAL_IP4_ADDRESS()
  INTERNAL_IP4_DNS()
  INTERNAL_DNS_DOMAIN()

CP(CFG_REPLY) =
  INTERNAL_IP4_ADDRESS(198.51.100.234)
  INTERNAL_IP4_DNS(198.51.100.2)
  INTERNAL_IP4_DNS(198.51.100.4)
  INTERNAL_DNS_DOMAIN(example.com)
```

3.4.2. Requesting Limited Domains

In this example exchange, the initiator requests INTERNAL_IP4_DNS and INTERNAL_DNS_DOMAIN attributes in its CFG_REQUEST, specifically requesting only "example.com" and "other.com". The responder replies with two DNS server addresses, 198.51.100.2 and 198.51.100.4, and two domains, "example.com" and "city.other.com". Note that one of the domains in the CFG_REPLY, "city.other.com", is a subset of the requested domain, "other.com". This indicates that hosts within "other.com" that are not within "city.other.com" should be resolved using an external DNS server. The CFG_REPLY would not be allowed to respond with "com" or "example.net", however, since these were contained within the limited set of requested domains.

Any subsequent DNS queries from the initiator for domains such as "www.example.com" or "city.other.com" should use 198.51.100.2 or 198.51.100.4 to resolve.

```
CP(CFG_REQUEST) =
  INTERNAL_IP4_ADDRESS()
  INTERNAL_IP4_DNS()
  INTERNAL_DNS_DOMAIN(example.com)
  INTERNAL_DNS_DOMAIN(other.com)

CP(CFG_REPLY) =
  INTERNAL_IP4_ADDRESS(198.51.100.234)
  INTERNAL_IP4_DNS(198.51.100.2)
  INTERNAL_IP4_DNS(198.51.100.4)
  INTERNAL_DNS_DOMAIN(example.com)
  INTERNAL_DNS_DOMAIN(city.other.com)
```

3.4.3. Requesting Domains and DNSSEC trust anchors

In this example exchange, the initiator requests INTERNAL_IP4_DNS, INTERNAL_DNS_DOMAIN and INTERNAL_DNS_TA attributess in its CFG_REQUEST

Any subsequent DNS queries from the initiator for domains such as "www.example.com" or "city.other.com" would be DNSSEC validated using the DNSSEC trust anchor received in the CFG_REPLY

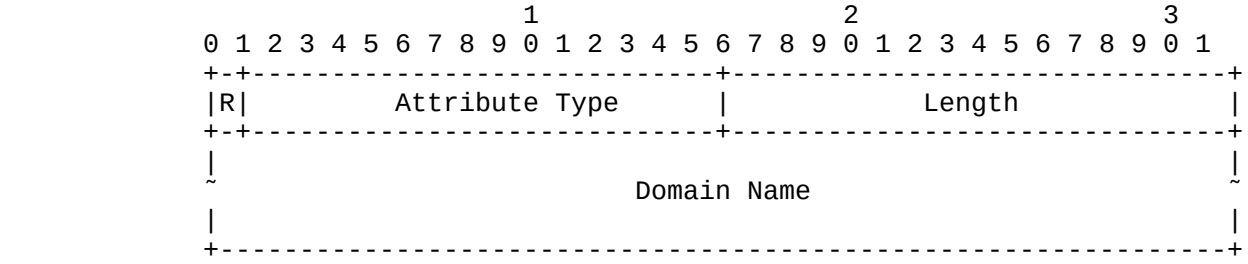
In this example, the initiator has no existing DNSSEC trust anchors would the requested domain. the "example.com" dommain has DNSSEC trust anchors that are returned, while the "other.com" domain has no DNSSEC trust anchors

```
CP(CFG_REQUEST) =
  INTERNAL_IP4_ADDRESS()
  INTERNAL_IP4_DNS()
  INTERNAL_DNS_DOMAIN(example.com)
  INTERNAL_DNS_TA()
  INTERNAL_DNS_DOMAIN(other.com)
  INTERNAL_DNS_TA()

CP(CFG_REPLY) =
  INTERNAL_IP4_ADDRESS(198.51.100.234)
  INTERNAL_IP4_DNS(198.51.100.2)
  INTERNAL_IP4_DNS(198.51.100.4)
  INTERNAL_DNS_DOMAIN(example.com)
  INTERNAL_DNS_TA(43547,8,1,B6225AB2CC613E0DCA7962BDC2342EA4F1B56083)
  INTERNAL_DNS_TA(31406,8,2,F78CF3344F72137235098ECBBD08947C2C90....)
  INTERNAL_DNS_DOMAIN(city.other.com)
```

4. Payload Formats

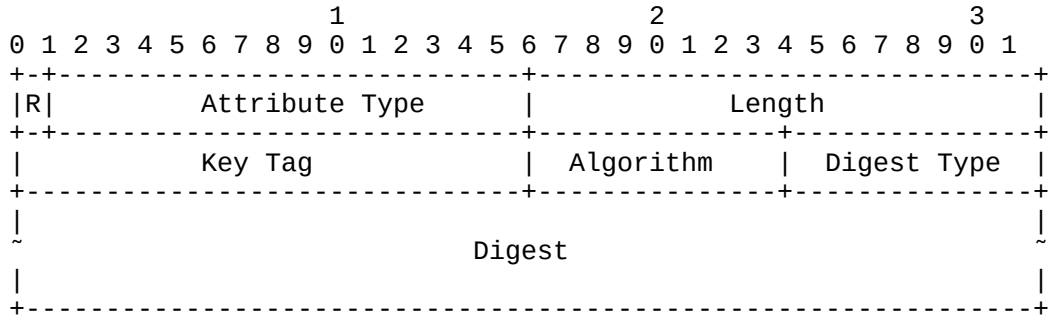
4.1. INTERNAL_DNS_DOMAIN Configuration Attribute Type



- Reserved (1 bit) - Defined in IKEv2 RFC [RFC7296].
- Attribute Type (15 bits) [TBD IANA] - INTERNAL_DNS_DOMAIN.
- Length (2 octets, unsigned integer) - Length of domain name.
- Domain Name (0 or more octets) - A domain or subdomain used for Split DNS rules, such as example.com. This is a string of ASCII

characters with labels separated by dots, with no trailing dot, using IDNA [RFC5890] for non-ASCII DNS domains. The value is NOT null-terminated.

4.2. INTERNAL_DNSSEC_TA Configuration Attribute



- Reserved (1 bit) - Defined in IKEv2 RFC [RFC7296].
- Attribute Type (15 bits) [TBD IANA] - INTERNAL_DNSSEC_TA.
- Length (2 octets, unsigned integer) - Length of DNSSEC Trust Anchor data.
- Key Tag value (0 or 2 octets, unsigned integer) - Key Tag as specified in [RFC4034] Section 5.1
- DNSKEY algorithm (0 or 1 octet) - Value from the IANA DNS Security Algorithm Numbers Registry
- DS algorithm (0 or 1 octet) - Value from the IANA Delegation Signer (DS) Resource Record (RR) Type Digest Algorithms Registry
- Digest (0 or more octets) - The raw digest as specified in [RFC4034] Section 5.1

5. Split DNS Usage Guidelines

If a CFG_REPLY payload contains no INTERNAL_DNS_DOMAIN attributes, the client MAY use the provided INTERNAL_IP4_DNS or INTERNAL_IP6_DNS servers as the default DNS server(s) for all queries.

For each INTERNAL_DNS_DOMAIN entry in a CFG_REPLY payload, the client SHOULD use the provided INTERNAL_IP4_DNS or INTERNAL_IP6_DNS DNS servers as the only resolvers for the listed domains and its sub-domains and it SHOULD NOT attempt to resolve the provided DNS domains using its external DNS servers.

If the initiator host is configured to block DNS answers containing IP addresses from special IP address ranges such as those of [RFC1918], the initiator SHOULD allow the DNS domains listed in the INTERNAL_DNS_DOMAIN attributes to contain these IP addresses.

If a CFG_REPLY contains one or more INTERNAL_DNS_DOMAIN attributes, the client SHOULD configure its DNS resolver to resolve those domains and all their subdomains using only the DNS resolver(s) listed in that CFG_REPLY message. If those resolvers fail, those names SHOULD NOT be resolved using any other DNS resolvers. All other domain names MUST be resolved using some other external DNS resolver(s), configured independently, and SHOULD NOT be sent to the internal DNS resolver(s) listed in that CFG_REPLY message. For example, if the INTERNAL_DNS_DOMAIN attribute specifies "example.com", then "example.com", "www.example.com" and "mail.eng.example.com" MUST be resolved using the internal DNS resolver(s), but "anotherexample.com" and "ample.com" MUST be resolved using the system’s external DNS resolver(s).

An initiator SHOULD ignore INTERNAL_DNS_DOMAIN attributes containing domains that are designated Special Use Domain Names in [RFC6761], such as "local", "localhost", "invalid", etc. Although it may explicitly wish to support some Special Use Domain Names.

When an IPsec connection is terminated, the DNS forwarding must be unconfigured. The DNS forwarding itself MUST be be deleted. All cached data of the INTERNAL_DNS_DOMAIN provided DNS domainis MUST be flushed. This includes negative cache entries. Obtained DNSSEC trust anchors MUST be removed from the list of trust anchors. The outstanding DNS request queue MAY be cleared.

A domain that is served via INTERNAL_DNS_DOMAIN MUST NOT have indirect references to DNS records that point to other Split DNS domains that are not served via INTERNAL_DNS_DOMAIN attributes. Indirect reference RRtypes include CNAME, DNAME, MX and SRV RR’s.

INTERNAL_DNS_DOMAIN and INTERNAL_DNSSEC_TA attributes SHOULD only be used on split tunnel configurations where only a subset of traffic is routed into a private remote network using the IPsec connection. If all traffic is routed over the IPsec connection, the existing global INTERNAL_IP4_DNS and INTERNAL_IP6_DNS can be used without creating specific DNS exemptions.

6. Security Considerations

The use of Split DNS configurations assigned by an IKEv2 responder is predicated on the trust established during IKE SA authentication. However, if IKEv2 is being negotiated with an anonymous or unknown

endpoint (such as for Opportunistic Security [RFC7435]), the initiator MUST ignore Split DNS configurations assigned by the responder.

If a host connected to an authenticated IKE peer is connecting to another IKE peer that attempts to claim the same domain via the INTERNAL_DNS_DOMAIN attribute, the IKE connection should be terminated.

If the IP address value of the received INTERNAL_IP4_DNS or INTERNAL_IP6_DNS attribute is not covered by the proposed IPsec connection, then the local DNS should not be reconfigured until a CREATE_CHILD Exchange is received that covers these IP addresses.

INTERNAL_DNSSEC_TA directives MUST immediately follow an INTERNAL_DNS_DOMAIN directive. As the INTERNAL_DNSSEC_TA format itself does not contain the domain name, it relies on the preceding INTERNAL_DNS_DOMAIN to provide the domain for which it specifies the trust anchor.

If the initiator is using DNSSEC validation for a domain in its public DNS view, and it requests and receives an INTERNAL_DNS_DOMAIN attribute without an INTERNAL_DNSSEC_TA, it will need to reconfigure its DNS resolver to allow for an insecure delegation. It SHOULD NOT accept insecure delegations for domains that are DNSSEC signed in the public DNS view, for which it has not explicitely requested such deletation by specifying the domain specifically using a INTERNAL_DNS_DOMAIN(domain) request.

7. IANA Considerations

This document defines two new IKEv2 Configuration Payload Attribute Types, which are allocated from the "IKEv2 Configuration Payload Attribute Types" namespace.

Value	Attribute Type	Multi-Valued	Length	Reference
[TBD]	INTERNAL_DNS_DOMAIN	YES	0 or more	[this document]
[TBD]	INTERNAL_DNSSEC_TA	YES	0 or more	[this document]

Figure 1

8. References

8.1. Normative References

[RFC1918] Rekhter, Y., Moskowitz, B., Karrenberg, D., de Groot, G., and E. Lear, "Address Allocation for Private Internets", BCP 5, RFC 1918, DOI 10.17487/RFC1918, February 1996, <http://www.rfc-editor.org/info/rfc1918>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <http://www.rfc-editor.org/info/rfc4034>.

[RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, DOI 10.17487/RFC5890, August 2010, <http://www.rfc-editor.org/info/rfc5890>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <http://www.rfc-editor.org/info/rfc7296>.

8.2. Informative References

[RFC2775] Carpenter, B., "Internet Transparency", RFC 2775, DOI 10.17487/RFC2775, February 2000, <http://www.rfc-editor.org/info/rfc2775>.

[RFC6761] Cheshire, S. and M. Krochmal, "Special-Use Domain Names", RFC 6761, DOI 10.17487/RFC6761, February 2013, <http://www.rfc-editor.org/info/rfc6761>.

[RFC7435] Dukhovni, V., "Opportunistic Security: Some Protection Most of the Time", RFC 7435, DOI 10.17487/RFC7435, December 2014, <http://www.rfc-editor.org/info/rfc7435>.

Authors' Addresses

Tommy Pauly
Apple Inc.
1 Infinite Loop
Cupertino, California 95014
US

Email: tpauly@apple.com

Paul Wouters
Red Hat

Email: pwouters@redhat.com