IPsec Maintenance and Extensions
Internet-Draft
Intended status: Experimental
Expires: April 22, 2010

S. Nagayama
R. Van Meter
Keio University
October 19, 2009

IKE for IPsec with QKD
draft-nagayama-ipsecme-ipsec-with-qkd-00

Status of this Memo

Copyright Notice

Abstract

Quantum Key Distribution (QKD) is a mechanism for creating shared, secret, random bits. This document describes extensions to the IKEv2 protocol to use random bits created via QKD as keys for IPsec. The Diffie-Hellman key agreement mechanism is replaced with QKD. The use of QKD-generated keys with standard IPsec will extend the lifetime of privacy guarantees for IPsec-protected data: future technological advances that break Diffie-Hellman key exchange will not disclose data until such time as the encryption algorithm used for the IPsec tunnel is broken.

Table of Contents

1. Quantum Key Distribution

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].

Quantum key distribution (QKD) [BB84] creates shared, secret, random bits using quantum effects to guarantee that the probability of an undetected eavesdropper learning the secret bits is vanishingly small. Thus, the secret bits are a good source of cryptographic keying material. In the terminology proposed by the SECOQC Project[SECOQC07], a QKD network includes a "secrets plane" which delivers secret key material to other subsystems.

2. Architecture and Assumptions

This document describes modifications to IKEv2 to use keys created via QKD for the Internet Key Exchange IKE_SA[RFC4306], the key agreement protocol for IPsec[RFC4301] . With the exception of the use of the new Payloads defined below and the removal of the Diffie-Hellman key agreement information, IKEv2 system operates in standard fashion.

The system design is shown in Figure 1. Each side has an IPsec Gateway and a QKD Device. The IPsec Gateways are connected via an IP network and the QKD Devices are connected through the QKD network. The IP network and QKD network MAY share all, some, or none of the physical links comprising their networks, e.g. via wavelength multiplexing. Either end MAY initiate the QKD connection.

System Design

```
    Initiator                                           Responder
+-+-+-+-+-+-+-+-+-+       +-+-+-+-+-+-+-+-+          +-+-+-+-+-+-+-+-+-+
! IPsec Gateway !--------!  IP network !-----------! IPsec Gateway !
+-+-+-+-+-+-+-+-+-+       +-+-+-+-+-+-+-+-+          +-+-+-+-+-+-+-+-+-+
       |                                                   |
       |local connection                                   |local
       |                                                   |connection
 +-+-+-+-+-+-+-+-+        +-+-+-+-+-+-+-+-+          +-+-+-+-+-+-+-+-+
 !  QKD Device !---------! QKD Network !------------! QKD Device  !
 +-+-+-+-+-+-+-+-+        +-+-+-+-+-+-+-+-+          +-+-+-+-+-+-+-+-+
```

Figure 1

The connection between the IPsec Gateway and the QKD device, marked "local connection" MUST be secret, authenticated, and reliable. This MAY be achieved by incorporating both the IPsec Gateway and QKD device into a single system.

The QKD device SHALL provide secret, shared, random bits to the IPsec gateway. The bits MUST be shared with an authenticated partner only. The key material SHALL be managed in such a manner that the IPsec gateways can independently map a Key ID to matching key material. Beyond this, the interface between the IPsec Gateway and QKD device is beyond the scope of this document.

The technical details of the operation of the QKD network (including device physics, data filtering, node addressing, authentication, synchronization, etc.) are beyond the scope of this document. The QKD channel operates independently from the IP network that connects the IPsec gateways. QKD requires an authenticated classical channel

which is not part of the IPsec connection; this channel can be unencrypted. The key name (Key ID) is chosen by the QKD subsystem. It is the QKD subsystem's responsibility to ensure that key names are unambiguous, e.g. that key names are not reused within a time frame that can cause confusion.

3. Data Formats and Information Exchange Sequences

IKE must exchange two parameters to use QKD: an identifier indicating which QKD-generated key is to be used (KeyID) and the choice of fallback methods. One Key ID represents one unit of shared random bits, large enough for use as bulk data encryption key. Fallback methods are used when the QKD system key generation underruns.

3.1. Data Formats

3.1.1. QKD Key ID Payload

Figure 2 defines the payload for the QKD Key ID.

QKD Key ID Payload

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
```

```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
! Next Payload  !C!  RESERVED   !         Payload Length        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!    version    !N!    flags    !             reserved          !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
!                             Key ID                            !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 2

The Next Payload field of the previous header MUST be set to the QKD Key ID payload number (see Section 7). The first 32 bits of the payload are the Generic Payload Header. To avoid a man-in-the-middle attack downgrading the negotiated security level, the Critical bit must be set to 1. The responder MUST reply with an error message when it it is incapable of using QKD (see Section 4).

- o Key ID (four octets) is used to communicate which key to use for the encryption.
- o Version (one octet) specifies the format and semantics of this message. The current version is 1.
- o Flags holds flag bits; this field MUST be zero.
- o N is the NoKey bit. 0 means that the KeyID field is valid. 1 means that the KeyID field is not valid, and the responder SHALL resort to the Fallback method, if one is specified in a Fallback Payload. This bit MUST be 0 for IKE_SA_INIT.

3.1.2. QKD Fallback Payload

The Next Payload field of the previous header MUST be set to the QKD Fallback payload number (see Section 7). The first 32 bits of the payload are the Generic Payload Header.

QKD Fallback Payload

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
! Next Payload  !C!  RESERVED   !         Payload Length        !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

```
!     version   !      FLAGS    !           Fallback            !
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 3

- o Version (one octet) specifies the format and semantics of this message. The current version is 1.
- o Flags holds flag bits; this field MUST be zero.
- o Fallback field contains the configure of fallback methods. There are three fallback methods, listed in Table 1.

Fallback methods

Fallback method	Method Value
WAIT_QKD	1
DIFFIE-HELLMAN	2
CONTINUE	3

Table 1

The Fallback methods are as follows:

WAIT_QKD indicating that IKE MUST wait for the QKD device to deliver a new key. When the IPsec tunnel key lifetime expires, the system MUST stop encrypting packets and forwarding them across the network; the tunnel should be considered to be down.

CONTINUE indicating that IPsec MAY continue to use the most recent key until a new key becomes available.

DIFFIE-HELLMAN indicating that IKE SHALL generate a new key in the existing IKE_SA using Diffie-Hellman as defined in [RFC4718].

The Fallback payload is encrypted, relying on the security of the IKE_SA, which is guaranteed by QKD.

3.2. Sequence

To use QKD-generated keys, the Initiator and Responder must agree on a Key ID to use. This key will be used to encrypt the IKE_AUTH exchange, and does not change the IKE Sequence. Other parameters, defining the Fallback method, must be exchanged in IKE_AUTH, in the encrypted connection.

Standard IKEv2 exchanges key data for Diffie-Hellman in IKE_SA_INIT in a synchronous fashion. The principle difficulty in using QKD-generated secret bits as keys for IPsec tunnels is coordinating the activity of the QKD secrets plane with IKE, because the QKD device must operate continuously and independently to monitor its path and create secret bits, as discussed in Appendix A.

3.2.1. Initializing IKE_SA

When the initiator wishes to use QKD-generated keys, it MUST wait until the QKD device delivers one or more valid keys, shared with the responder, before sending the IKE_SA_INIT message. The initiator chooses a key and sends the Key ID in IKE_SA_INIT. The responder echos the Key ID. QKD fallback methods are exchanged in IKE_AUTH. The way to choose fallback methods follows IKE's algorithm to share configuration in [RFC4306] Section 2.7."Cryptographic Algorithm Negotiation".

The key negotation process is described below. Payload names in this document are to be interpreted as described in [RFC4306].

The IKE_SA_INIT with QKD

```
 Initiator                                               Responder
-----------                                             -----------
      HDR, SAi1, KeyID   -->
```

HDR and SAi1 are the IKE Header and a payload which states the cryptographic algorithms the initiator supports for the IKE_SA. KeyID is the QKD Key ID Payload described in Section 3.1.1. The

NoKey bit MUST be 0 in IKE_SA_INIT. The KEi and Ni payloads that are cantained in standard IKEv2 MUST be omitted because they are for Diffie-Hellman, and are not used with QKD.

```
                              <--   HDR, SAr1, KeyID
```

Responder echos Key ID in its KeyID payload.

The IKE_AUTH with QKD exchange

```
 Initiator                                       Responder
-----------                                     -----------
 HDR, SK{IDi, [CERT,] [CERTREQ,] [IDr,]
       QKDfallback, AUTH, SAi2, TSi, TSr}  -->
```

The notation SK{...} means that payloads between {} are encrypted by the SA whose key is chosen in IKE_SA_INIT. QKDfallback payload is the QKD Fallback Payload, containing the initiator's proposed fallback method. IDi, AUTH, SAi2, TSi, TSr are payloads which state the initiator's identification, authentication and CHILD_SA's parameters and traffic selectors of initiator and responder.

```
                   <--  HDR, SK{IDr, [CERT,] QKDfallback
                                    AUTH, SAr2, TSi, TSr}
```

Responder replies with its acceptance of fallback methods in its QKD fallback payload. If the Responder does not agree with the Initiator's requested fallback method, it MUST respond with an error message and abort the IKE negotiation, as discussed in Section 4.

Normal Exchanges in initializing IKE_SA

```
Initiator                                  Responder
    |               IKE_SA_INIT                |
    |----------------------------------------->|
    |    Critical=1, Key ID=foo, No key=0      |
    |                                          |
    |                                          |
    |               IKE_SA_INIT                |
    |<-----------------------------------------|
    |    Critical=1, Key ID=foo, No key=0      |
    |                                          |
    |                                          |
    |                 IKE_AUTH                 |
    |----------------------------------------->|
    |                Critical=1                |
    |    Fallback=[WAIT|DH|CONTINUE]           |
    |                                          |
    |                                          |
    |                 IKE_AUTH                 |
    |<-----------------------------------------|
    |                Critical=1                |
    |    Fallback=[WAIT|DH|CONTINUE]           |
    |                                          |
    |                                          |
```

Figure 4

During initialization, IKE_SA cannot use a fallback method. The key must be generated by QKD. Thus, the Critical bit is set to 1. If the system underruns in key generation, it MUST wait for the QKD device to generate a new key.

3.2.2. Rekeying IKE_SA

In IKE two ways are defined to rekey an IKE_SA: repeating the original initiation sequence by exchanging IKE_SA_INIT and IKE_AUTH, and using the CREATE_CHILD_SA exchange. Because IKE_SA_INIT is exchanged without encryption, if the Initiator wishes to specify fallback behavior, it MUST create a child SA, rather than re-initialize.

The CREATE_CHILD_SA with QKD Exchange

```
 Initiator                                         Responder
------------                                      -----------
    HDR, SK{[N], SA, Ni, KeyID,
              [KEi,] [TSi, TSr]}  -->
```

Figure 5

The initiator sends CREATE_CHILD_SA including an IKE header, optionally a notify, a new security association, a nonce, Key ID for QKD, optionally a key exchange for Diffie-Hellman and optionally traffic selectors.

```
                          <--   HDR, SK{SA, Nr, KeyID,
                                     [KEr,] [TSi, TSr]}
```

The Responder sends CREATE_CHILD_SA which includes an IKE header, a new security association, a nonce, Key ID for QKD, optionally a key exchange for Diffie-Hellman and optionally traffic selectors.

The system SHOULD use QKD to rekey IKE_SA when possible. When the initiator rekeys using a new QKD-generated key, the KeyID payload from the initiator carries the new Key ID and the No key bit is set to 0. Responder repeats the Key ID and sets the No key bit set to 0. The Critical bits are ignored in this case. Both KEi and KEr MUST be omitted.

Normal Message Sequence in CREATE_CHILD_SA for rekeying IKE_SA

```
Initiator                                    Responder
    |              CREATE_CHILD_SA               |
    |------------------------------------------->|
    |                 Critical=1                 |
    |           Key ID=foo, No key=0             |
    |                                            |
    |                                            |
    |              CREATE_CHILD_SA               |
    |<-------------------------------------------|
    |                 Critical=1                 |
```

```
     |           Key ID=foo, No key=0           |
     |                                          |
     |                                          |
```


Figure 6

When the SA lifetime nears expiration and it becomes necessary to rekey, if no QKD-generated key is available the Initiator SHALL rekey using the specified fallback method, if one was specified. The Initiator SHALL send the Fallback Payload with the No key bit set to 1. The initiator SHALL send the Key ID Payload with the KeyID field set to 0 and the NoKey bit set to 1. The Responder SHALL reply with the same Key ID and Fallback payloads. If Diffie-Hellman is permitted as a fallback method and the Perfect Forward Security(PFS) is configured to work, CREATE_CHILD_SA carries KEi and KEr.

Fallback Exchanges in CREATE_CHILD_SA for rekeying IKE_SA

```
Initiator                                  Responder
     |             CREATE_CHILD_SA              |
     |----------------------------------------->|
     |                Critical=1                |
     |           Key ID=NULL, No key=1          |
     |                                          |
     |                                          |
     |             CREATE_CHILD_SA              |
     |<-----------------------------------------|
     |                Critical=1                |
     |           Key ID=NULL, No key=1          |
     |                                          |
     |                                          |
```

Figure 7

3.3. Considerations for Multiple SAs

IPsec can have multiple SAs between two IPsec gateways. QKD provides node-to-node keys, thus the system described in this document can also manage multiple SAs. The Initiator is free to use any QKD-generated keys for any SAs, but MUST NOT reuse any key.

4. Error Handling

Error handling beyond that already described in this document is TBD.

5. Recommendations for use of QKD-generated keys

From a data privacy point of view, the ideal use of QKD-generated keys would be as a one-time pad (OTP) to protect the data carried in the IPsec tunnel. However, as of 2009, QKD key generation rates are not adequate for high-speed OTP use; the QKD-generated keys instead will be used most commonly as key material for symmetric encryption of the IPsec tunnel. Thus, the upper bound on the secret lifetime of data remains the time until the chosen symmetric cipher can be broken. An eavesdropper who records encrypted packets today can store those packets, and decrypt them later by directly attacking the symmetric cipher, when it becomes technically feasible to do so.

However, existing IPsec/IKE implementations actually have a lower data secrecy lifetime, due to their dependence on Diffie-Hellman key agreement. The security of Diffie-Hellman depends on the difficulty of the factoring problem, which remains uncertain; factoring may prove vulnerable either to theoretical advances in algorithms, or the deployment of large-scale quantum computers. An eavesdropper who records encrypted packets today can store those packets, and decrypt them later by discovering the key, when it becomes technically feasible to do so.

QKD+IKE+IPsec offers a different point in the security space, providing secrecy under different assumptions about computational difficulty than D-H+IKE+IPsec, for all choices of IPsec tunnel encryption protocol.

In summary:

o QKD+IKE+IPsec depends on the availability of an authentication mechanism that is secure at the time of key negotiation.

o If QKD keys are used as an OTP, transferred data remains secret forever (or until disclosed through alternate means).

o If QKD keys are used for symmetric encryption, an eavesdropper may copy and store packets but cannot decrypt them until the symmetric cipher can be broken.

In contrast:

o D-H+IKE+IPsec depends on the availability of an authentication mechanism that is secure at the time of key negotiation.

o If the D-H keys are used for symmetric encryption, an eavesdropper may copy and store packets, and will be able to decrypt them when it becomes possible EITHER to factor large numbers (breaking the

D-H key agreement) OR to break the symmetric cipher.

Thus, QKD+IKE+IPsec can remove one uncertainty about the future evolution of computational security. If factoring is easier than breaking symmetric encryption, the use of QKD will extend the timeframe for maintaining the secrecy of data, even if standard, symmetric encryption is used for the bulk data encryption.

Key lifetime could be matched to QKD key generation rate; the mechanism is not specified here.

6. Security Considerations

Because QKD's principal role is to detect eavesdropping and discard possibly compromised bits, eavesdropping is a very effective denial of service (DoS) attack. One purpose of the fallback behavior negotiation is to provide network managers with a tool for alleviating this problem. Fallback methods should be used with extreme care, and SHOULD be coupled with event notification and monitoring.

One possible practice would be to define the fallback policy for an SA carrying user traffic as WAIT_QKD, and define a second, primarily dormant, SA with a more liberal fallback policy for a management station. The second SA might be used only to diagnose problems and for low-security network monitoring and management activity until the QKD connection can be restored.

This document describes a form of Internet Key Exchange protocol which is not based on the difficulty of factorization. Thus, under the circumstances described in Section 5, security may be improved.

The system consists of two logically separate channels: a classical channel between IPsec gateways and a quantum channel between QKD-devices. The QKD devices require a classical channel and authentication to prevent a man-in-the-middle attack. One keys are securely transferred to the IPsec gateway, those keys could be used as an alternative method for authenticating the IPsec gateways. Careful integration of the classical and quantum networks could eliminate authentication on one path by sharing the authentication information from the other; such a use is not specified here.

7. IANA Considerations

The following new assignments can only be made via a Standards Action as specified in [refs.IANA].

7.1. Payload Type Values

The IANA should allocate IKE Payload Type Values for the QKD Key ID Payload and the QKD Fallback Payload upon publication of the first RFC.

8. References

8.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

[RFC4306] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol", RFC 4306, December 2005.

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

[refs.IANA]
Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", RFC 2434, October 1998.

8.2. Informative References

[BB84] Bennett, C. and G. Brassard, "Quantum cryptography: Public key distribution and coin tossing", 1984.

[EPT03] Elliot, C., Pearson, D., and G. Troxel, "Quantum cryptography in practice", 2003.

[SECOQC07]
Alleaume, R. and et al., "SECOQC White Paper on Quantum Key Distribution and Cryptography", 2007.

[UQC09] Dodson, D. and et al., "Updating Quantum Cryptography Report ver. 1", 2009.

Appendix A. Implementation Considerations and Current Status

As of 2009, available QKD products use single photons over dedicated optical fibers and are limited in distance. Experimental demonstrations of wireless links and multi-hop networks using trusted intermediate nodes have been conducted [EPT03]. Progress is also being made toward use of satellite links and quantum entanglement-based networks of quantum repeaters that will not require trusting intermediate nodes [SECOQC07][UQC09].

In general, because QKD relies heavily on statistical evidence to determine the presence of an eavesdropper, it requires time to create a key. Thus, the IPsec implementation should be prepared for a long delay before keys become available. Moreover, the key generation rate may vary over time, typically rising over a long period from the initiation of a connection as statistical certainty improves, then settling near a sustained value around which the rate may vary as conditions change.

Authors' Addresses

Shota Nagayama
Keio University
Faculty of Policy Management
5322 Endo
Fujisawa, Kanagawa 252-8520
Japan

Email: kurosagi@sfc.wide.ad.jp

Rodney Van Meter
Keio University
Faculty of Environment and Information Studies
5322 Endo
Fujisawa-shi, Kanagawa-ken 252-8520
Japan

Email: rdv@sfc.wide.ad.jp