Internet Engineering Task Force (IETF)
Request for Comments: 8263
Category: Standards Track
ISSN: 2070-1721

B. Weis
Cisco Systems
U. Mangla
Juniper Networks Inc.
T. Karl
Deutsche Telekom
N. Maheshwari
November 2017

Group Domain of Interpretation (GDOI) GROUPKEY-PUSH Acknowledgement Message

Abstract

The Group Domain of Interpretation (GDOI) includes the ability of a Group Controller/Key Server (GCKS) to provide a set of current Group Member (GM) devices with additional security associations (e.g., to rekey expiring security associations). This memo adds the ability of a GCKS to request that the GM devices return an acknowledgement of receipt of its rekey message and specifies the acknowledgement method.

Status of This Memo

Table of Contents

1. Introduction

The Group Domain of Interpretation (GDOI) [RFC6407] is a group key management method by which a Group Controller/Key Server (GCKS) distributes security associations (i.e., cryptographic policy and keying material) to a set of Group Member (GM) devices. The GDOI meets the requirements set forth in [RFC4046] ("Multicast Security (MSEC) Group Key Management Architecture"), including a Registration Protocol and a Rekey Protocol. The GDOI describes the Rekey Protocol as a GROUPKEY-PUSH message.

A GDOI GCKS uses a GROUPKEY-PUSH message (Section 4 of [RFC6407]) to alert GMs to updates in policy for the group, including new policy and keying material, replacement policy and keying material, and indications of deleted policy and keying material. Usually, the GCKS does not require a notification that the GM actually received the policy. However, in some cases it is beneficial for a GCKS to be told by each receiving GM that it received the rekey message and, by implication, has reacted to the policy contained within. For example, a GCKS policy can use the acknowledgements to determine which GMs are receiving the current group policy and which GMs are no longer participating in the group.

This memo introduces a method by which a GM returns an Acknowledgement Message to the GCKS. Initially, a GCKS requests that a GM acknowledge GROUPKEY-PUSH messages as part of a distributed group policy. Then, as shown in Figure 1, when the GCKS delivers a GROUPKEY-PUSH message, each GM that honors the GCKS request returns a GROUPKEY-PUSH Acknowledgement Message. The rest of this memo describes this method in detail.

```
             GCKS                          GM1       GM2
              |                             |         |
              |                 +---------->|         |
              |   GROUPKEY-PUSH |           |         |
              |-----------------+           |         |
              |                 |           |         |
              |                 +-------------------->|
              |                             |         |
              |<----------------------------|         |
              |      GROUPKEY-PUSH ACK      |         |
              |                             |         |
              |<--------------------------------------|
              |      GROUPKEY-PUSH ACK      |         |
```

Figure 1: GROUPKEY-PUSH Rekey Event

Implementation of the GROUPKEY-PUSH Acknowledgement Message is OPTIONAL.

1.1. Requirements Notation

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

1.2. Acronyms and Abbreviations

The following acronyms and abbreviations are used throughout this document.

ACK	Acknowledgement Message
D	Delete
GCKS	Group Controller/Key Server
GDOI	Group Domain of Interpretation
GM	Group Member
HDR	Header
HMAC	Hashed Message Authentication Code
IV	Initialization Vector
KD	Key Download
KDF	Key Derivation Function
KEK	Key Encryption Key
LKH	Logical Key Hierarchy
MSEC	Multicast Security
PRF	Pseudorandom Function
SA	Security Association

SEQ	Sequence Number
SIG	Signature
SPI	Security Parameter Index

2. Acknowledgement Message Request

When a GM is ready to join a group, it contacts the GCKS with a GROUPKEY-PULL Registration Protocol. When the GCKS has authenticated and verified that the GM is an authorized member of the group, it downloads several sets of policy in a Security Association (SA) payload. If the group includes the use of a GROUPKEY-PUSH Rekey Protocol, the SA payload includes an SA Key Encryption Key (KEK) payload (Section 5.3 of [RFC6407]). When necessary, the GROUPKEY-PUSH Rekey Protocol also contains an SA payload that includes the SA KEK policy. The SA KEK policy indicates how the GM will be receiving and handling the GROUPKEY-PUSH Rekey Protocol.

When the GCKS policy includes the use of the GROUPKEY-PUSH Acknowledgement Message, the GCKS reports this policy to the GM within the SA KEK policy. The GCKS includes a new KEK attribute with the name KEK_ACK_REQUESTED (9), which indicates that the GM is requested to return a GROUPKEY-PUSH Acknowledgement Message.

As part of the SA KEK policy, the GCKS specifies information on the keying material that is used to protect the GROUPKEY-PUSH Rekey Protocol (e.g., the presence of a KEK management algorithm). Parts of this information are used by a GM to derive the ack_key (defined in Section 3.2), which protects the GROUPKEY-PUSH Acknowledgement Message. There are different types of Rekey Acknowledgement Messages; they share an identical message format but differ in the keying material used.

The following values of the KEK_ACK_REQUESTED attribute are defined in this memo.

2.1. REKEY_ACK_KEK_SHA256 Type

This type of Rekey ACK is used when the KEK Download Type (Section 5.6.2 of [RFC6407]) is part of the group policy. The prf (defined in Section 3.2) is PRF-HMAC-SHA-256 [RFC4868]. The base_key (also defined in Section 3.2) is the KEK_ALGORITHM_KEY used to decrypt the GROUPKEY-PUSH message. Note that for some algorithms the KEK_ALGORITHM_KEY will include an explicit Initialization Vector (IV) before the actual key (Section 5.6.2.1 of [RFC6407]), but it is not used in the definition of the base_key.

2.2. REKEY_ACK_LKH_SHA256 Type

This type of Rekey ACK can be used when the KEK_MANAGEMENT_ALGORITHM KEK attribute with a value representing the Logical Key Hierarchy (LKH) is part of the group policy (Section 5.3.1.1 of [RFC6407]). The prf is PRF-HMAC-SHA-256. The base_key is the Key Data field value taken from the first LKH Key structure in an LKH_DOWNLOAD_ARRAY attribute (see Section 5.6.3.1 of [RFC6407]). This is a secret symmetric key that the GCKS shares with the GM. Note that for some algorithms the LKH Key structure will include an explicit IV before the actual key (Section 5.6.3.1 of [RFC6407]), but it is not used in the definition of the base_key.

2.3. REKEY_ACK_KEK_SHA512 Type

This type of Rekey ACK is identical to the REKEY_ACK_KEK_SHA256 Type, except that the prf is PRF-HMAC-SHA-512 (defined in [RFC4868]).

2.4. REKEY_ACK_LKH_SHA512 Type

This type of Rekey ACK is identical to the REKEY_ACK_LKH_SHA256 Type, except that the prf is PRF-HMAC-SHA-512 (defined in [RFC4868]).

3. GROUPKEY-PUSH Acknowledgement Message

The GROUPKEY-PUSH message defined in [RFC6407] is reproduced in Figure 2. The SA and Key Download (KD) payloads contain the actual policy and keying material being distributed to the GM. The Sequence Number (SEQ) payload contains a sequence number that is used by the GM for replay protection. This sequence number defines a unique rekey message delivered to that GM. One or more Delete (D) payloads optionally specify the deletion of the existing group policy. The Signature (SIG) payload includes a signature of a hash of the entire GROUPKEY-PUSH message (excepting the SIG payload octets) before it has been encrypted.

```
      GM                   GCKS
      --                   ----
                           <---- HDR*, SEQ, [D,] SA, KD, SIG

      * Protected by the Rekey SA KEK; encryption occurs after HDR
```

Figure 2: GROUPKEY-PUSH Message (from RFC 6407)

When the GM has received a KEK_ACK_REQUESTED attribute in an SA KEK and it chooses to respond, it returns the value of the Sequence Number taken from the GROUPKEY-PUSH message to the GCKS along with its identity. This tuple alerts the GCKS that the GM has received the GROUPKEY-PUSH message and implemented the policy contained therein. The GROUPKEY-PUSH Acknowledgement Message is shown in Figure 3.

```
                   GM                              GCKS
                   --                              ----
                      HDR, HASH, SEQ, ID   ---->
```

Figure 3: GROUPKEY-PUSH Acknowledgement Message

The IP header for the GROUPKEY-PUSH Acknowledgement Message is constructed as if it were a reply to the GROUPKEY-PUSH message. That is, the source address of the GROUPKEY-PUSH message becomes the destination address of the GROUPKEY-PUSH Acknowledgement Message, and the GM includes its own IP address as the source address of the GROUPKEY-PUSH Acknowledgement Message. The source port in the GROUPKEY-PUSH message UDP header becomes the destination port of the GROUPKEY-PUSH Acknowledgement Message UDP header, and the destination port of the GROUPKEY-PUSH message UDP header becomes the source port of the GROUPKEY-PUSH Acknowledgement Message UDP header.

The following sections describe the payloads in the GROUPKEY-PUSH Acknowledgement Message.

3.1. HDR

The message begins with a header as defined for the GDOI GROUPKEY-PUSH message in Section 4.2 of [RFC6407]. The fields in the HDR MUST be initialized as follows. The cookies of a GROUPKEY-PUSH message act as a Security Parameter Index (SPI) and are copied to the Acknowledgement Message. "Next Payload" identifies a "Hash (HASH)" payload (value 8) [ISAKMP-NP]. Major Version is 1 and Minor Version is 0. The Exchange Type has value 35 for the GDOI GROUPKEY-PUSH Acknowledgement Message. Flags are set to 0. Message ID MUST be set to 0. Length is according to Section 4.2 of [RFC6407].

3.2. HASH

The HASH payload is the same one used in the GDOI GROUPKEY-PULL exchange defined in Section 3.2 of [RFC6407]. The hash data in the HASH payload is created as follows:

```
HASH = prf(ack_key, SEQ | ID)
```

where:

- o "prf" is specific to the KEK_ACK_REQUESTED value and is described as part of that description.
- o "|" indicates concatenation.
- o "SEQ" and "ID" represent the bytes comprising the Sequence Number and Identification payloads.

The ack_key is computed from a Key Derivation Function (KDF) that conforms to KDF in feedback mode as defined in NIST SP800-108 [SP800-108], where the length of the derived keying material is the same as the output of the prf, there is no IV, and the optional counter is not used. Note: When the derived ack_key is smaller than the prf block size (i.e., 512 bits for PRF-HMAC-SHA-256), it is zero-filled to the right, as specified in Section 2.1.2 of [RFC4868].

```
ack_key = prf(base_key, "GROUPKEY-PUSH ACK" | SPI | L)
```

where:

- o "prf" is specific to the KEK_ACK_REQUESTED value and is described as part of that description.
- o "base_key" is specific to the KEK_ACK_REQUESTED value and is described as part of that description. If the base_key is smaller than the prf block size (i.e., 512 bits for PRF-HMAC-SHA-256), then it is zero-filled to the right, as specified in Section 2.1.2 of [RFC4868].
- o "|" indicates concatenation.
- o "GROUPKEY-PUSH ACK" is a label encoded as a null-terminated ASCII string.
- o "SPI" (per [RFC6407]) is the Initiator Cookie followed by the Responder Cookie taken from the GROUPKEY-PUSH message HDR, which describes the context of the key usage.

- "L" is a length field matching the number of bits in the ack_key. L MUST match the length of the base_key (i.e., 512 bits for PRF-HMAC-SHA-256). The value L is represented as two octets in network byte order (that is, most significant byte first).

3.3. SEQ

The Sequence Number payload is defined in Section 5.7 of [RFC6407]. The value in the GROUPKEY-PUSH SEQ payload is copied to the GROUPKEY-PUSH ACK SEQ payload.

3.4. ID

The Identification payload is used as defined in Section 5.1 of [RFC6407]. The ID payload contains an ID Type of ID_IPV4_ADDR, ID_IPV6_ADDR, or ID_OID as defined in [RFC8052] for GDOI exchanges. The Protocol ID and Port fields MUST be set to 0. The address provided in the ID payload represents the IP address of the GM and MUST match the source IP address used for the most recent GROUPKEY-PULL exchange.

4. Group Member Operations

When a GM receives an SA KEK payload (in a GROUPKEY-PULL exchange or GROUPKEY-PUSH message) including a KEK_ACK_REQUESTED attribute, it records in its group state some indication that it is expected to return a GROUPKEY-PUSH ACK. A GM recognizing the attribute MUST honor the KEK_ACK_REQUESTED attribute by returning Acknowledgements, because it can be expected that the GCKS is likely to take some policy-specific action regarding unresponsive GMs, including ceasing to deliver GROUPKEY-PUSH messages to it.

If a GM cannot respond with the requested type of Acknowledgement, it continues with protocol exchange and participates in the group. In any case, if a GM stops receiving GROUPKEY-PUSH messages from a GCKS, it will re-register before existing SAs expire, so omitting the sending of Acknowledgements should not be critical.

When a GM receives a GROUPKEY-PUSH message that contains a KEK_ACK_REQUESTED attribute in the SA KEK payload, it processes the message according to RFC 6407. When it concludes successful processing of the message, it formulates the GROUPKEY-PUSH ACKs as described in Section 3 and delivers the message to the GCKS from which the GROUPKEY-PUSH message was received. A GROUPKEY-PUSH ACK is sent even if the GROUPKEY-PUSH message contains a Delete payload for the KEK used to protect the GROUPKEY-PUSH message.

5. GCKS Operations

When a GCKS policy includes requesting a GROUPKEY-PUSH ACK from GMs, it includes the KEK_ACK_REQUESTED attribute in the SA KEK payload. It does this each time the SA KEK is delivered, in both GROUPKEY-PULL exchanges and GROUPKEY-PUSH messages. The value of the KEK_ACK_REQUESTED attribute will depend upon the type of SA KEK policy, as described in Section 2.

When a GCKS receives a GROUPKEY-PUSH ACK (identified by an Exchange Type of GROUPKEY-PUSH-ACK), it first verifies that the group policy includes receiving GROUPKEY-PUSH ACKs. If not, the message is discarded. GCKS implementations SHOULD keep a record (e.g., a hash value) of recently received GROUPKEY-PUSH Acknowledgement Messages and reject duplicate messages prior to performing cryptographic operations. This enables an early discard of the replayed messages.

If the message is expected, the GCKS validates the format of the message and verifies that the HASH has been properly constructed as described in Section 3.2. If validation fails, the message is discarded. The GCKS extracts the sequence number and identity of the GM from the SEQ and ID payloads, respectively, and records the fact that the GM received the GROUPKEY-PUSH message represented by its sequence number.

6. Management Considerations

The GCKS manages group policy as well as determining which GM devices are presently "live" members of the group (i.e., members either sending or receiving messages). Group policy includes a strategy to ensure that rekey messages with current group policy reach all live GMs. This is discussed briefly in Section 5.3 of [RFC4046]. The GROUPKEY-PUSH Acknowledgement Message specified in this memo provides the GCKS with an additional method to assess if a GM is live and has received the current group policy. But it is possible for a rekey message or GROUPKEY-PUSH Acknowledgement Message to be discarded in the network, resulting in a live GM appearing to be unresponsive. Also, a GM might not be able to respond with a GROUPKEY-PUSH ACK, so the GCKS should use caution in using a lack of an Acknowledgement Message as the only factor in determining whether a GM is live. In particular, a GCKS SHOULD NOT consider a GM to have left the group until it has received at least one ACK from the GM.

Some management considerations for determining how a GM handles Acknowledgement Messages are as follows:

- o A GM MUST respond with Acknowledgement Messages when requested, as a GCKS can subsequently determine when a GM unexpectedly becomes unresponsive.

- o A GM receiving a GROUPKEY-PUSH message as a multicast message MAY introduce jitter to the timing of its Acknowledgement Message to help the GCKS better manage replies from GMs. A GM MUST NOT delay sending an Acknowledgement Message for more than 5 seconds. a GCKS SHOULD NOT declare an Acknowledgement Message as missing until it has waited at least 10 seconds. Implementations SHOULD make these timers configurable.

Some management considerations for determining how the GCKS handles Acknowledgement Messages are as follows:

- o Non-receipt of an Acknowledgement Message is an indication that a GM is unable to respond. A GCKS SHOULD wait at least several seconds before determining non-receipt, as GMs could add jitter to the response time before sending an Acknowledgement Message.

- o If the GCKS is aware that GMs are expected to respond, then non-receipt of an Acknowledgement Message SHOULD trigger a logging event. The GCKS MAY be configured with such additional policy actions as transmitting the GROUPKEY-PUSH message several times in a short period of time (as suggested in [RFC4046]), thereby mitigating loss of either the GROUPKEY-PUSH message or an Acknowledgement Message. Another policy action could be to alert GCKS administrators of GMs that do not return several consecutive Acknowledgement Messages or even removing unresponsive GMs from the group. However, a GCKS with a policy of removing GMs from the group needs to be aware that a GM that has not responded will not receive a newer group policy until it initiates contact with the GCKS again.

- o When a GROUPKEY-PUSH message includes a Delete payload for the KEK used to protect the GROUPKEY-PUSH message, the GCKS SHOULD NOT itself delete the KEK until it has given GMs the opportunity to acknowledge receipt of the GROUPKEY-PUSH message. This could be several seconds, as GMs could add jitter to the response time before sending an Acknowledgement Message.

- o A GCKS SHOULD log failure events, such as receiving Acknowledgement Messages for a group in which the GCKS has not requested Acknowledgements, receiving malformed Acknowledgements, and Acknowledgements that fail validation.

7. Security Considerations

There are three areas of security considerations to consider: the protection of the GROUPKEY-PUSH ACK, whether the GM should transmit a GROUPKEY-PUSH ACK, and whether a GCKS should accept a GROUPKEY-PUSH ACK. These are addressed in the following subsections.

The construction of the HASH defined in this memo uses PRF-HMAC-SHA-256 or PRF-HMAC-SHA-512. The strengths of PRF-HMAC-SHA-256 and PRF-HMAC-SHA-512 were unquestioned at the time this memo was developed. When a HASH construction using a different prf becomes necessary, a new KEK_ACK_REQUESTED value will be defined in a new specification.

7.1. Protection of the GROUPKEY-PUSH ACK

The GROUPKEY-PUSH ACK is an Internet Security Association and Key Management Protocol (ISAKMP) message as discussed in [RFC2408]. (Note: RFC 2408 has been obsoleted by RFC 7296, but only RFC 2408 applies in this context.) Message authentication and protection against man-in-the-middle attacks are provided by the inclusion of a HASH payload that includes the output of an HMAC computation over the bytes of the message.

Because the KEK is a group secret, when the value of REKEY_ACK_KEK is specified, impersonation of a victim GM by another authorized GM is possible. However, security considerations regarding such an impersonation are limited to a false claim that a victim GM has received a GROUPKEY-PUSH when the victim GM has in fact not received it (e.g., because an active attacker has discarded the GROUPKEY-PUSH). If a GCKS policy includes sending retransmissions of the GROUPKEY-PUSH message to that victim GM, then the victim GM might not receive replacement SAs. However, this does not introduce any additional threats over a use case where the GROUPKEY-PUSH ACK is not deployed and GROUPKEY-PUSH messages are withheld from a victim GM by an active attacker. These threats can be mitigated by using a value of REKEY_ACK_LKH, due to the use of a secret pairwise key shared between the GCKS and an individual GM.

Confidentiality is not provided for the GROUPKEY-PUSH ACK. The contents of the message, including the hash value, the sequence number from the GROUPKEY-PUSH message to which it is acknowledging receipt, and the identity of the GM, can be observed by a passive attacker. Observation of a hash value or set of hash values will not compromise the hash key. The identity of the GM is also available to the passive attacker as the source IP address of the packet. Note that the sequence number in the GROUPKEY-PUSH ACK does reveal the sequence number (previously not available to the attacker) that was included in the GROUPKEY-PUSH message. However, the attacker is assumed to not be in possession of the key used to encrypt the message and thus cannot create a spoofed GROUPKEY-PUSH message. Therefore, the attacker does not derive any direct value from learning the sequence number.

7.2. Transmitting a GROUPKEY-PUSH ACK

A GM transmits an ACK only when the policy of the most recently received SA KEK includes a request by the GCKS for ACKs, and the ACK is only returned after processing the GROUPKEY-PUSH message according to Section 4.4 of [RFC6407]. In other words, the form of the GROUPKEY-PUSH message will have been validated, replay protection completed, and the digital signature verified as being genuine. Therefore, the threat of a GM responding to a spoofed or resent GROUPKEY-PUSH message, and the possibility of the GM being used to propagate a Distributed Denial of Service (DDoS) attack on a GCKS, are mitigated. For more information, see the security considerations for a GROUPKEY-PUSH message as described in Section 7.3 of [RFC6407].

7.3. Receiving a GROUPKEY-PUSH ACK

A GCKS receiving ACKs will follow the validation steps described in Section 5 before interpreting the contents of the message. The GCKS will then be sure to operate only on messages that have been sent by an authorized GM.

A GCKS SHOULD be prepared to receive GROUPKEY-PUSH ACKs from each GM to which it was sent. That is, it needs to ensure that it has sufficient resources (e.g., receive queue size) so that it does not unnecessarily drop ACKs. A GCKS should be aware that a large number of replayed or invalid GROUPKEY-PUSH messages could be addressed to it. However, this is no worse a threat than if it received a large number of other types of replayed or invalid GDOI or other messages containing a HASH payload.

How a GCKS processes the sequence number and identity included in an ACK is a matter of local policy and is outside the scope of this memo.

8. IANA Considerations

The following additions have been made to the "Group Domain of Interpretation (GDOI) Payloads" [GDOI-REG] registry.

A new attribute has been added to the "SA KEK Payload Values - KEK Attributes" registry. The ID Class name is KEK_ACK_REQUESTED with a value of 9 and is a Basic attribute.

A new registry defining values for KEK_ACK_REQUESTED, "SA KEK Payload Values - KEK_ACK_REQUESTED", has been added; the initial registrations are shown in the following table. The terms "Reserved", "Unassigned", and "Private Use" are to be applied as defined in [RFC8126]. The registration procedure is Specification Required.

Value	Type
0	Reserved
1	REKEY_ACK_KEK_SHA256
2	REKEY_ACK_LKH_SHA256
3	REKEY_ACK_KEK_SHA512
4	REKEY_ACK_LKH_SHA512
5-128	Unassigned
129-255	Private Use

A new registry describing ISAKMP Exchange Types for the GDOI, "GDOI DOI Exchange Types", has been added under the "Group Domain of Interpretation (GDOI) Payloads" registry [GDOI-REG]. This new registry defines DOI Specific Use values [ISAKMP-EXCH], which are Exchange Type values used with the ISAKMP GDOI DOI. The registration procedure is Specification Required. The terms "Known Unregistered Use" and "Unassigned" are to be applied as defined in [RFC8126].

Value	Phase	Reference
GROUPKEY-PULL	32	RFC 6407
GROUPKEY-PUSH	33	RFC 6407
Known Unregistered Use	34	
GROUPKEY-PUSH-ACK	35	RFC 8263
Unassigned	36-239	

9. References

9.1. 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, <https://www.rfc-editor.org/info/rfc2119>.

[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, <https://www.rfc-editor.org/info/rfc4868>.

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

[RFC8052] Weis, B., Seewald, M., and H. Falk, "Group Domain of Interpretation (GDOI) Protocol Support for IEC 62351 Security Services", RFC 8052, DOI 10.17487/RFC8052, June 2017, <https://www.rfc-editor.org/info/rfc8052>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

[GDOI-REG]
Internet Assigned Numbers Authority, "Group Domain of Interpretation (GDOI) Payload Type Values", IANA Registry, September 2017, <https://www.iana.org/assignments/gdoi-payloads/>.

[ISAKMP-EXCH]
Internet Assigned Numbers Authority, "Internet Key Exchange (IKE) Attributes Exchange Type Values", IANA Registry, May 2013, <https://www.iana.org/assignments/ipsec-registry/>.

[ISAKMP-NP]
Internet Assigned Numbers Authority, "Internet Key Exchange (IKE) Attributes Next Protocol Types", IANA Registry, May 2013, <https://www.iana.org/assignments/ipsec-registry/>.

[RFC2408] Maughan, D., Schertler, M., Schneider, M., and J. Turner, "Internet Security Association and Key Management Protocol (ISAKMP)", RFC 2408, DOI 10.17487/RFC2408, November 1998, <https://www.rfc-editor.org/info/rfc2408>.

[RFC4046] Baugher, M., Canetti, R., Dondeti, L., and F. Lindholm, "Multicast Security (MSEC) Group Key Management Architecture", RFC 4046, DOI 10.17487/RFC4046, April 2005, <https://www.rfc-editor.org/info/rfc4046>.

[SP800-108]
Chen, L., "Recommendation for Key Derivation Using Pseudorandom Functions (Revised)", National Institute of Science and Technology, NIST Special Publication 800-108, DOI 10.6028/NIST.SP.800-108, October 2009, <http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-108.pdf>.

Acknowledgements

Mike Hamada, Adrian Farrel, and Yaron Sheffer provided many useful technical and editorial comments and suggestions for improvement.

Authors' Addresses

Brian Weis
Cisco Systems
170 W. Tasman Drive
San Jose, California 95134-1706
United States of America

Phone: +1-408-526-4796
Email: bew@cisco.com

Umesh Mangla
Juniper Networks Inc.
1133 Innovation Way
Sunnyvale, California 94089
United States of America

Phone: +1-408-936-1022
Email: umangla@juniper.net

Thomas Karl
Deutsche Telekom
Landgrabenweg 151
Bonn 53227
Germany

Phone: +49-228-18138122
Email: thomas.karl@telekom.de

Nilesh Maheshwari

Email: nileshm@gmail.com