Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: January 2, 2019

C. Tjhai
M. Tomlinson
Post-Quantum
G. Bartlett
S. Fluhrer
Cisco Systems
D. Van Geest
ISARA Corporation
Z. Zhang
Onboard Security
O. Garcia-Morchon
Philips

July 1, 2018

Framework to Integrate Post-quantum Key Exchanges into Internet Key Exchange Protocol Version 2 (IKEv2) draft-tjhai-ipsecme-hybrid-qske-ikev2-02

Abstract

This document describes how to extend Internet Key Exchange Protocol Version 2 (IKEv2) so that the shared secret exchanged between peers has resistance against quantum computer attacks. The basic idea is to exchange one or more post-quantum key exchange payloads in conjunction with the existing (Elliptic Curve) Diffie-Hellman payload.

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 https://datatracker.ietf.org/drafts/current/.

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

This Internet-Draft will expire on January 2, 2019.

Table of Contents

1. Introduction

1.1. Problem Description

Internet Key Exchange Protocol (IKEv2) as specified in RFC 7296 [RFC7296] uses the Diffie-Hellman (DH) or Elliptic Curve Diffie-Hellman (ECDH) algorithm to establish a shared secret between an initiator and a responder. The security of the DH and ECDH algorithms relies on the difficulty to solve a discrete logarithm problem in multiplicative and elliptic curve groups respectively when

the order of the group parameter is large enough. While solving such a problem remains difficult with current computing power, it is believed that general purpose quantum computers will be able to solve this problem, implying that the security of IKEv2 is compromised. There are, however, a number of cryptosystems that are conjectured to be resistant against quantum computer attack. This family of cryptosystems are known as post-quantum cryptography (PQC). It is ometime also referred to as quantum-safe cryptography (QSC) or quantum-resistant cryptography (QRC).

1.2. Proposed Extension

This document describes a framework to integrate QSC for IKEv2, while maintaining backwards compatibility, to derive a set of IKE keys that have resistance to quantum computer attacks. Our framework allows the negotiation of one or more QSC algorithm to exchange data, in addition to the existing DH or ECDH key exchange data. We believe that the feature of using more than one post-quantum algorithm is important as many of these algorithms are relatively new and there may be a need to hedge the security risk with multiple key exchange data from several distinct QSC algorithms.

The secrets established from each key exchange are combined in a way such that should the post-quantum secrets not be present, the derived shared secret is equivalent to that of the standard IKEv2; on the other hand, a post-quantum shared secret is obtained if both classical and post-quantum key exchange data are present. This framework also applies to key exchanges in IKE Security Associations (SAs) for Encapsulating Security Payload (ESP) [ESP] or Authentication Header (AH) [AH], i.e. Child SAs, in order to provide a stronger guarantee of forward security.

Some post-quantum key exchange payloads may have size larger than the standard MTU size, and therefore there could be issues with fragmentation at IP layer. IKE does allow transmission over TCP where fragmentation is not an issue [RFC8229]; however, we believe that a UDP-based solution will be required too. IKE does have a mechanism to handle fragmentation within UDP [RFC7383], however that is only applicable to messages exchanged after the IKE_SA_INIT. To use this mechanism, we use the IKE_AUX exchange as outlined in [I-D.smyslov-ipsecme-ikev2-aux]. With this mechanism, we do an initial key exchange, using a smaller, possibly non-quantum resistant primitive, such as ECDH. Then, before we do the IKE_AUTH exchange, we perform one or more IKE_AUX exchanges, each of which includes a secondary key exchange. As the IKE_AUX exchange is encrypted, the IKE fragmentation protocol RFC7383 can be used. The IKE SK values will be updated after each exchange, and so the final IKE SK values

will depend on all the key exchanges, hence they are secure if any of the key exchanges are secure.

Note that readers should consider the approach in this document as providing a long term solution in upgrading the IKEv2 protocol to support post-quantum algorithms. A short term solution to make IKEv2 key exchange quantum secure is to use post-quantum pre-shared keys as discussed in [I-D.ietf-ipsecme-qr-ikev2].

1.3. Changes

Changes in this draft in each version iterations.

draft-tjhai-ipsecme-hybrid-qske-ikev2-01

- Use IKE_AUX to perform multiple key exchanges in succession.
- Handle fragmentation by keeping the first key exchange (a standard IKE_SA_INIT with a few extra notifies) small, and encrypting the rest of the key exchanges.
- Simplify the negotiation of the 'extra' key exchanges.

draft-tjhai-ipsecme-hybrid-qske-ikev2-00

- We added a feature to allow more than one post-quantum key exchange algorithms to be negotiated and used to exchange a post-quantum shared secret.
- Instead of relying on TCP encapsulation to deal with IP level fragmentation, we introduced a new key exchange payload that can be sent as multiple fragments within IKE_SA_INIT message.

1.4. Document organization

The remainder of this document is organized as follows. Section 2 summarizes design criteria. Section 3 describes how post-quantum key exchange is performed between two IKE peers and how keying materials are derived. The rationale behind the approach of this extension is described in Section 3. Section 4 discusses security considerations an lastly, Section 5 discusses IANA considerations for the name spaces introduced in this document.

The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD, SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this document, are to be interpreted as described in RFC 2119 [RFC2119].

2. Design criteria

The design of the proposed post-quantum IKEv2 is driven by the following criteria:

1) Need for post-quantum cryptography in IPsec. Quantum computers might become feasible in the next 5-10 years. If current Internet communications are monitored and recorded today (D), the communications could be decrypted as soon as a quantum-computer is available (e.g., year Q) if key negotiation only relies on non post-quantum primitives. This is a high threat for any information that must remain confidential for a long period of time T > Q-D. The need is obvious if we assume that Q is 2040, D is 2020, and T is 30 years. Such a value of T is typical in classified or healthcare data.

2) Hybrid. Currently, there does not exist a post-quantum key exchange that is trusted at the level that ECDH is trusted against conventional (non-quantum) adversaries. A hybrid approach allows introducing promising post-quantum candidates next to well-established primitives, since the overall security is at least as strong as each individual primitive.

3) Focus on quantum-resistant confidentiality. A passive attacker can eavesdrop on IPsec communication today and decrypt it once a quantum computer is available in the future. This is a very serious attack for which we do not have a solution. An attacker can only perform active attacks such as impersonation of the communicating peers once a quantum computer is available, sometime in the future. Thus, our design focuses on quantum-resistant confidentiality due to the urgency of this problem. This document does not address quantum-resistant authentication since it is less urgent at this stage.

4) Limit amount of exchanged data. The protocol design should be such that the amount of exchanged data, such as public-keys, is kept as small as possible even if initiator and responder need to agree on a hybrid group or multiple public-keys need to be exchanged.

5) Future proof. Any cryptographic algorithm could be potentially broken in the future by currently unknown or impractical attacks: quantum computers are merely the most concrete example of this. The design does not categorize algorithms as "post-quantum" or "non post-quantum" and does not create assumptions about the properties of the algorithms, meaning that if algorithms with different properties become necessary in the

future, this framework can be used unchanged to facilitate migration to those algorithms.

6) Limited amount of changes. A key goal is to limit the number of changes required when enabling a post-quantum handshake. This ensures easier and quicker adoption in existing implementations.

7) Localized changes. Another key requirement is that changes to the protocol are limited in scope, in particular, limiting changes in the exchanged messages and in the state machine, so that they can be easily implemented.

8) Deterministic operation. This requirement means that the hybrid post-quantum exchange, and thus, the computed key, will be based on algorithms that both client and server wish to support.

9) Fragmentation support. Some PQC algorithms could be relatively bulky and they might require fragmentation. Thus, a design goal is the adaptation and adoption of an existing fragmentation method or the design of a new method that allows for the fragmentation of the key shares.

10) Backwards compatibility and interoperability. This is a fundamental requirement to ensure that hybrid post-quantum IKEv2 and a non-post-quantum IKEv2 implementations are interoperable.

11) FIPS compliance. IPsec is widely used in Federal Information Systems and FIPS certification is an important requirement. However, algorithms that are believed to be post-quantum are not FIPS compliant yet. Still, the goal is that the overall hybrid post-quantum IKEv2 design can be FIPS compliant.

3. The Framework of Hybrid Post-quantum Key Exchange

3.1. Overall design

This design assigns new group identifiers (Transform Type 4) to the various post-quantum key exchanges (which will be defined later). We specifically do not make a distinction between classical (DH and ECDH) and post-quantum key exchanges, nor post-quantum algorithms which are true key exchanges versus post-quantum algorithms that act as key transport mechanisms; all are treated equivalently by the protocol. In order to support both hybrid key exchanges (that is, relying on distinct key exchanges) and fragmentation, the proposed hybrid post-quantum IKEv2 protocol extends IKE [RFC7296] by adding additional key exchange messages (IKE_AUX) between the IKE_SA_INIT and the IKE_AUTH exchanges. In order to minimize communication overhead, only the key shares that are agreed to be used are actually

exchanged. In order to achieve this, the IKE_SA_INIT exchange now includes notify payloads that negotiate the extra key exchanges to be used. The initiator IKE_SA_INIT message includes a notify that lists the extra key exchange policy required by the initiator; the responder selects one of the listed policies, and includes that as a notify in the response IKE_SA_INIT message. Then, the initiator and the responder perform one (or possibly more) IKE_AUX exchange; each such exchange includes a KE payload for the key exchange that was negotiated.

Here is an overview of the initial exchanges:

```
     Initiator                                    Responder
  ------------------------------------------------------------
  <-- IKE_SA_INIT (and extra key exchange negotiation) -->

    <-- {IKE_AUX (hybrid post-quantum key exchange)} -->
                              ...
    <-- {IKE_AUX (hybrid post-quantum key exchange)} -->

                      <-- {IKE_AUTH} -->
```

The extra post-quantum key exchanges can use algorithms that are currently considered to be resistant to quantum computer attacks. These algorithms are collectively referred to as post-quantum algorithms in this document.

3.2. Overall Protocol

In the simplest case, the initiator is happy with a single key exchange (and has no interest in supporting multiple), and he is not concerned with possible fragmentation of the IKE_SA_INIT messages (either because the key exchange he selects is small enough not to fragment, or he is confident that fragmentation will be handled either by IP fragmentation, or transport via TCP). In the following we overview the two protocol rounds involved in the hybrid post-quantum protocol.

In this case, the initiator performs the IKE_SA_INIT as standard, inserting this prefered key exchange (which is possibly a post-quantum algorithm) as the listed Transform Type 4, and including the initiator KE payload. If the responder accepts the policy, he responds with an IKE_SA_INIT response, and IKE continues as usual.

If the initiator desires to negotiate multiple key exchanges, or he needs IKE to handle any possible fragmentation, then he uses the protocol listed below.

3.2.1. First Protocol Round

In the first round, the IKE_SA_INIT request and response messages negotiate the initial IKE SAs (as currently), as well as the key exchanges that will be used within the IKE_AUX phase below.

The initiator negotiates cryptographic suites as per RFC7296, with the listed Transform Type 4 (and KE payload) being either the first key exchange on his desired list of key exchanges, or alternatively a small classical one (in order to enable fragmentation support of the later key exchanges). In addition, the initial IKE_SA_INIT message will include the following two Notify payloads:

- The N(AUX_EXCHANGE_SUPPORTED) notify, as specified in [I-D.smyslov-ipsecme-ikev2-aux]. This draft makes no requirements about the included data.

- An N(EXTRA_KEY_EXCHANGE_POLICY) notify, which has a Protocol ID and SPI Size of 0, and includes the below data.

This data will be the list of groups that the initiator is willing to negotiate during the IKE_AUX phase below. The initiator signifies this by specifying the specific list of the sets of key exchanges that he will allow. The list MUST be ordered from most prefered to least prefered. This is encoded as a series of 2 byte values; a specified list of acceptable groups is given as the specific Transform IDs, followed by a 0x00 value. For example, if the NewHope post-quantum key exchange is 0x40, Round2 is 0x42, and SIKE is 0x47, then the data payload:

```
0040 0000
0042 0047 0000
0042 0000
```

will signify that the initiator is willing to perform IKE_AUX with either NewHope, Round2 followed by SIKE, or only Round2.

If the initiator is willing to skip the IKE_AUX phase, he can signify that by including a 0000 value as a list; for example:

```
0040 0000
0042 0047 0000
0042 0000
0000
```

would signify either (NewHope), (Round2, SIKE), (Round2) or skipping the IKE_AUX entirely.

When the responder that supports the hybrid exchange receives an IKE_SA_INIT message with the AUX_EXHANGE_SUPPORTED and EXTRA_KEY_EXCHANGE_POLICY notifies, then (after processing the IKE message as normal), it scans through the policy listed within the EXTRA_KEY_EXCHANGE_POLICY Notify payload. If the responder finds a list of key exchanges that is consistent with its own policy, it includes N(AUX_EXCHANGE_SUPPORTED) and N(EXTRA_KEY_EXCHANGE_LIST) notifies, which both have 0 Protocol IDs and SPI sizes. The data for the EXTRA_KEY_EXCHANGE_LIST notify would have data specifying the list of acceptable Transform IDs as a series of 2 byte values. If the responder's policy requires it to perform the extra key exchange, but none of the key exchange lists are acceptable, it returns an error in a notification with type NO_PROPOSAL_CHOSEN.

For example, if the single transform Round2 is accepted, then the data payload will consist of:

```
0042
```

If the set Round2 and SIKE is accepted, then the data payload will consist of:

```
0042 0047
```

If no IKE_AUX transforms is desired, then the data payload will be empty (or alternatively no such notification is included, which implies the same thing).

On success, the responder will create the IKE SA and SK values based on SAi1, SAr1 and KE payloads as normal.

When the initiator receives the reply IKE_SA_INIT message, it checks for the existence of the AUX_EXCHANGE_SUPPORTED and EXTRA_KEY_EXCHANGE_LIST notifies. If those notifies are not present, then the initiator treats it as if no extra key exchanges were chosen (and then can proceed by either rejecting the exchange, or proceed using the single negotiated key exchange, depending on local policy).

If those notifies are present, then the responder verifies that the key exchanges listed within the EXTRA_KEY_EXCHANGE_LIST are one of the options within its local policy; if so, it processes the IKE_SA_INIT message as normal, and then proceeds to the IKE_AUX round.

3.2.1.1. Note on responder policy check

One reason that the initiator may select the initial key exchange (the type 4 transform within the SAi1 payload) is not for security, but instead to simply establish keys to allow fragmentation of the IKE_AUX message. Because of this possibility, if the receiver sees a list of key exchanges listed within the EXTRA_KEY_EXCHANGE_LIST that satisfies its policies, it SHOULD accept it (assuming that the SAi1 payload is otherwise acceptable), even if the key payload within the SAi1 is not necessary according to its policy.

3.2.2. IKE_AUX round

For each extra key exchange agreed to in the IKE_SA_INIT exchange, the initiator and the responder perform an IKE_SA_AUX exchange, as described in [I-D.smyslov-ipsecme-ikev2-aux].

This exchange is as follows:

```
Initiator                       Responder
-------------------------------------------------
HDR, SK {Ni2, KEi2}     -->
                        <--     HDR, SK {Nr2, KEr2}
```

The initiator sends a nonce in the Ni2 payload, and the key exchange payload in the KEi2; the group id of the KEi2 payload MUST match the negotiated extra key exchange. This packet is encrypted with the current IKE SK keys.

On receiving this, the responder sends a nonce in the Nr2 payload, and the key exchange payload KEr2; again, this packet is encrypted with the current IKE SA keys.

Once this exchange is done, then both sides compute an updated keying material:

```
SKEYSEED = prf(SK_d(old), KE2result | Ni2 | Nr2)
```

where KE2result is the shared secret of the key exchange. Then, SK_d, SK_ai, SK_ar, SK_ei, SK_er, SK_pi, SK_pr are updated as:

```
{SK_d | SK_ai | SK_ar | SK_ei | SK_er | SK_pi | SK_pr}
        = prf+ (SKEYSEED, Ni2 | Nr2 | SPIi | SPIr)
```

Note that the negotiated transform types (the encryption type, hash type, prf type) are not modified.

Both the initiator and the responder will use this updated key values for the next message.

If the EXTRA_KEY_EXCHANGE_LIST has negotiated more than one key exchange, then this exchange is performed once for every key exchange on the list.

3.2.3. IKE_AUX exchange

After the IKE_AUX exchanges have completed, then the initiator and the responder will perform an IKE_AUTH exchange. This exchange is the standard IKE exchange, except that the initiator and responder signed octets are modified as described in [I-D.smyslov-ipsecme-ikev2-aux].

3.3. Post-quantum Group Transform Type and Group Identifiers

In generating keying material within IKEv2, both initiator and responder negotiate up to four cryptographic algorithms in the SA payload of an IKE_SA_INIT or a CREATE_CHILD_SA exchange. One of the negotiated algorithms is a Diffie-Hellman algorithm, which is used for key exchange. This negotiation is done using the Transform Type 4 (Diffie-Hellman Group) where each Diffie-Hellman group is assigned a unique value.

We expect that in the future, IANA will assign permanent values to these transforms. Until it does, we will use the following values for the below key exchanges (which will need to be specified in more detail elsewhere). Official identifiers will be maintained by IANA and updated during the NIST standardization process.

Name	Number	Key exchange
NIST_CANDIDATE_1	0x9100	The 1st candidate of NIST PQC submission
NIST_CANDIDATE_2	0x9101	The 2nd candidate of NIST PQC submission

Because we are using transforms in the private use space, both the initiator and responder must include a vendor id with this payload:

```
d4 48 11 94 c0 c3 4c 9d d1 22 76 aa 9a 4e 80 d5
```

This payload is the MD5 hash of "IKEv2 Quantum Safe Key Exchange v1"). If the other side does not include this vendor id, an implementation MUST NOT process these private use transforms as listed in this draft.

3.4. Hybrid Group Negotiation

Most post-quantum key agreement algorithms are relatively new, and thus are not fully trusted. There are also many proposed algorithms, with different trade-offs and relying on different hard problems. The concern is that some of these hard problems may turn out to be easier to solve than anticipated (and thus the key agreement algorithm not be as secure as expected). A hybrid solution allows us to deal with this uncertainty by combining a classical key exchanges with a post-quantum one, as well as leaving open the possibility of multiple post-quantum key exchanges.

The method that we use to perform hybrid key exchange also addresses the fragmentation issue. The initial IKE_INIT messages do not have any inherent fragmentation support within IKE; however that can include a relatively short KE payload (e.g. one for group 14, 19 or 31). The rest of the KE payloads are encrypted within IKE_AUX messages; because they are encrypted, the standard IKE fragmentation solution [RFC7383] is available.

3.5. Child SAs

This method of performing hybrid key exchanges, by performing multiple exchanges in series, solves the issue by making the IKE SK values be a function of all the key exchanges performed. Hence, we achieve the goal of making the IKE exchange secure if any of the key exchanges are secure.

This proposal allows the support of multiple post-quantum algorithms (in case we don't have full confidence in any one); this is implemented by having the initiator list all the combinations of extra key exchanges he finds acceptable. It is not anticipated that there will be a need for a large number of different combinations of key exchanges, hence this relatively simple encoding method was selected as a reasonable compromise between simplicity and functionality.

This method also allows us to fragment large post-quantum key exchanges; all the initiator needs to assure is that the initial key exchange (which has the KE payloads exchanged during IKE_SA_INIT) is small enough not to cause fragmentation.

4. Alternative Design

This section gives an overview on a number of alternative approaches that we have considered, but later discarded. These approaches are:

- Sending the classical and post-quantum key exchanges as a single transform

  We considered combining the various key exchanges into a single large KE payload; this effort is documented in a previous version of this draft (draft-tjhai-ipsecme-hybrid-qske-ikev2-01). This does allow us to cleanly apply hybrid key exchanges during the child SA; however it does add considerable complexity, and requires an independant fragmentation solution.

- Sending post-quantum proposals and policies in KE payload only

  With the objective of not introducing unnecessary notify payloads, we considered communicating the hybrid post-quantum proposal in the KE payload during the first pass of the protocol exchange. Unfortunately, this design is susceptible to the following downgrade attack. Consider the scenario where there is an MitM attacker sitting between an initiator and a responder. The initiator proposes, through SAi payload, to use a hybrid post-quantum group and as a backup a Diffie-Hellman group, and through KEi payload, the initiator proposes a list of hybrid post-quantum proposals and policies. The MitM attacker intercepts this traffic and replies with N(INVALID_KE_PAYLOAD) suggesting to downgrade to the backup Diffie-Hellman group instead. The initiator then resends the same SAi payload and the KEi payload containing the public value of the backup Diffie-Hellman group. Note that the attacker may forward the second IKE_SA_INIT message only to the responder, and therefore at this point in time, the responder will not have the information that the initiator prefers the hybrid group. Of course, it is possible for the responder to have a policy to reject an IKE_SA_INIT message that (a) offers a hybrid group but not offering the corresponding public value in the KEi payload; and (b) the responder has not specifically acknowledged that it does not supported the requested hybrid group. However, the checking of this policy introduces unnecessary protocol complexity. Therefore, in order to fully prevent any downgrade attacks, using KE payload alone is not sufficient and that the initiator MUST always indicate its preferred post-quantum proposals and policies in a notify payload in the subsequent IKE_SA_INIT messages following a N(INVALID_KE_PAYLOAD) response.

- New payload types to negotiate hybrid proposal and to carry post-quantum public values

  Semantically, it makes sense to use a new payload type, which mimics the SA payload, to carry a hybrid proposal. Likewise, another new payload type that mimics the KE payload, could be used to transport hybrid public value. Although, in theory a new

payload type could be made backwards compatible by not setting its critical flag as per Section 2.5 of RFC7296, we believe that it may not be that simple in practice. Since the original release of IKEv2 in RFC4306, no new payload type has ever been proposed and therefore, this creates a potential risk of having a backward compatibility issue from non-conforming RFC IKEv2 implementations. Since we could not see any other compelling advantages apart from a semantic one, we use the existing transform type and notify payloads instead. In fact, as described above, we use the KE payload in the first IKE_SA_INIT request round and the notify payload to carry the post-quantum proposals and policies. We use one or more of the existing KE payloads to carry the hybrid public values.

- Hybrid public value payload

  One way to transport the negotiated hybrid public payload, which contains one classical Diffie-Hellman public value and one or more post-quantum public values, is to bundle these into a single KE payload. Alternatively, these could also be transported in a single new hybrid public value payload, but following the same reasoning as above, this may not be a good idea from a backward compatibility perspective. Using a single KE payload would require an encoding or formatting to be defined so that both peers are able to compose and extract the individual public values. However, we believe that it is cleaner to send the hybrid public values in multiple KE payloads--one for each group or algorithm. Furthermore, at this point in the protocol exchange, both peers should have indicated support of handling multiple KE payloads.

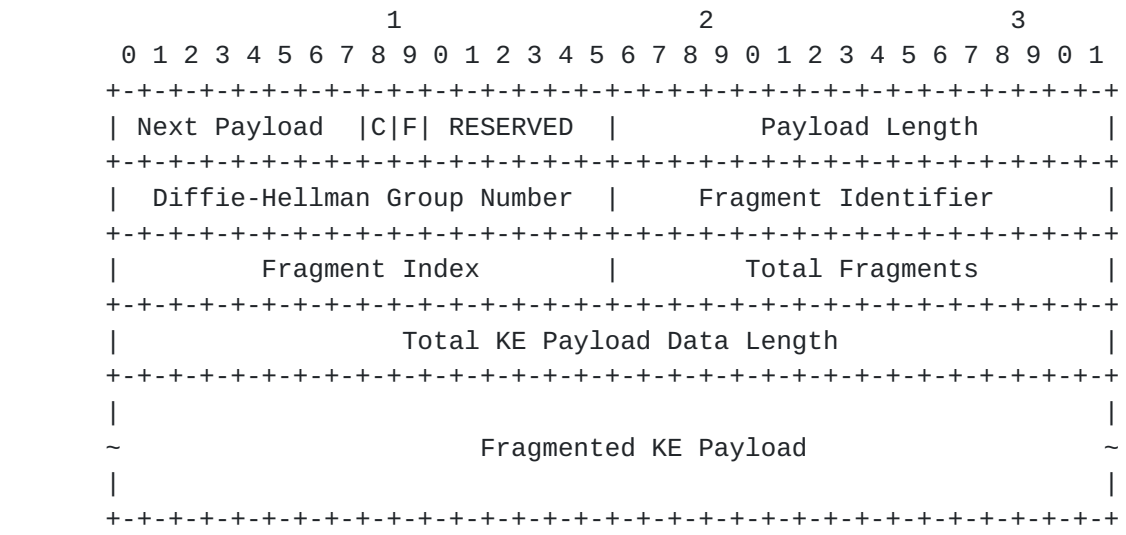
- Fragmentation

  Handling of large IKE_SA_INIT messages has been one of the most challenging tasks. A number of approaches have been considered and the two prominent ones that we have discarded are outlined as follows.

  The first approach was to treat the entire IKE_SA_INIT message as a stream of bytes, which we then split it into a number of fragments, each of which is wrapped onto a payload that would fit into the size of the network MTU. The payload that wraps each fragment is a new payload type and it was envisaged that this new payload type will not cause a backward compatibility issue because at this stage of the protocol, both peers should have indicated support of fragmentation in the first pass of the IKE_SA_INIT exchange. The negotiation of fragmentation is performed using a notify payload, which also defines supporting parameters such as the size of fragment in octets and the fragment identifier. The

new payload that wraps each fragment of the messages in this exchange is assigned the same fragment identifier. Furthermore, it also has other parameters such as a fragment index and total number of fragments. We decided to discard this approach due to its blanket approach to fragmentation. In cases where only a few payloads need to be fragmented, we felt that this approach is overly complicated.

Another idea that was discarded was fragmenting an individual payload without introducing a new payload type. The idea was to use the 9-th bit (the bit after the critical flag in the RESERVED field) in the generic payload header as a flag to mark that this payload is fragmented. As an example, if a KE payload is to be fragmented, it may look as follows.



When the flag F is set, this means the current KE payload is a fragment of a larger KE payload. The Payload Length field denotes the size of this payload fragment in octets--including the size of the generic payload header. The two-octet RESERVED field following Diffie-Hellman Group Number was to be used as a fragment identifier to help assembly and disassembly of fragments. The Fragment Index and Total Fragments fields are self-explanatory. The Total KE Payload Data Length indicates the size of the assembled KE payload data in octets. Finally, the actual fragment is carried in Fragment KE Payload field.

We discarded this approach because we believe that the working group may not be happy using the RESERVED field to change the format of a packet and that implementers may not like the complexity added from checking the fragmentation flag in each received payload. More importantly, fragmenting the messages in this way may leave the system to be more prone to denial of

service (DoS) attacks. By using IKE_AUX to transport the large post-quantum key exchange payloads, there is no longer any issue with fragmentation.

- Group sub-identifier

  As discussed in Section 3.3, each group identifier is used to distinguish a post-quantum algorithm. Further classification could be made on a particular post-quantum algorithm by assigning additional value alongside the group identifier. This sub-identifier value may be used to assign different security parameter sets to a given post-quantum algorithm. However, this level of details does not fit the principles of the document where it should deal with generic hybrid key exchange protocol, not a specific ciphersuite. Furthermore, there are enough Diffie-Hellman group identifiers should this be required in the future.

5. Security considerations

The key length of the Encryption Algorithm (Transform Type 1), the Pseudorandom Function (Transform Type 2) and the Integrity Algorithm (Transform Type 3), all have to be of sufficient length to prevent attacks using Grover's algorithm [GROVER]. In order to use the extension proposed in this document, the key lengths of these transforms SHALL be at least 256 bits long in order to provide sufficient resistance to quantum attacks. Accordingly the post-quantum security level achieved is at least 128 bits.

SKEYSEED is calculated from shared, KEx, using an algorithm defined in Transform Type 2. While a quantum attacker may learn the value of KEx', if this value is obtained by means of a classical key exchange, other KEx values generated by means of a quantum-resistant algorithm ensure that the final SKEYSEED is not compromised. This assumes that the algorithm defined in the Transform Type 2 is post-quantum.

The main focus of this document is to prevent a passive attacker performing a "harvest and decrypt" attack. In other words, an attacker that records messages exchanges today and proceeds to decrypt them once he owns a quantum computer. This attack is prevented due to the hybrid nature of the key exchange. Other attacks involving an active attacker using a quantum-computer are not completely solved by this document. This is for two reasons.

The first reason is because the authentication step remains classical. In particular, the authenticity of the SAs established under IKEv2 is protected using a pre-shared key, RSA, DSA, or ECDSA algorithms. Whilst the pre-shared key option, provided the key is long enough, is post-quantum, the other algorithms are not.

Moreover, in implementations where scalability is a requirement, the pre-shared key method may not be suitable. Quantum-safe authenticity may be provided by using a quantum-safe digital signature and several quantum-safe digital signature methods are being explored by IETF. For example, if the implementation is able to reliably track state, the hash based method, XMSS has the status of an RFC, see [RFC8391]. Currently, quantum-safe authentication methods are not specified in this document, but are planned to be incorporated in due course.

It should be noted that the purpose of post-quantum algorithms is to provide resistance to attacks mounted in the future. The current threat is that encrypted sessions are subject to eavesdropping and archived with decryption by quantum computers taking place at some point in the future. Until quantum computers become available there is no point in attacking the authenticity of a connection because there are no possibilities for exploitation. These only occur at the time of the connection, for example by mounting a MitM attack. Consequently there is not such a pressing need for quantum-safe authenticity.

This draft does not attempt to address key exchanges with KE payloads longer than 64k; the current IKE payload format does not allow that as a possibility. If such huge KE payloads are required, a work around (such as making the KE payload a URL and a hash of the real payload) would be needed. At the current time, it appears likely that there will be plenty of key exchanges available that would not require such a workaround.

6. References

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

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

[GROVER] Grover, L., "A Fast Quantum Mechanical Algorithm for Database Search", Proc. of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing (STOC 1996), 1996.

[I-D.ietf-ipsecme-qr-ikev2] Fluhrer, S., McGrew, D., Kampanakis, P., and V. Smyslov, "Postquantum Preshared Keys for IKEv2", draft-ietf-ipsecme-qr-ikev2-03 (work in progress), June 2018.

[I-D.smyslov-ipsecme-ikev2-aux]
Smyslov, V., "Auxiliary Exchange in the IKEv2 Protocol", draft-smyslov-ipsecme-ikev2-aux-00 (work in progress), January 2018.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://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, <https://www.rfc-editor.org/info/rfc7296>.

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

[RFC8229] Pauly, T., Touati, S., and R. Mantha, "TCP Encapsulation of IKE and IPsec Packets", RFC 8229, DOI 10.17487/RFC8229, August 2017, <https://www.rfc-editor.org/info/rfc8229>.

[RFC8391] Huelsing, A., Butin, D., Gazdag, S., Rijneveld, J., and A. Mohaisen, "XMSS: eXtended Merkle Signature Scheme", RFC 8391, DOI 10.17487/RFC8391, May 2018, <https://www.rfc-editor.org/info/rfc8391>.

Acknowledgements

The authors would like to thanks Frederic Detienne and Olivier Pelerin for their comments and suggestions, including the idea to negotiate the post-quantum algorithms using the existing KE payload.

Authors' Addresses

C. Tjhai
Post-Quantum

Email: cjt@post-quantum.com

M. Tomlinson
Post-Quantum

Email: mt@post-quantum.com

G. Bartlett
Cisco Systems

Email: grbartle@cisco.com

S. Fluhrer
Cisco Systems

Email: sfluhrer@cisco.com

D. Van Geest
ISARA Corporation

Email: daniel.vangeest@isara.com

Z. Zhang
Onboard Security

Email: zzhang@onboardsecurity.com

O. Garcia-Morchon
Philips

Email: oscar.garcia-morchon@philips.com