Stream: Internet Engineering Task Force (IETF)
RFC: 9838
Obsoletes: 6407
Category: Standards Track
Published: October 2025
ISSN: 2070-1721

Authors: V. Smyslov *ELVIS-PLUS* B. Weis *Independent*

RFC 9838
Group Key Management Using the Internet Key Exchange Protocol Version 2 (IKEv2)

Abstract

This document presents an extension to the Internet Key Exchange Protocol Version 2 (IKEv2) for the purpose of group key management. The protocol is in conformance with the Multicast Security (MSEC) Group Key Management architecture, which contains two components: member registration and group rekeying. Both components are required for a Group Controller/Key Server (GCKS) to provide authorized Group Members (GMs) with IPsec Group Security Associations (GSAs). The GMs then exchange IP multicast or other group traffic as IPsec packets.

This document obsoletes RFC 6407.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction and Overview

This document presents an extension to IKEv2 [RFC7296] called G-IKEv2, which accommodates group key management. A group key management protocol provides IPsec keys and policy to a set of IPsec devices that are authorized to communicate using a Group Security Association (GSA) defined in Multicast Group Security Architecture [RFC3740]. The data communications within the group (e.g., IP multicast packets) are protected by a key pushed to the Group Members (GMs) by the Group Controller/Key Server (GCKS).

G-IKEv2 conforms to "The Multicast Group Security Architecture" [RFC3740], "Multicast Extensions to the Security Architecture for the Internet Protocol" [RFC5374], and "Multicast Security (MSEC) Group Key Management Architecture" [RFC4046]. G-IKEv2 replaces "The Group Domain of Interpretation" [RFC6407], which defines a similar group key management protocol using IKEv1 [RFC2409] (since deprecated by IKEv2). When G-IKEv2 is used, group key management use cases can benefit from the simplicity, increased robustness, and cryptographic improvements of IKEv2 (see Appendix A of [RFC7296]).

G-IKEv2 is composed of two phases: registration and rekeying. In the registration phase, a GM contacts a GCKS to register to a group and to receive the necessary policy and the keying material to be able communicate with the other GMs in the group as well as with the GCKS. The rekeying phase allows the GCKS to periodically renew the keying material for both GM-to-GM communications as well as for communication between the GM and the GCKS.

G-IKEv2 defines two ways to perform registration. When a GM first contacts a GCKS, it uses the GSA_AUTH exchange (Section 2.3.1) to register to a group. This exchange happens after the IKE_SA_INIT exchange (similarly to the IKE_AUTH exchange in IKEv2) and results in establishing an IKE Security Association (SA) between the GM and the GCKS. During this exchange, the GCKS authenticates and authorizes the GM and then pushes policy and keys used by the group to the GM. The second new exchange type is the GSA_REGISTRATION exchange (Section 2.3.2), which can be used by the GM within the already-established IKE SA with the GCKS (e.g., for registering to another group).

Refreshing the group keys can be performed either in a unicast mode via the GSA_INBAND_REKEY exchange (Section 2.4.2) performed over a specific IKE SA between a GM and a GCKS or in a multicast mode with the GSA_REKEY pseudo-exchange (Section 2.4.1) when new keys are being distributed to all GMs.

Large and small groups may use different sets of these mechanisms. When a large group of devices are communicating, the GCKS is likely to use the GSA_REKEY message for efficiency. This is shown in Figure 1, where multicast communications are indicated with a double line.

> Note: For clarity, IKE_SA_INIT is omitted from Figures 1 and 2.

```
                      +--------+
       +----IKEv2---->|  GCKS  |<----IKEv2----+
       |              +--------+              |
       |               ||    ^                |
       |               ||    |                |
       |               || GSA_AUTH            |
       |               ||   or                |
       |               || GSA_REGISTRATION    |
       |               ||    |                |
    GSA_AUTH           ||  IKEv2           GSA_AUTH
      or               ||    |               or
GSA_REGISTRATION   GSA_REKEY |         GSA_REGISTRATION
       |               ||    |                |
       |    *==========**================*    |
       |   ||          ||    |           ||   |
       v   \/          \/    v           \/   v
     +-------+        +--------+        +-------+
     |  GM   |  ...   |   GM   |  ...   |  GM   |
     +-------+        +--------+        +-------+
        ||                ||               ||
         *=====ESP/AH=====**=====ESP/AH====*
```

Figure 1: G-IKEv2 Used in Large Groups

Alternatively, a small group may simply use the GSA_AUTH or GSA_REGISTRATION as registration protocols, where the GCKS issues rekeys using the GSA_INBAND_REKEY within the same IKE SA.

```
   GSA_AUTH or GSA_REGISTRATION, GSA_INBAND_REKEY
 +--------------------IKEv2----------------------+
 |                                               |
 |   GSA_AUTH or GSA_REGISTRATION,               |
 |         GSA_INBAND_REKEY                      |
 |   +-----------IKEv2-------------+             |
 |   |                             |             |
 |   |GSA_AUTH or GSA_REGISTRATION,|             |
 |   |      GSA_INBAND_REKEY       |             |
 |   |   +--IKEv2-+                |             |
 v   v   v        v                v             v
+---------+    +----+           +----+        +----+
| GCKS/GM |    | GM |           | GM |        | GM |
+---------+    +----+           +----+        +----+
     ||           ||               ||            ||
      *==ESP/AH==**=====ESP/AH====**===ESP/AH===*
```

Figure 2: G-IKEv2 Used in Small Groups

A combination of these approaches is also possible. For example, the GCKS may use more robust GSA_INBAND_REKEY to provide keys for some GMs (for example, those acting as senders in the group) and GSA_REKEY for the rest.

> Note: GCKS may also be a GM (as shown in Figure 2).

IKEv2 message semantics are preserved in that all communications consist of message request-response pairs. The exception to this rule is the GSA_REKEY pseudo-exchange, which is a single message delivering group updates to the GMs.

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. Terminology

It is assumed that readers are familiar with the IPsec architecture [RFC4301] and its extension for multicast [RFC5374]. This document defines an extension to the IKEv2 protocol [RFC7296] and skips many of its details. The notation and conventions from [RFC7296] are used for describing G-IKEv2 payloads and exchanges.

The following key terms are used throughout this document (mostly borrowed from [RFC3740], [RFC5374], and [RFC6407]).

Group:
A set of IPsec devices that communicate to each other using multicast.

Group Member (GM):
An IPsec device that belongs to a group. A GM is authorized to be a group sender and/or a group receiver.

Group Receiver:
A GM that is authorized to receive packets sent to a group by a group sender.

Group Sender:
A GM that is authorized to send packets to a group.

Group Key Management (GKM) Protocol:
A key management protocol used by a GCKS to distribute IPsec Security Association policy and keying material. A GKM protocol is needed because a group of IPsec devices require the same SAs. For example, when an IPsec SA describes an IP multicast destination, the sender and all receivers need to have the group SA.

Group Controller/Key Server (GCKS):
A Group Key Management (GKM) protocol server that manages IPsec state for a group. A GCKS authenticates and provides the IPsec SA policy and keying material to GMs.

Data-Security SA:
: A multicast SA between each multicast sender and the group's receivers. The Data-Security SA protects data between member senders and member receivers. One or more SAs are required for the multicast transmission of data messages from the group sender to other GMs. This specification relies on Encapsulating Security Payload (ESP) and Authentication Header (AH) as protocols for Data-Security SAs.

Rekey SA:
: A single multicast SA between the GCKS and all of the GMs. This SA is used for multicast transmission of key management messages from the GCKS to all GMs.

Group SA:
: A Data-Security SA or Rekey SA that is shared as part of group policy.

Group Security Association (GSA):
: A collection of Data-Security SAs and Rekey SAs necessary for a GM to receive key updates. A GSA describes the working policy for a group. Refer to the MSEC Group Key Management Architecture [RFC4046] for additional information.

Traffic Encryption Key (TEK):
: The symmetric cipher key used in a Data-Security SA (e.g., IPsec ESP) to protect traffic.

Key Encryption Key (KEK):
: The symmetric key (or a set of keys) used in a Rekey SA to protect its messages. The set of keys may include keys for encryption and authentication, as well as keys for key wrapping.

Key Wrap Key (KWK):
: The symmetric cipher key used to protect another key.

Group-Wide (GW) policy:
: Group policy not related to a particular SA.

Activation Time Delay (ATD):
: Defines how long group senders should wait after receiving new SAs before sending traffic over them.

Deactivation Time Delay (DTD):
: Defines how long GMs should wait after receiving a request to delete Data-Security SAs before actually deleting them.

Sender-ID:
: A unique identifier of a group sender in the context of an active GSA used to form the Initialization Vector (IV) in counter-based cipher modes.

Logical Key Hierarchy (LKH):
: A group management method defined in Section 5.4 of [RFC2627].

2. G-IKEv2 Protocol

G-IKEv2 is an extension to the IKEv2 protocol [RFC7296] that provides group authorization, secure policy, and keys download from the GCKS to GMs.

2.1. G-IKEv2 Integration into the IKEv2 Protocol

G-IKEv2 is compatible with most IKEv2 extensions defined so far (see Section 6 for details). In particular, it is assumed that, if necessary, the IKE_INTERMEDIATE exchanges [RFC9242] may be utilized while establishing the registration SA. It is also believed that future IKEv2 extensions will be possible to use with G-IKEv2. However, some IKEv2 extensions may require special handling when used with G-IKEv2.

2.1.1. G-IKEv2 Transport and Port

As an IKEv2 extension, G-IKEv2 **SHOULD** use the IKEv2 ports (500, 4500). G-IKEv2 **MAY** use TCP transport for the IKE SA used for registration (which is unicast), as defined in TCP Encapsulation of IKEv2 and IPsec [RFC9329]. G-IKEv2 **MAY** also use UDP port 848, the same as Group Domain of Interpretation (GDOI) [RFC6407], because they serve a similar function. The version number in the IKE header distinguishes the G-IKEv2 protocol from the GDOI protocol [RFC6407].

Section 2.23 of [RFC7296] describes how IKEv2 supports paths with NATs. The G-IKEv2 registration SA doesn't create any unicast IPsec SAs; thus, if a NAT is present between the GM and the GCKS, there is no unicast ESP traffic to encapsulate in UDP. However, the actions described in this section regarding the IKE SA **MUST** be honored. The behavior of GMs and GCKS **MUST NOT** depend on the port used to create the initial IKE SA. For example, if the GM and the GCKS used UDP port 848 for the IKE_SA_INIT exchange, they will operate the same as if they had used UDP port 500.

2.2. G-IKEv2 Payloads

In the following descriptions, the payloads contained in the G-IKEv2 messages are indicated by names as listed below.

Notation	Payload	Defined in
AUTH	Authentication	[RFC7296]
CERT	Certificate	[RFC7296]
CERTREQ	Certificate Request	[RFC7296]
D	Delete	[RFC7296]
GSA	Group Security Association	Section 4.4

Notation	Payload	Defined in
HDR	IKE header (not a payload)	[RFC7296]
IDg	Group Identification	Section 4.2
IDi	Identification - Initiator	[RFC7296]
IDr	Identification - Responder	[RFC7296]
KD	Key Download	Section 4.5
KE	Key Exchange	[RFC7296]
Ni, Nr	Nonce	[RFC7296]
N	Notify	[RFC7296]
SA	Security Association	[RFC7296]
SAg	Security Association - GM Supported Transforms	Section 4.3
SK	Encrypted and Authenticated (also known as Encrypted)	[RFC7296]

Table 1: Payloads Used in G-IKEv2

Payloads defined as part of other IKEv2 extensions **MAY** also be included in these messages. Payloads that may optionally appear in G-IKEv2 messages will be shown in brackets, such as [CERTREQ].

G-IKEv2 defines several new payloads not used in IKEv2:

Group Identification (IDg):
: The GM requests the GCKS for membership into the group by sending its IDg payload.

Security Association - GM Supported Transforms (SAg):
: The GM optionally sends supported transforms so that GCKS may select a policy appropriate for all members of the group (which is not negotiated, unlike SA parameters in IKEv2).

Group Security Association (GSA):
: The GCKS sends the group policy to the GM using this payload.

Key Download (KD):
: The GCKS sends the keys and the security parameters to the GMs using this payload.

The details of the contents of each payload are described in Section 4.

2.3. G-IKEv2 Member Registration and Secure Channel Establishment

Initial registration is combined with establishing a secure connection between the entity seeking registration and the GCKS. This process consists of a minimum of two exchanges, IKE_SA_INIT and GSA_AUTH; member registration may have a few more messages exchanged if the Extensible Authentication Protocol (EAP) method, cookie challenge (for DoS protection), negotiation of key exchange method, or IKEv2 extensions based on the IKEv2 Intermediate Exchange [RFC9242] are used. Each exchange consists of request/response pairs. The first exchange, called IKE_SA_INIT, is defined in IKEv2 [RFC7296]. This exchange negotiates cryptographic algorithms, exchanges nonces, and computes a shared key between the GM and the GCKS. In addition to the cryptographic algorithms negotiated for use in IKEv2 SA, a key wrap algorithm is also negotiated in this exchange by means of a new "Key Wrap Algorithm" transform. See Section 4.5.4 for details.

The second exchange, called GSA_AUTH, is similar to the IKEv2 IKE_AUTH exchange [RFC7296]. It authenticates the previously exchanged messages and exchanges identities and certificates. The GSA_AUTH messages are encrypted and integrity protected with keys established through the previous exchanges, so the identities are hidden from eavesdroppers and all fields in all the messages are authenticated. The GCKS authorizes GMs to be allowed into the group as part of the GSA_AUTH exchange. Once the GCKS accepts a GM to join a group, it will provide the GM with the data-security keys (TEKs) and/or a group key encrypting key (KEK) as part of the GSA_AUTH response message.

The established secure channel between the GM and the GCKS is in fact IKE SA (as defined in [RFC7296]) and is referred to as such throughout this document. However, it is **NOT RECOMMENDED** to use this IKE SA for the purpose of creating unicast Child SAs between the GM and the GCKS since authentication requirements for group admission and for unicast communication may differ. In addition, the life cycle of this IKE SA is determined by the GCKS and this SA can be deleted at any time.

2.3.1. GSA_AUTH Exchange

The GSA_AUTH exchange is used to authenticate the previous exchanges and exchange identities and certificates. G-IKEv2 also uses this exchange for GM registration and authorization.

The GSA_AUTH exchange is similar to the IKE_AUTH exchange with the difference that its goal is to establish a multicast Data-Security SA(s) and optionally provide GM with the keys for a Rekey SA. The set of payloads in the GSA_AUTH exchange is slightly different because policy is not negotiated between the GM and the GCKS; instead, it is provided by the GCKS for the GM. Also note that GSA_AUTH has its own exchange type, which is different from the IKE_AUTH exchange type.

> Note: Due to the similarities between IKE_AUTH and GSA_AUTH, most IKEv2 extensions to the IKE_AUTH exchange (like secure password authentication [RFC6467]) can also be used with the GSA_AUTH exchange.

```
 Initiator (GM)                                  Responder (GCKS)
--------------------                            ------------------
 HDR, SK{IDi, [CERT,] [CERTREQ,] [IDr,]
      AUTH, IDg, [SAg,] [N(GROUP_SENDER),] [N]}   -->
```

Figure 3: GSA_AUTH Request

A GM initiates a GSA_AUTH request to join a group indicated by the IDg payload. The GM may include an SAg payload declaring which Transforms it is willing to accept. A GM that intends to act as group sender **MUST** include a Notify payload status type of GROUP_SENDER, which enables the GCKS to provide any additional policy necessary by group senders.

```
 Initiator (GM)                 Responder (GCKS)
--------------------           ------------------
                          <--   HDR, SK{IDr, [CERT,]
                                     AUTH, GSA, KD, [N]}
```

Figure 4: GSA_AUTH Normal Response

The GCKS responds with IDr, optional CERT, and AUTH payloads with the same meaning as in IKE_AUTH. It also informs the GM of the cryptographic policies of the group in the GSA payload and the key material in the KD payload.

Possible errors should be handled in accordance with Section 2.21.2 of [RFC7296]. In addition to the IKEv2 error handling, the GCKS can reject the registration request when the IDg is invalid or authorization fails, etc. In these cases (see Section 4.7), the GSA_AUTH response will not include the GSA and KD but will include a Notify payload indicating errors. If a GM included an SAg payload and the GCKS chooses to evaluate it and detects that the GM cannot support the security policy defined for the group, then the GCKS returns the NO_PROPOSAL_CHOSEN notification. Other types of error notifications can be INVALID_GROUP_ID, AUTHORIZATION_FAILED, or REGISTRATION_FAILED.

```
 Initiator (GM)                   Responder (GCKS)
--------------------             ------------------
                           <--   HDR, SK{IDr, [CERT,] AUTH, N}
```

Figure 5: GSA_AUTH Error Response for Group-Related Errors

If the GSA_AUTH exchange is completed successfully but the GM finds that the policy sent by the GCKS is unacceptable, the member **SHOULD** inform the GCKS about this by initiating the GSA_REGISTRATION exchange with the IDg payload and the NO_PROPOSAL_CHOSEN notification (see Figure 8).

2.3.2. GSA_REGISTRATION Exchange

Once the IKE SA between the GM and the GCKS is established, the GM can use it for other registration requests if needed. In this scenario, the GM will use the GSA_REGISTRATION exchange. Payloads in the exchange are generated and processed as defined in Section 2.3.1.

```
 Initiator (GM)                    Responder (GCKS)
--------------------             ------------------
 HDR, SK{IDg, [SAg,]
      [N(GROUP_SENDER),] [N]} -->
                             <--  HDR, SK{GSA, KD, [N]}
```

Figure 6: GSA_REGISTRATION Normal Exchange

As with GSA_AUTH exchange, the GCKS can reject the registration request when the IDg is invalid or authorization fails, or GM cannot support the security policy defined for the group (which can be concluded by the GCKS by evaluation of the SAg payload). In this case, the GCKS returns an appropriate error notification as described in Section 2.3.1.

```
 Initiator (GM)                     Responder (GCKS)
--------------------              ------------------
 HDR, SK{IDg, [SAg,]
      [N(GROUP_SENDER),] [N]} -->
                            <--    HDR, SK{N}
```

Figure 7: GSA_REGISTRATION Error Exchange

This exchange can also be used if the GM finds that the policy sent by the GCKS is unacceptable or wants to leave the group for some reason. The GM **SHOULD** notify the GCKS by sending IDg and the Notify type NO_PROPOSAL_CHOSEN or REGISTRATION_FAILED as shown below. In this case, the GCKS **MUST** remove the GM from the group denoted in IDg.

```
 Initiator (GM)                      Responder (GCKS)
--------------------               ------------------
 HDR, SK{IDg, N}      -->
                         <--        HDR, SK{}
```

Figure 8: GM Reporting Errors in GSA_REGISTRATION Exchange

2.3.3. GM Registration Operations

A GM requesting registration contacts the GCKS using the IKE_SA_INIT exchange. This exchange is unchanged from IKE_SA_INIT in the IKEv2 protocol. The IKE_SA_INIT exchange may optionally be followed by one or more of the IKE_INTERMEDIATE exchanges if the GM and the GCKS negotiated use of IKEv2 extensions based on this exchange.

Next, the GM sends the GSA_AUTH request message with the IKEv2 payloads from IKE_AUTH (without the SAi2, TSi, and TSr payloads) along with the Group ID informing the GCKS of the group the GM wishes to join. A GM intending to emit data traffic **MUST** send a GROUP_SENDER notification. The GROUP_SENDER notification not only signifies that it is a sender but provides the GM the ability to request Sender-ID values in case the Data-Security SA supports a counter-mode cipher. Section 2.5.1 includes guidance on requesting Sender-ID values.

A GM may be limited in the Transforms IDs that it is able or willing to use and may find it useful to inform the GCKS which Transform IDs it is willing to accept for different security protocols by including the SAg payload into the request message. Proposals for Rekey SA and for Data-Security (AH [RFC4302] and/or ESP [RFC4303]) SAs may be included into SAg. Proposals for Rekey SA are identified by a new Security Protocol Identifier GIKE_UPDATE with the value 6. Each Proposal contains a list of Transforms that the GM is able and willing to support for that protocol. Valid Transform Types depend on the protocol (AH, ESP, GIKE_UPDATE) and are defined in Table 2. Other Transform Types **SHOULD NOT** be included as they will be ignored by the GCKS. The Security Parameter Index (SPI) length of each Proposal in an SAg is set to zero, and thus the SPI field is empty. The GCKS **MUST NOT** use SPI length and SPI fields in the SAg payload.

Generally, a single Proposal for each protocol (GIKE_UPDATE, AH/ESP) will suffice. Because the transforms are not negotiated, the GM simply alerts the GCKS to restrictions it may have. In particular, the restriction from Section 3.3 of [RFC7296] that Authenticated Encryption with Associated Data (AEAD) and non-AEAD transforms not be combined in a single proposal doesn't hold when the SAg payload is being formed. However, if the GM has restrictions on the combination of algorithms, this can be expressed by sending several proposals.

The Proposal Num field in the Proposal substructure is treated specially in the SAg payload: it allows a GM to indicate that algorithms used in Rekey SA and in Data-Security (AH and/or ESP) SAs are dependent. In particular, Proposals for different protocols having the same value in the Proposal Num field are treated as a set so that if GCKS uses transforms from one of such Proposal for one protocol, then it **MUST** only use transforms from one of the Proposals with the same value in the Proposal Num field for other protocols. For example, a GM may support algorithms X and Y for both Rekey and Data-Security SAs, but with a restriction that if X is used in Rekey SAs, then only X can be used in Data-Security SAs, and the same for Y. Use of the same value in the Proposal Num field of different proposals indicates that the GM expects these proposals to be used in conjunction with each other. In the simplest case when no dependency between transforms exists, all Proposals in the SAg payload will have the same value in the Proposal Num field.

Although the SAg payload is optional, it is **RECOMMENDED** that the GM include this payload into the GSA_AUTH request to allow the GCKS to select an appropriate policy.

A GM **MAY** also indicate the support for IPcomp by including one or more the IPCOMP_SUPPORTED notifications along with the SAg payload in the request. The Compression Parameter Index (CPI) in these notifications is set to zero and **MUST** be ignored by the GCKS.

Upon receiving the GSA_AUTH response, the GM parses the response from the GCKS authenticating the exchange using the IKEv2 method, then processes the GSA and KD payloads.

The GSA payload contains the security policy and cryptographic protocols used by the group. This policy describes zero or more Data-Security SAs (TEK), zero or one Rekey SA (KEK), and zero or one GW policy (although at least one TEK or KEK policy **MUST** be Present). If the policy in the GSA payload is not acceptable to the GM, it **SHOULD** notify the GCKS by initiating a GSA_REGISTRATION exchange with a NO_PROPOSAL_CHOSEN Notify payload (see Section 2.3.2). Note that this should normally not happen if the GM includes the SAg payload in the GSA_AUTH request and the GCKS takes it into account. Finally, the KD payload is parsed, providing the keying material for the TEK and/or KEK. The KD payload contains a list of key bags, where each key bag includes the keying material for SAs distributed in the GSA payload. Keying material is matched by comparing the SPIs in the key bags to SPIs previously included in the GSA payloads. Once TEK keys and policy are matched, the GM provides them to the data-security subsystem, and it is ready to send or receive packets matching the TEK policy.

If the GM is not a sender for a received Data-Security SA, then it **MUST** install this SA only in the inbound direction. If the GM is a sender for a received Data-Security SA, and it is not going to receive back the data it sends, then it **MUST** install this SA only in the outgoing direction.

If the first Message ID the GM should expect to receive is non-zero, the GSA KEK policy includes the attribute GSA_INITIAL_MESSAGE_ID with the expected non-zero value. The value of the attribute **MUST** be checked by a GM against any previously received Message ID for this group. If it is less than the previously received number, it should be considered stale and **MUST** be ignored. This could happen if two GSA_AUTH exchanges happened in parallel and the Message ID changed. This attribute is used by the GM to prevent GSA_REKEY message replay attacks. The first GSA_REKEY message that the GM receives from the GCKS will have a Message ID greater than or equal to the Message ID received in the GSA_INITIAL_MESSAGE_ID attribute.

GMs **MUST** install the Rekey SA only in the inbound direction.

Once a GM successfully registers to the group, it **MUST** replace any information related to this group (policy, keys) that it might have as a result of a previous registration with a new one.

Once a GM has received the GIKE_UPDATE policy during a registration, the IKE SA **MAY** be closed. By convention, the GCKS closes the IKE SA; the GM **SHOULD NOT** close it. The GCKS **MAY** choose to keep the IKE SA open for inband rekey, especially for small groups. If inband rekey is used, then the initial IKE SA can be rekeyed by any side with the standard IKEv2 mechanism described in Section 1.3.2 of [RFC7296]. If for some reason the IKE SA is closed and no GIKE_UPDATE policy is received during the registration process, the GM **MUST** consider itself excluded from the group. To continue participating in the group, the GM needs to re-register.

2.3.4. GCKS Registration Operations

A G-IKEv2 GCKS listens for incoming requests from GMs. When the GCKS receives an IKE_SA_INIT request, it selects an IKE proposal and generates a nonce and Diffie-Hellman (DH) to include in the IKE_SA_INIT response.

Upon receiving the GSA_AUTH request, the GCKS authenticates the GM via the GSA_AUTH exchange. The GCKS then authorizes the GM according to group policy before preparing to send the GSA_AUTH response. If the GCKS fails to authorize the GM, it responds with an AUTHORIZATION_FAILED notification. The GCKS may also respond with an INVALID_GROUP_ID notification if the requested group is unknown to the GCKS or with an REGISTRATION_FAILED notification if there is a problem with the requested group (e.g., if the capacity of the group is exceeded).

The GSA_AUTH response will include the group policy in the GSA payload and keys in the KD payload. If the GCKS policy includes a group rekey option and the initial Message ID value the GCKS will use when sending the GSA_REKEY messages to the GMs is non-zero, then this value is specified in the GSA_INITIAL_MESSAGE_ID attribute. This Message ID is used to prevent GSA_REKEY message replay attacks and will be increased each time a GSA_REKEY message is sent to the group. The GCKS data traffic policy is included in the GSA TEK and keys are included in the KD TEK. The GW policy **MAY** also be included to provide the Activation Time Delay (ATD) and/or Deactivation Time Delay (DTD) (Section 4.4.3.1.1) to specify activation and deactivation delays for SAs generated from the TEKs. If the GM has indicated that it is a sender of data traffic and one or more Data-Security SAs distributed in the GSA payload included a counter mode of operation, the GCKS responds with one or more Sender-ID values (see Section 2.5).

Multicast Extensions to the Security Architecture [RFC5374] defines two modes of operation for multicast Data-Security SAs: transport mode and tunnel mode with address preservation. In the latter case, outer source and destination addresses are taken from the inner IP packet. The mode of operation for the Data-Security SAs is determined by the presence of the USE_TRANSPORT_MODE notification in the GCKS's response message of the registration exchange. If it is present, then SAs are created in transport mode; otherwise, SAs are created in tunnel mode. If multiple Data-Security SAs are being created in a single registration exchange, then all of them will have the same mode of operation.

If the GCKS receives a GSA_REGISTRATION exchange with a request to register a GM to a group, the GCKS will need to authorize the GM with the new group (IDg) and respond with the corresponding group policy and keys. If the GCKS fails to authorize the GM, it will respond with the AUTHORIZATION_FAILED notification. The GCKS may also respond with an INVALID_GROUP_ID or REGISTRATION_FAILED notify messages for the reasons described above.

If a GM includes an SAg in its GSA_AUTH or GSA_REGISTRATION request, the GCKS may evaluate it according to an implementation-specific policy.

- The GCKS could evaluate the list of Transforms and compare it to its current policy for the group. If the GM did not include all of the ESP, AH, or GIKE_UPDATE Transforms that match the current group policy or the capabilities of all other currently active GMs, then the GCKS **SHOULD** return a NO_PROPOSAL_CHOSEN notification. Alternatively, the GCKS can change the group policy as defined below.
- The GCKS could store the list of Transforms with the goal of migrating the group policy from the current set of transforms to a different one once all of the GMs indicate that they can support transforms from the new set.
- The GCKS could store the list of Transforms and adjust the current group policy based on the capabilities of the devices as long as they fall within the acceptable security policy of the GCKS.

Depending on its policy, the GCKS may have no further need for the IKE SA (e.g., it does not plan to initiate a GSA_INBAND_REKEY exchange). If the GM does not initiate another registration exchange or Notify (e.g., NO_PROPOSAL_CHOSEN) and the GCKS is not intended to use the SA, then the GCKS **SHOULD** close the IKE SA to save resources after a short period of time.

2.4. Group Maintenance Channel

The GCKS is responsible for rekeying the secure group per the group policy. Rekeying is an operation whereby the GCKS provides replacement TEK(s) and/or KEK, deleting TEK(s), and/or excluding GMs. The GCKS may initiate a rekey message if group membership and/or policy has changed or if the keys are about to expire. Two forms of group maintenance channels are provided in G-IKEv2 to push new policy to GMs.

GSA_REKEY:

The GSA_REKEY is a pseudo-exchange, consisting of a one-way IKEv2 message sent by the GCKS where the rekey policy is delivered to GMs using IP multicast as a transport. This method is valuable for large and dynamic groups and where policy may change frequently and a scalable rekey method is required. When the GSA_REKEY is used, the IKE SA protecting the member registration exchanges is usually terminated and GMs await policy changes from the GCKS via the GSA_REKEY messages.

GSA_INBAND_REKEY:

The GSA_INBAND_REKEY is a normal IKEv2 exchange using the IKE SA that was set up to protect the member registration exchange. This exchange allows the GCKS to rekey without using an independent GSA_REKEY pseudo-exchange. The GSA_INBAND_REKEY exchange provides a reliable policy delivery and is useful when G-IKEv2 is used with a small group of cooperating devices.

Depending on its policy, the GCKS **MAY** combine these two methods. For example, the GCKS may use the GSA_INBAND_REKEY to deliver a key to the GMs in the group acting as senders (as this would provide reliable keys delivery) and the GSA_REKEY for the rest of the GMs.

2.4.1. GSA_REKEY

The GCKS initiates the G-IKEv2 rekey by sending a protected message to the GMs, usually using IP multicast. Since the Rekey messages do not require responses and are sent to multiple GMs, the windowing mechanism described in Section 2.3 of [RFC7296] **MUST NOT** be used for the Rekey messages. The GCKS rekey message replaces the current rekey GSA KEK or KEK array (e.g., in the case of LKH) and/or creates new Data-Security SAs. The GM_SENDER_ID attribute in the Key Download payload (defined in Section 4.5.3.3) **MUST NOT** be part of the Rekey Exchange, as this is sender-specific information and the Rekey Exchange is group specific. The GCKS initiates the GSA_REKEY pseudo-exchange as following:

```
  GMs (Receivers)               GCKS (Sender)
 -----------------            ---------------
                         <--  HDR, SK{GSA, KD, [N,] [AUTH]}
```

Figure 9: GSA_REKEY Pseudo-Exchange

HDR is defined in Section 4.1. While GSA_REKEY reuses the IKEv2 header, the "IKE SA Initiator's SPI" and the "IKE SA Responder's SPI" fields are treated as a single field with a length of 16 octets containing the SPI of a Rekey SA. The value for this field is provided by the GCKS in the GSA payload (see Section 4.4.2). The Message ID in this message will start with the value the GCKS sent to the GMs in the attribute GSA_INITIAL_MESSAGE_ID or from zero if this attribute wasn't sent. The Message ID will be incremented each time a new GSA_REKEY message is sent to the GMs.

The GSA payload contains the current policy for rekey and Data-Security SAs. The GSA may contain a new Rekey SA and/or a new Data-Security SA(s) (Section 4.4).

The KD payload contains the keys for the policy included in the GSA. If one or more Data-Security SAs are being refreshed in this rekey message, the IPsec keys are updated in the KD, and/or if the Rekey SA is being refreshed in this rekey message, the rekey Key or the LKH KEK array (e.g., in case of LKH) is updated in the KD payload.

A Delete payload **MAY** be included to instruct the GM to delete existing SAs. See Section 4.6 for more detail.

The AUTH payload **MUST** be included to authenticate the GSA_REKEY message if the authentication method is based on public key signatures and **MUST NOT** be included if authentication is implicit. In the latter case, the fact that a GM can decrypt the GSA_REKEY message and verify its Integrity Check Value (ICV) proves that the sender of this message knows the current KEK, thus authenticating the sender as a member of the group. Note that implicit authentication doesn't provide source origin authentication. For this reason, using implicit authentication for GSA_REKEY is **NOT RECOMMENDED** unless source origin authentication is not required (for example, in a small group of highly trusted GMs). See more about authentication methods in Section 4.4.2.1.1.

During GM registration, the GCKS sends the authentication key in the KD payload, the AUTH_KEY attribute, which the GM uses to authenticate the key server. Before the current authentication key expires, the GCKS will send a new AUTH_KEY to the GMs in a GSA_REKEY message. The authentication key that is sent in the rekey message may not be the same as the authentication key sent during the GM registration. If implicit authentication is used, then AUTH_KEY **MUST NOT** be sent to GMs.

2.4.1.1. GSA_REKEY Message Authentication

The content of the AUTH payload generally depends on the authentication method from the Group Controller Authentication Method (GCAUTH) transform (Section 4.4.2.1.1). This specification defines the use of only one authentication method, Digital Signature, and the AUTH payload contains a digital signature calculated over the content of the not-yet-encrypted GSA_REKEY message.

The digital signing is applied to the concatenation of two chunks: A and P. Chunk A starts with the first octet of the G-IKEv2 header (not including prepended four octets of zeros, if port 4500 is used) and continues to the last octet of the Encrypted Payload header. Chunk P consists of the not-yet-encrypted content of the Encrypted payload, excluding the Initialization Vector, the Padding, the Pad Length, and the Integrity Checksum Data fields (see Section 3.14 of [RFC7296] for the description of the Encrypted payload). In other words, chunk P is the inner payloads of the Encrypted payload in plaintext form. Figure 10 illustrates the layout of chunks P and A in the GSA_REKEY message.

Before the calculation of the AUTH payload, the inner payloads of the Encrypted payload must be fully formed and ready for encryption except for the content of the AUTH payload. The AUTH payload must have correct values in the Payload header, the Auth Method, and the RESERVED fields. The Authentication Data field is zeroed, but the ASN.1 Length and the AlgorithmIdentifier fields must be properly filled in; see Signature Authentication in [RFC7427].

For the purpose of the AUTH payload calculation, the Length field in the IKE header and the Payload Length field in the Encrypted Payload header are adjusted so that they don't count the lengths of Initialization Vector, Integrity Checksum Data, and Padding (along with Pad Length field). In other words, the Length field in the IKE header (denoted as AdjustedLen in Figure 10) is set to the sum of the lengths of A and P, and the Payload Length field in the Encrypted Payload header (denoted as AdjustedPldLen in Figure 10) is set to the length of P plus the size of the Payload header (four octets).

The input to the digital signature algorithm that computes the content of the AUTH payload can be described as:

```
DataToAuthenticate = A | P
GsaRekeyMessage = GenIKEHDR | EncPayload
GenIKEHDR = [ four octets 0 if using port 4500 ] | AdjustedIKEHDR
AdjustedIKEHDR =  SPIi | SPIr |  . . . | AdjustedLen
EncPayload = AdjustedEncPldHdr | IV | InnerPlds | Pad | PadLen | ICV
AdjustedEncPldHdr = NextPld | C | RESERVED | AdjustedPldLen
A = AdjustedIKEHDR | AdjustedEncPldHdr
P = InnerPlds
```

```
                     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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ^ ^
|                       IKE SA Initiator's SPI                  | | |
|                                                               | | |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ I |
|                       IKE SA Responder's SPI                  | K |
|                                                               | E |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+   |
|  Next Payload | MjVer | MnVer | Exchange Type |     Flags     | h A
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ d |
|                           Message ID                          | r |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ | |
|                          AdjustedLen                          | | |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ x |
| Next Payload  |C|  RESERVED   |         AdjustedPldLen        | | |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ | v
|                                                               | |
~                     Initialization Vector                     ~ E
|                                                               | n
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ c ^
|                                                               | r |
~             Inner Payloads (not yet encrypted)                ~   P
|                                                               | P |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ l v
~              Padding (0-255 octets)           |  Pad Length   | d
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |
|                                                               | |
~                    Integrity Checksum Data                    ~ |
|                                                               | |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ v
```

Figure 10: Data to Authenticate in the GSA_REKEY Messages

The authentication data is calculated using the authentication algorithm from the Group Controller Authentication Method transform (Section 4.4.2.1.1) and the current authentication key provided in the AUTH_KEY attribute (Section 4.5.3.2). The calculated authentication data is placed into the AUTH payload, the Length fields in the IKE header and the Encryption Payload header are restored, the content of the Encrypted payload is encrypted and the ICV is computed using the current KEK.

2.4.1.2. IKE Fragmentation

IKEv2 fragmentation [RFC7383] can be used to perform fragmentation of large GSA_REKEY messages; however, when the GSA_REKEY message is emitted as an IP multicast packet, there is a lack of response from the GMs. This has the following implications.

- Policy regarding the use of IKE fragmentation is implicit. If a GCKS detects that all GMs have negotiated support of IKE fragmentation in IKE_SA_INIT, then it **MAY** use IKE fragmentation on large GSA_REKEY messages.
- The GCKS must always use IKE fragmentation based on a preconfigured fragmentation threshold, as there is no way to check if fragmentation is needed by first sending unfragmented messages and waiting for response. Section 2.5.1 of [RFC7383] contains recommendations on selecting the fragmentation threshold.
- The Path MTU (PMTU) mechanism, defined in Section 2.5.2 of [RFC7383], cannot be used due to lack of GSA_REKEY response messages.

The calculation of authentication data **MUST** be applied to whole messages only before possible IKE Fragmentation. If the message was received in fragmented form, it should be reconstructed before verifying its authenticity as if it were received unfragmented. The RESERVED field in the reconstructed Encrypted Payload header **MUST** be set to the value of the RESERVED field in the Encrypted Fragment payload header from the first fragment (with the Fragment Number equal to 1).

2.4.1.3. GSA_REKEY GCKS Operations

The GCKS builds the rekey message with a Message ID value that is one greater than the value included in the previous rekey message. The first message sent over a new Rekey SA **MUST** use a Message ID of 0. The GSA, KD, and N payloads follow with the same characteristics as in the GSA Registration exchange. The AUTH payload (if present) is created as defined in Section 2.4.1.1.

Because GSA_REKEY messages are not acknowledged and could be discarded by the network, one or more GMs may not receive the new policy. To mitigate such lost messages, during a rekey event, the GCKS may transmit several copies of an encrypted GSA_REKEY message with the new policy. The (encrypted) retransmitted messages **MUST** be bitwise identical and should be sent within a short time interval (a few seconds) to ensure that the SA lifetime calculations would not be substantially skewed for the GMs that would receive different copies of the messages.

GCKS may also include one or several GSA_NEXT_SPI attributes specifying SPIs for the prospected rekeys so that listening GMs are able to detect lost rekey messages and recover from this situation. See Section 4.4.2.2.3 for more detail.

2.4.1.4. GSA_REKEY GM Operations

When a GM receives the rekey message from the GCKS, it decrypts the message and verifies its integrity using the current KEK. If the AUTH payload is present in the decrypted message, then the GM validates authenticity of the message using the key retrieved in a previous G-IKEv2 exchange. Then the GM verifies the Message ID and processes the GSA and KD payloads. The GM then installs the new Data-Security SA(s) and/or a new Rekey SA. The parsing of the payloads is identical to the parsing done in the registration exchange.

Replay protection is achieved by a GM rejecting a GSA_REKEY message that has a Message ID smaller than the current Message ID that the GM is expecting. The GM expects the Message ID in the first GSA_REKEY message it receives to be equal to or greater than the Message ID it receives in the GSA_INITIAL_MESSAGE_ID attribute. Note that if the GSA_INITIAL_MESSAGE_ID attribute is not received for the Rekey SA, the GM **MUST** assume zero as the first expected Message ID. The GM expects the Message ID in subsequent GSA_REKEY messages to be greater than the last valid GSA_REKEY message ID it received.

This specification assumes that the GSA_REKEY messages are sent with intervals that are significantly greater than typical network packet reordering intervals.

If the GSA payload includes a Data-Security SA using cipher in a counter-mode of operation and the receiving GM is a sender for that SA, the GM uses its current Sender-ID value with the Data-Security SAs to create counter-mode nonces. If it is a sender and does not hold a current Sender-ID value (for example, when no counter-mode is employed for other Data-Security SAs), it **MUST NOT** install the Data-Security SAs. It **MUST** initiate a re-registration to the GCKS in order to obtain a Sender-ID value (along with the current group policy).

Once a new Rekey SA is installed as a result of a GSA_REKEY message, the current Rekey SA (over which the message was received) **MUST** be silently deleted after waiting the DEACTIVATION_TIME_DELAY interval regardless of its expiration time. If the message includes a Delete payload for an existing Data-Security SA, then after installing a new Data-Security SA, the old one (identified by the Protocol and SPI fields in the Delete payload) **MUST** be silently deleted after waiting the DEACTIVATION_TIME_DELAY interval regardless of its expiration time.

If a Data-Security SA is not rekeyed yet and is about to expire (a "soft lifetime" expiration is described in Section 4.4.2.1 of [RFC4301]), the GM **SHOULD** initiate a registration to the GCKS. This registration serves as a request for current SAs and will result in the download of replacement SAs, assuming the GCKS policy has created them. A GM **SHOULD** also initiate a registration request if a Rekey SA is about to expire and not yet replaced with a new one.

2.4.2. GSA_INBAND_REKEY Exchange

When the IKE SA protecting the member registration exchange is maintained while a GM participates in the group, the GCKS can use the GSA_INBAND_REKEY exchange to individually provide policy updates to the GM.

```
 GM (Responder)                GCKS (Initiator)
----------------             ------------------
                      <--    HDR, SK{GSA, KD, [N]}
 HDR, SK{}            -->
```

Figure 11: GSA_INBAND_REKEY Exchange

Because this is a normal IKEv2 exchange, the HDR is treated as defined in IKEv2 [RFC7296].

2.4.2.1. GSA_INBAND_REKEY GCKS Operations

The GSA, KD, and N payloads are built in the same manner as in a registration exchange.

2.4.2.2. GSA_INBAND_REKEY GM Operations

The GM processes the GSA, KD, and N payloads in the same manner as if they were received in a registration exchange.

2.4.3. Deletion of SAs

There are occasions when the GCKS may want to signal to GMs to delete policy when the application sending data traffic has ended or if group policy has changed. Deletion of SAs is accomplished by sending the Delete Payload described in Section 3.11 of [RFC7296] as part of the GSA_REKEY pseudo-exchange as shown below.

```
GMs (Receivers)            GCKS (Sender)
----------------          ---------------
                    <--   HDR, SK{D, [N,] [AUTH]}
```

Figure 12: SA Deletion in GSA_REKEY

If GCKS has a unicast SA with a GM, then it can use the GSA_INBAND_REKEY exchange to delete SAs.

```
GM (Responder)                 GCKS (Initiator)
---------------              ------------------
                      <--    HDR, SK{D, [N]}
 HDR, SK{}            -->
```

Figure 13: SA Deletion in GSA_INBAND_REKEY

There may be circumstances where the GCKS may want to start over with a clean state, e.g., in case it runs out of available Sender-IDs. The GCKS can signal deletion of all the Data-Security SAs by sending a Delete payload with an SPI value equal to zero. For example, if the GCKS wishes to remove the Rekey SA and all the Data-Security SAs, the GCKS sends a Delete payload with an SPI of zero and a Protocol ID of AH or ESP, followed by another Delete payload with an SPI of zero and a Protocol ID of GIKE_UPDATE.

If a GM receives a Delete payload with zero SPI and a Protocol ID of GIKE_UPDATE, it means that the GM is excluded from the group. Such Delete payload may be received either in the GSA_REKEY pseudo-exchange or in the GSA_INBAND_REKEY exchange. In this situation, the GM **MUST** re-register if it wants to continue participating in this group. The registration is performed as described in Section 2.3. It is **RECOMMENDED** that a GM waits some randomly chosen time before initiating a registration request in this situation to avoid overloading the GCKS. This document doesn't specify the maximum delay, which is implementation-dependent, but it is believed that the order of seconds suits most situations. Note that if the unicast SA between the GM and the GCKS exists, then the GM may use the GSA_REGISTRATION exchange to re-register. However, after excluding a GM from the group, the GCKS **MAY** immediately delete the unicast SA with this GM (if any) if the credentials of this GM are revoked.

2.5. Counter-Based Modes of Operation

Several counter-based modes of operation have been specified for ESP (e.g., AES-CTR [RFC3686], AES-GCM [RFC4106], AES-CCM [RFC4309], ChaCha20-Poly1305 [RFC7634], and AES-GMAC [RFC4543]) and AH (e.g., AES-GMAC [RFC4543]). These counter-based modes require that no two senders in the group ever send a packet with the same IV using the same cipher key and mode. This requirement is met in G-IKEv2 when the following measures are taken:

- The GCKS distributes a unique key for each Data-Security SA.
- The GCKS uses the method described in [RFC6054], which assigns each sender a portion of the IV space by provisioning each sender with one or more unique Sender-ID values.

2.5.1. Allocation of Sender-ID

When at least one Data-Security SA included in the group policy includes a counter-based mode of operation, the GCKS automatically allocates and distributes one Sender-ID to each GM acting in the role of sender on the Data-Security SA. The Sender-ID value is used exclusively by the group sender to which it was allocated. The group sender uses the same Sender-ID for each Data-Security SA specifying the use of a counter-based mode of operation. A GCKS **MUST** distribute unique keys for each Data-Security SA, including a counter-based mode of operation in order to maintain unique key and nonce usage.

During registration, the group sender can choose to request one or more Sender-ID values. Requesting a value of 1 is not necessary since the GCKS will automatically allocate exactly one to the group sender. A group sender **MUST** request as many Sender-ID values matching the number of encryption modules in which it will be installing the TEKs in the outbound direction. Alternatively, a group sender **MAY** request more than one Sender-ID and use them serially. This could be useful when it is anticipated that the group sender will exhaust their range of Data-Security SA nonces using a single Sender-ID too quickly (e.g., before the time-based policy in the TEK expires).

When the group policy includes a counter-based mode of operation, a GCKS should use the following method to allocate Sender-ID values, which ensures that each Sender-ID will be allocated to just one group sender.

1. A GCKS maintains a Sender-ID counter, which records the Sender-IDs that have been allocated. Sender-IDs are allocated sequentially with zero as the first allocated value.
2. Each time a Sender-ID is allocated, the current value of the counter is saved and allocated to the group sender. The Sender-ID counter is then incremented in preparation for the next allocation.
3. When the GCKS specifies a counter-based mode of operation in the Data-Security SA, a group sender may request a count of Sender-IDs during registration in a Notify payload information of type SENDER. When the GCKS receives this request, it increments the Sender-ID counter once for each requested Sender-ID and distributes each Sender-ID value to the group sender. The GCKS should have a policy-defined upper bound for the number of Sender-ID values that it will return irrespective of the number requested by the GM.
4. A GCKS allocates new Sender-ID values for each registration operation by a group sender, regardless of whether the group sender had previously contacted the GCKS. In this way, the GCKS is not required to maintain a record of which Sender-ID values it had previously allocated to each group sender. More importantly, since the GCKS cannot reliably detect whether the group sender had sent data on the current group Data-Security SAs, it does not know what Data-Security counter-mode nonce values that a group sender has used. By distributing new Sender-ID values, the key server ensures that each time a conforming group sender installs a Data-Security SA, it will use a unique set of counter-based mode nonces.
5. When the Sender-ID counter maintained by the GCKS reaches its final Sender-ID value, no more Sender-ID values can be distributed. Before distributing any new Sender-ID values, the GCKS **MUST** exclude all GMs from the group as described in Section 2.4.3. This will result in the GMs performing re-registration, during which they will receive new Data-Security SAs and group senders will additionally receive new Sender-ID values. The new Sender-ID values can safely be used because they are only used with the new Data-Security SAs.

2.5.2. GM Usage of Sender-ID

A GM applies the Sender-ID to Data-Security SAs as follows:

- The most significant bits of the IV indicated in the GWP_SENDER_ID_BITS attribute (Section 4.4.3.1.2) are taken to be the Sender-ID field of the IV.
- The Sender-ID is placed in the least significant bits of the Sender-ID field, where any unused most significant bits are set to zero. If the Sender-ID value doesn't fit into the number of bits from the GWP_SENDER_ID_BITS attributes, then the GM **MUST** treat this as a fatal error and re-register to the group.

2.6. Replay Protection for Multicast Data-Security SAs

IPsec provides anti-replay service as part of its security services. With multicast extensions for IPsec, replay protection is not always possible to achieve (see Section 6.1 of [RFC3740]). In particular, if there are many group senders for a Data-Security SA, then each of them will independently increment the Sequence Number field in the ESP header (see Section 2.2 of [RFC4303] and Section 2.5 of [RFC4302]), thus making it impossible for the group receivers to filter out replayed packets. However, if there is only one group sender for a Data-Security SA, then it is possible to achieve replay protection with some restrictions (see Section 4.4.2.1.3). The GCKS **MAY** create several Data-Security SAs with the same traffic selectors allowing only a single group sender in each SA if it is desirable to get replay protection with multiple (but still a limited number) of group senders.

IPsec architecture assumes that whether anti-replay service is enabled or not is a local matter for an IPsec receiver. In other words, an IPsec sender always increments the Sequence Number field in the ESP/AH header and a receiver decides whether to check for replayed packets or not. Since it is known in some cases that the replay protection is not possible (like in an SA with many group senders), a new Transform ID "32-bit Unspecified Numbers" is defined for the Sequence Numbers (SNs) Transform Type. Using this Transform ID, the GCKS can inform GMs that the uniqueness of sequence numbers for a given SA is not guaranteed. The decision of whether to enable anti-replay service is still a local matter of a GM (in accordance with IPsec architecture).

The GCKS **MUST** include the Sequence Numbers transform in the GSA payload for every Data-Security SA. See Section 4.4.2.1.3 for more details.

When a Data-Security SA has a single sender, the GCKS **MUST** be configured to rekey the SA frequently enough so that the 32-bit sequence numbers do not wrap.

2.7. Encryption Transforms with Implicit IV

The "Transform Type 1 - Encryption Algorithm Transform IDs" IANA registry [IKEV2-IANA] defines several transforms with implicit IV. These transforms rely on ESP Sequence Numbers for constructing IV (see [RFC8750] for details). It requires anti-replay service to be enabled for an ESP SA using these encryption transforms. Unless the properties of sequence numbers for a multicast ESP SA include their uniqueness (see Section 2.6), encryption transforms that rely on Sequence Numbers for IV construction **MUST NOT** be used. In any case, such transforms **MUST NOT** be used for any G-IKEv2 SA (both unicast and multicast).

3. Group Key Management and Access Control

Through the G-IKEv2 rekey, G-IKEv2 supports algorithms such as Logical Key Hierarchy (LKH) that have the property of denying access to a new group key by a member removed from the group (forward access control) and to an old group key by a member added to the group (backward access control). This is unrelated to the Perfect Forward Secrecy (PFS) property as defined in Section 2.12 of [RFC7296].

Group management algorithms providing forward and backward access control other than LKH have also been proposed, for example, OFT [OFT] and Subset Difference [NNL]. These algorithms could be used with G-IKEv2 but are not specified as a part of this document.

This specification assumes that all group keys, that are sent to the GMs by the GCKS, are encrypted with some other keys, called Key Wrap Keys (KWKs). The Key Wrap Algorithm transform defines the algorithm used for key wrapping in the context of an SA.

3.1. Key Wrap Keys

Every GM always knows at least one KWK -- the KWK that is associated with the IKE SA or multicast Rekey SA over which wrapped keys are sent. In this document, it is called default KWK and is denoted as "GSK_w".

For the purpose of forward access control, the GCKS may provide each GM with its personal KWK at the time of registration. Additionally, several intermediate KWKs that form a key hierarchy and are shared among several GMs may be provided by the GCKS.

Each KWK is associated with a key wrap algorithm specified in the Key Wrap Algorithm transform. The size of these KWKs is determined by the key wrap algorithm used, but it **SHOULD NOT** be less than the size of the key for the Encryption Algorithm transform for the Rekey SA and for all Data-Security SAs in the group (taking the Key Length attribute into consideration if it is present).

3.1.1. Default Key Wrap Key

The default KWK (GSK_w) is only used in the context of a single IKE SA. Every IKE SA (unicast IKE SA or multicast Rekey SA) will have its own GSK_w.

For the unicast IKE SA (used for the GM registration and for GSA_INBAND_REKEY exchanges if they appear), the GSK_w is computed as follows:

```
GSK_w = prf+(SK_d, "Key Wrap for G-IKEv2")
```

where the string "Key Wrap for G-IKEv2" is 20 ASCII characters without null termination.

For the multicast Rekey SA, the GSK_w is provided along with other SA keys as defined in Section 3.4.

3.2. GCKS Key Management Semantics

The Wrapped Key Download method allows the GCKS to employ various key management methods.

A simple key management method: The GCKS always sends group SA keys encrypted with the GSK_w.

An LKH key management method: The GCKS provides each GM with an individual key at the time of the GM registration (encrypted with GSK_w). Then, the GCKS forms a hierarchy of keys so that the group SA keys are encrypted with other keys that are encrypted with other keys and so on, tracing back to the keys for each GM.

Other key policies may also be employed by the GCKS.

3.2.1. Forward Access Control Requirements

When a group membership is altered using a group management algorithm, new Data-Security SAs and their associated keys are usually also needed. New Data-Security SAs and keys ensure that members who were denied access can no longer participate in the group.

If forward access control is a desired property of the group, a new TEK policy and the associated keys **MUST NOT** be included in a G-IKEv2 rekey message, which changes group membership. This is required because the GSA TEK policy and the associated keys are not protected with the new KEK. A second G-IKEv2 rekey message can deliver the new GSA TEK policies and their associated keys because it will be protected with the new KEK and thus will not be visible to the members who were denied access.

If forward access control policy for the group includes keeping group policy changes from members that are denied access to the group, then two sequential G-IKEv2 rekey messages changing the group KEK **MUST** be sent by the GCKS. The first G-IKEv2 rekey message creates a new KEK for the group. GMs, which are denied access, will not be able to access the new KEK, but they will see the group policy since the G-IKEv2 rekey message is protected under the current KEK. A subsequent G-IKEv2 rekey message containing the changed group policy and again changing the KEK allows complete forward access control. A G-IKEv2 rekey message **MUST NOT** change the policy without creating a new KEK.

If other methods of using LKH or other group management algorithms are added to G-IKEv2, those methods **MAY** remove the above restrictions requiring multiple G-IKEv2 rekey messages, providing those methods specify how the forward access control policy is maintained within a single G-IKEv2 rekey message.

3.3. GM Key Management Semantics

This specification defines GM Key Management semantics in such a way that it doesn't depend on the key management method employed by the GCKS. This allows having all the complexity of key management in the GCKS, which is free to implement various key management methods such as direct transmitting of group SA keys or using some kind of key hierarchy (e.g., LKH). The GM behavior is the same for all of these policies.

All keys in G-IKEv2 are transmitted in encrypted form as specified in Section 4.5.4. This format includes a 32-bit Key ID (ID of a key that is encrypted) and a 32-bit KWK ID (ID of a key that was used to encrypt this key). Keys may be encrypted either with a default KWK (GSK_w) or with other keys, which the GM has received in the WRAP_KEY attributes. If a key was encrypted with GSK_w, then the KWK ID field is set to zero. Otherwise, the KWK ID field identifies the key used for encryption. A zero Key ID always identifies the key from which the keys for protecting Data-Security SAs and Rekey SA are taken.

When a GM receives a message from the GCKS installing the new Data-Security or Rekey SA, it will contain a KD payload with an SA_KEY attribute containing keying material for this SA. For a Data-Security SA, exactly one SA_KEY attribute will be present with both Key ID and KWK ID fields set to zero. This means that the default KWK (GSK_w) should be used to extract this keying material.

For a multicast Rekey SA, multiple SA_KEY attributes may be present depending on the key management method employed by the GCKS. If multiple SA_KEY attributes are present, then all of them **MUST** contain the same keying material encrypted using different KWKs. The GM in general is unaware of the key management method used by the GCKS and can always use the same procedure to get the keys. The GM tries to decrypt at least one of the SA_KEY attributes using either the GSK_w or the keys from the WRAP_KEY attributes that are present in the same message or were received in previous messages.

We will use the term "Key Path" to describe an ordered sequence of keys where each subsequent key was used to encrypt the previous one. The GM keeps its own Key Path (called Working Key Path) in the memory associated with each group it is registered to and updates it when needed. When the GSA_REKEY message is received, the GM processes the received SA_KEY attributes one by one and tries to construct a new key path that starts from one of these attributes and ends with any key in the Working Key Path or with the default KWK (GSK_w).

In the simplest case, the SA_KEY attribute is encrypted with GSK_w so that the new Key Path is empty. If more complex key management methods are used, then a Key Path will contain intermediate keys from the WRAP_KEY attributes received by a GM so far, starting from its registration to the group. If the GM is able to construct a new Key Path using intermediate keys it has, then it is able to decrypt the SA_KEY attribute and use its content to form new SA keys. If it is unable to build a new Key Path, then it means that the GM is excluded from the group.

Depending on the new Key Path, the GM should do the following actions to be prepared for future key updates:

- If the new Key Path is empty, then no actions are needed. This may happen if no WRAP_KEY attributes from the received message were used.
- If the new Key Path is non-empty and it ends with the default KWK (GSK_w), then the whole new Key Path is stored by the GM as the GM's Working Key Path. This situation may only happen at the time the GM is registering to the group, when the GCKS is providing the GM with its personal key and the other keys from the key tree that are needed. These keys form an initial Working Key Path for this GM.
- In all other cases, the new Key Path will end at some intermediate key from the GM's current Working Key Path. In this case, the new Key Path is constructed by replacing a part of the GM's current Working Key Path from the beginning and up to (but not including) the key that the GM has used to decrypt the last key in the new Key Path.

Appendix A contains an example of how this algorithm works in case of LKH key management method.

3.4. SA Keys

The keys that are used for Data-Security SAs or a Rekey SA (called SA keys here) are downloaded to GMs in the form of keying material from which, according to policy, a set of keys are deterministically extracted.

For a Data-Security SA, the keys are taken in accordance to the third bullet from Section 2.17 of [RFC7296]. In particular, for the ESP and AH SAs, the encryption key (if any) **MUST** be taken from the leftmost bits of the keying material and the integrity key (if any) **MUST** be taken from the remaining bits.

For a Rekey SA, the following keys are taken from the keying material:

```
GSK_e | GSK_a | GSK_w = KEYMAT
```

where GSK_e and GSK_a are the keys used for the Encryption Algorithm and the Integrity Algorithm transforms, respectively, for the corresponding SA and GSK_w is a default KWK for this SA. Note that GSK_w is used with the key wrap algorithm specified in the Key Wrap Algorithm transform. If an AEAD algorithm is used for encryption, then the GSK_a key will not be used (GM can use the formula above assuming the length of GSK_a is zero).

4. Header and Payload Formats

The G-IKEv2 is an IKEv2 extension and thus inherits its wire format for data structures. However, the processing of some payloads are different. Several new payloads are defined: Group Identification (IDg) (Section 4.2), Security Association - GM Supported Transforms (SAg) (Section 4.3), Group Security Association (GSA) (Section 4.4), and Key Download (KD) (Section 4.5).

The G-IKEv2 header (Section 4.1), IDg payload, and SAg payload reuse the IKEv2 format for the IKEv2 header, IDi/IDr payloads, and SA payload, respectively. New exchange types GSA_AUTH, GSA_REGISTRATION, GSA_REKEY, and GSA_INBAND_REKEY are also added.

This section describes new payloads and the differences in the processing of existing IKEv2 payloads.

4.1. G-IKEv2 Header

G-IKEv2 uses the same IKE header format as specified in Section 3.1 of [RFC7296]. The Major Version is 2 and the Minor Version is 0, as in IKEv2. IKE SA Initiator's SPI, IKE SA Responder's SPI, Flags, Message ID, and Length are as specified in [RFC7296].

4.2. Group Identification Payload

The Group Identification (IDg) payload allows the GM to indicate which group it wants to join. The payload is constructed by using the IKEv2 Identification Payload (Section 3.5 of [RFC7296]). ID type ID_KEY_ID **MUST** be supported. ID types ID_IPV4_ADDR, ID_FQDN, ID_RFC822_ADDR, and ID_IPV6_ADDR **SHOULD** be supported. ID types ID_DER_ASN1_DN and ID_DER_ASN1_GN are not expected to be used. The Payload Type for the IDg payload is fifty (50).

4.3. Security Association - GM Supported Transforms Payload

The Security Association - GM Supported Transforms (SAg) payload declares which Transforms a GM is willing to accept. The payload is constructed using the format of the IKEv2 Security Association payload (Section 3.3 of [RFC7296]). The Payload Type for SAg payloads is thirty-three (33), which is identical to the SA Payload Type.

4.4. Group Security Association Payload

The GSA payload is used by the GCKS to assert security attributes for both Rekey and Data-Security SAs. The Payload Type for the GSA payload is fifty-one (51).

```
                     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        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                        <Group Policies>                       ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 14: GSA Payload Format

The GSA payload fields are defined as follows:

Next Payload, C, RESERVED, and Payload Length fields:
Comprise the IKEv2 generic payload header and are defined in Section 3.2 of [RFC7296].

Group Policies (variable):
A set of group policies for the group.

4.4.1. Group Policies

Group policies are comprised of two types: group SA policy and group-wide (GW) policy. Group SA policy defines parameters for the Security Association of the group. Depending on the employed security protocol, a group SA policy may be either a Rekey SA (GSA KEK) policy or a Data-Security (GSA TEK) SA policy. GSA payload may contain zero or one GSA KEK policy, zero or more GSA TEK policies, and zero or one GW policy, where either one GSA KEK or one GSA TEK policy **MUST** be present.

This latitude allows various group policies to be accommodated. For example, if the group policy does not require the use of a Rekey SA, the GCKS would not need to send a GSA KEK policy to the group member since all SA updates would be performed using the GSA_INBAND_REKEY exchange via the unicast IKE SA. Alternatively, group policy might use a Rekey SA but choose to download a KEK to the GM only as part of the unicast IKE SA. Therefore, the GSA KEK policy would not be necessary as part of the GSA_REKEY message.

Specifying multiple GSA TEKs allows multiple related data streams (e.g., video, audio, and text) to be associated with a session, but each are protected with an individual Security Association.

A GW policy allows for the distribution of group-wide policy, such as instructions for when to activate and deactivate SAs.

Policies are distributed in substructures to the GSA payload. The format of the substructures is defined in Section 4.4.2 (for group SA policy) and in Section 4.4.3 (for GW policy). The first octet of the substructure unambiguously determines its type; it is zero for GW policy and non-zero (actually, it contains a Security Protocol Identifier) for group SA policies.

4.4.2. Group Security Association Policy Substructure

The group SA policy substructure contains parameters for a single SA that is used with this group. Depending on the security protocol, the SA is either a Rekey SA or a Data-Security SA (ESP and AH). The GCKS **MUST NOT** distribute both ESP and AH policies for the same set of Traffic Selectors.

```
                     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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Protocol   |   SPI Size    |             Length            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                              SPI                              ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                   Source Traffic Selector                     ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                 Destination Traffic Selector                  ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                     <Group SA Transforms>                     ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                     <Group SA Attributes>                     ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 15: Group SA Policy Substructure Format

The group SA policy fields are defined as follows:

Protocol (1 octet):
Identifies the security protocol for this group SA. The values are defined in the "IKEv2 Security Protocol Identifiers" registry in [IKEV2-IANA]. The valid values for this field are 6 (GIKE_UPDATE) for Rekey SA and 2 (AH) or 3 (ESP) for Data-Security SAs.

SPI Size (1 octet):
Size of the SPI for the SA. SPI size depends on the SA protocol. It is 16 octets for GIKE_UPDATE and 4 octets for AH and ESP.

Length (2 octets, unsigned integer):
Length of this substructure including the header.

SPI (variable):
Security Parameter Index for the group SA. The size of this field is determined by the SPI Size field. As described above, these SPIs are assigned by the GCKS. In the case of GIKE_UPDATE, the SPI is the IKEv2 header SPI pair where the first 8 octets become the "IKE SA Initiator's SPI" field in the G-IKEv2 rekey message IKEv2 HDR, and the second 8 octets become the "IKE SA Responder's SPI" in the same HDR.

Source & Destination Traffic Selectors (variable):

Substructures describing the source and destination of the network identities. The format for these substructures is defined in IKEv2 (Section 3.13.1 of [RFC7296]).

For the Rekey SA (with the GIKE_UPDATE protocol), the destination traffic selectors **MUST** define a single multicast IP address, an IP protocol (assumed to be UDP), and a single port the GSA_REKEY messages will be destined to. In this case, the source traffic selector **SHOULD** define a single IP address, an IP protocol (assumed to be UDP), and a single port the GSA_REKEY messages will be originated from. The source traffic selector **MAY** define a wildcard IP address and/or wildcard port. For the Data-Security (AH and ESP) SAs, the destination traffic selectors will usually define a single multicast IP address. The source traffic selector in this case will usually define a single IP address or be a wildcard selector. An IP protocol and ports define the characteristics of traffic protected by this Data-Security SA.

If the Data-Security SAs are created in tunnel mode, then it **MUST** be tunnel mode with address preservation (see Multicast Extensions to the Security Architecture [RFC5374]. UDP encapsulation of ESP packets [RFC3948] cannot be specified in G-IKEv2 and thus is not used for the multicast Data-Security SAs.

Group SA Transforms (variable):

A list of Transform Substructures specifies the policy information for the SA. The format is defined in IKEv2 (Section 3.3.2 of [RFC7296]). The "Last Substruc" field in each Transform Substructure is set to 3 except for the last Transform Substructure, where it is set to 0. Section 4.4.2.1 describes using IKEv2 transforms in the group SA policy substructure.

Group SA Attributes (variable):

Contains policy attributes associated with the group SA. The following sections describe the possible attributes. Any or all attributes may be optional, depending on the protocol and the group policy. Section 4.4.2.2 defines attributes used in the group SA policy substructure.

4.4.2.1. Group SA Transforms

Group SA policy is defined by the means of transforms in the group SA policy substructure. For this purpose, the transforms defined in [RFC7296] are used. In addition, new Transform Types are defined for use in G-IKEv2: Group Controller Authentication Method (GCAUTH) and Key Wrap Algorithm (KWA); see Section 9.

Valid Transform Types depend on the SA protocol and are summarized in the table below. Exactly one instance of each mandatory Transform Type and at most one instance of each optional Transform Type **MUST** be present in the group SA policy substructure.

Protocol	Mandatory Types	Optional Types
GIKE_UPDATE	ENCR, INTEG*, GCAUTH**, KWA	
ESP	ENCR, SN	INTEG

Protocol	Mandatory Types	Optional Types
AH	INTEG, SN	

Table 2: Valid Transform Types

Notes:

(*): If the AEAD encryption algorithm is used, then INTEG transform either **MUST NOT** be specified or **MUST** contain value NONE; otherwise, it **MUST** be specified and **MUST** contain a value other than NONE.

(**): May only appear at the time of a GM registration (in the GSA_AUTH and GSA_REGISTRATION exchanges).

4.4.2.1.1. Group Controller Authentication Method Transform

The Group Controller Authentication Method (GCAUTH) transform is used to convey information on how the GCKS will authenticate the GSA_REKEY messages.

This document creates a new IKEv2 IANA registry for Transform IDs of this Transform Type, which has been initially populated as described in Section 9. In particular, the following entries have been added:

Value	Group Controller Authentication Method
0	Reserved
1	Implicit
2	Digital Signature

Table 3: Group Controller Authentication Method Transform IDs

These Transform IDs are defined as follows:

Implicit:
No authentication of the GSA_REKEY messages will be provided by the GCKS besides the ability for the GMs to correctly decrypt them and verify their ICV. In this case, the GCKS **MUST NOT** include the AUTH_KEY attribute into the KD payload. Additionally, the AUTH payload **MUST NOT** be included in the GIKE_UPDATE messages.

Digital Signature
Digital signatures will be used by the GCKS to authenticate the GSA_REKEY messages. In this case, the GCKS **MUST** include the AUTH_KEY attribute containing the public key into the KD payload at the time the GM is registered to the group. To specify the details of the signature algorithm, a new attribute Signature Algorithm Identifier (value 18) is defined. This attribute

contains DER-encoded ASN.1 object AlgorithmIdentifier, which specifies the signature algorithm and the hash function that the GCKS will use for authentication. The AlgorithmIdentifier object is defined in Section 4.1.1.2 of [RFC5280]. Also, see [RFC7427] for the list of common AlgorithmIdentifier values used in IKEv2.

In the case of the Digital Signature Transform ID, the GCKS **MUST** include the Signature Algorithm Identifier attribute in the Group Controller Authentication Method transform. In this case, the AUTH payload in the GIKE_UPDATE messages **MUST** contain the Digital Signature authentication method (value 14) and be formatted as defined in Section 3 of [RFC7427]. The AlgorithmIdentifier ASN.1 object in the AUTH payload **MUST** match the content of the Signature Algorithm Identifier attribute in the Group Controller Authentication Method transform. The Signature Algorithm Identifier attribute is only meaningful for the Digital Signature Transform ID and **MUST NOT** be used with other Transform IDs.

More authentication methods may be defined in the future.

The authentication method **MUST NOT** change as a result of rekey operations. This means that the Group Controller Authentication Method transform **MUST NOT** appear in the rekey messages; it may only appear in the registration exchange (either GSA_AUTH or GSA_REGISTRATION).

The type of the Group Controller Authentication Method transform is 14.

4.4.2.1.2. Key Wrap Algorithm Transform

The Key Wrap Algorithm (KWA) transform is used to convey information about an algorithm that is used for key wrapping in G-IKEv2. See Section 4.5.4 for details.

This document creates a new IKEv2 IANA registry for the key wrap algorithms, which has been initially populated as described in Section 9. In particular, the following entries have been added:

Value	Key Wrap Algorithm
0	Reserved
1	KW_5649_128
2	KW_5649_192
3	KW_5649_256
4	KW_ARX

Table 4: Key Wrap Algorithm Transform IDs

These algorithms are defined as follows:

KW_5649_128, KW_5649_192, KW_5649_256:
The key wrap algorithm defined in [RFC5649] with a 128-bit, 192-bit, and 256-bit key, respectively. This key wrap algorithm is designed for use with AES block cipher.

KW_ARX:
The ARX-KW-8-2-4-GX key wrap algorithm defined in [ARX-KW]. This key wrap algorithm is designed for use with Chacha20 stream cipher.

More key wrap algorithms may be defined in the future. The requirement is that these algorithms must be able to wrap key material of size up to 256 bytes.

The type of the Key Wrap Algorithm transform is 13.

4.4.2.1.3. Sequence Numbers Transform

The Sequence Numbers (SNs) Transform Type is defined in [RFC9827]. This transform describes the properties of sequence numbers of IPsec packets. There are currently two Transform IDs defined for this Transform Type: "32-bit Sequential Numbers" and "Partially Transmitted 64-bit Sequential Numbers" that correspond to non-ESN and ESN cases from AH [RFC4302] and ESP [RFC4303] specifications.

Transform ID "32-bit Sequential Numbers" **SHOULD** be used by the GCKS for single-sender multicast Data-Security SAs utilizing protocols ESP or AH.

Since both AH [RFC4302] and ESP [RFC4303] are defined in such a way that high-order 32 bits of extended sequence numbers are never transmitted, it makes using ESN in multicast Data-Security SAs problematic because GMs that join the group long after it is created will have to somehow learn the current high-order 32 bits of ESN for each sender in the group. The algorithm for doing this described in AH [RFC4302] and ESP [RFC4303] is resource-consuming and is only suitable when a receiver is able to guess the high-order 32 bits close enough to its real value, which is not the case for multicast SAs. For this reason, the "Partially Transmitted 64-bit Sequential Numbers" Transform ID **MUST NOT** be used for multicast Data-Security SAs utilizing protocols ESP or AH.

This document defines a new Transform ID for this Transform Type: "32-bit Unspecified Numbers" (2). This Transform ID defines the following properties:

- Sequence numbers are 32 bits in size and are transmitted in the Sequence Number field of AH and ESP packets.
- The value of sequence numbers is not guaranteed to be unique for the duration of an SA, thus they are not suitable for replay protection.

This Transform ID **MUST** be used by the GCKS in case of multi-sender multicast Data-Security SAs utilizing protocols ESP or AH to inform the GMs that the replay protection is not expected to be possible. The GCKS **MAY** also use this Transform ID for single-sender multicast Data-Security SAs if replay protection is not needed (e.g., it is done on the application level).

4.4.2.2. Group SA Attributes

Group SA attributes are generally used to provide GMs with additional parameters for the group SA policy. Unlike security parameters distributed via transforms, which are expected not to change over time (unless the policy changes), the parameters distributed via group SA attributes may depend on the time the provision takes place, on the existence of other group SAs, or on other conditions.

This document creates a new IKEv2 IANA registry for the types of group SA attributes, which has been initially populated as described in Section 9. In particular, the following attributes have been added:

Value	Group SA Attributes	Format	Multi-Valued	Used in Protocol
0	Reserved			
1	GSA_KEY_LIFETIME	TLV	NO	GIKE_UPDATE, AH, ESP
2	GSA_INITIAL_MESSAGE_ID	TLV	NO	GIKE_UPDATE
3	GSA_NEXT_SPI	TLV	YES	GIKE_UPDATE, AH, ESP

Table 5: Group SA Attributes

The attributes follow the format defined in IKEv2 (Section 3.3.5 of [RFC7296]). The "Format" column defines what attribute format is allowed: Type/Length/Value (TLV) or Type/Value (TV). The "Multi-Valued" column defines whether multiple instances of the attribute can appear. The "Used in Protocol" column lists the security protocols, for which the attribute can be used.

4.4.2.2.1. GSA_KEY_LIFETIME Attribute

The GSA_KEY_LIFETIME attribute (1) specifies the maximum time for which the SA is valid. The value is a 4-octet unsigned integer in network byte order, specifying a valid time period in seconds. When the lifetime expires, the group SA and all associated keys **MUST** be deleted. The GCKS may delete the SA at any time before the end of the validity period.

A single attribute of this type **MUST** be included into any group SA policy substructure if multicast rekey is employed by the GCKS. This attribute **SHOULD NOT** be used if inband rekey (via the GSA_INBAND_REKEY exchange) is employed by the GCKS for the GM.

4.4.2.2.2. GSA_INITIAL_MESSAGE_ID Attribute

The GSA_INITIAL_MESSAGE_ID attribute (2) defines the initial Message ID to be used by the GCKS in the GSA_REKEY messages. The Message ID is a 4-octet unsigned integer in network byte order.

A single attribute of this type is included into the GSA KEK policy substructure if the initial Message ID of the Rekey SA is non-zero. Note that it is always true if a GM joins the group after some multicast rekey operations have already taken place in this group. In this case, this attribute will be included into the group SA policy when the GM is registered.

This attribute **MUST NOT** be used if inband rekey (via the GSA_INBAND_REKEY exchange) is employed by the GCKS for the GM.

4.4.2.2.3. GSA_NEXT_SPI Attribute

The optional GSA_NEXT_SPI attribute (3) contains the SPI that the GCKS reserved for the next Rekey SA or Data-Security SAs replacing the current ones. The length of the attribute data is determined by the SPI Size field in the group SA policy substructure the attribute resides in (see Section 4.4.2), and the attribute data contains the SPI as it would appear on the network. Multiple attributes of this type **MAY** be included, meaning that any of the supplied SPIs can be used in the replacement group SA.

The GM **MAY** store these values. Later on, if the GM starts receiving messages with one of these SPIs without seeing a rekey message over the current Rekey SA, then it may be used as an indication that the rekey message got lost on its way to this GM. In this case, the GM **SHOULD** re-register to the group.

Note that this method of detecting lost rekey messages can only be used by group receivers. Additionally, there is no point to include this attribute in the GSA_INBAND_REKEY messages since they use reliable transport. Also note that the GCKS is free to forget its promises and not to use the SPIs it sent in the GSA_NEXT_SPI attributes before (e.g., in cases where the GCKS is rebooted), so the GM must only treat this information as a "best effort" made by the GCKS to prepare for future rekeys.

This attribute **MUST NOT** be used if inband rekey (via the GSA_INBAND_REKEY exchange) is employed by the GCKS for the GM.

4.4.3. Group-Wide Policy Substructure

Group-specific policy that does not belong to any SA policy can be distributed to all GMs using the GW policy substructure.

The GW policy substructure is defined as follows:

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Protocol   |   RESERVED    |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                      <GW Policy Attributes>                   ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 16: GW Policy Substructure Format

The GW policy substructure fields are defined as follows:

Protocol (1 octet):
MUST be zero. This value is reserved (see Section 9) and is never used for any security protocol, so it is used here to indicate that this substructure contains policy not related to any specific protocol.

RESERVED (1 octet):
MUST be zero on transmission and MUST be ignored on receipt.

Length (2 octets, unsigned integer):
Length of this substructure including the header.

GW Policy Attributes (variable):
Contains policy attributes associated with no specific SA. The following sections describe possible attributes. Any or all attributes may be optional depending on the group policy.

4.4.3.1. GW Policy Attributes

This document creates a new IKEv2 IANA registry for the types of group-wide policy attributes, which has been initially populated as described in Section 9. In particular, the following attributes have been added:

Value	GW Policy Attributes	Format	Multi-Valued
0	Reserved		
1	GWP_ATD	TV	NO
2	GWP_DTD	TV	NO
3	GWP_SENDER_ID_BITS	TV	NO

Table 6: GW Policy Attributes

The attributes follow the format defined in the IKEv2 (Section 3.3.5 of [RFC7296]). The "Format" column defines what attribute format is allowed: Type/Length/Value (TLV) or Type/Value (TV). The "Multi-Valued" column defines whether multiple instances of the attribute can appear.

4.4.3.1.1. GWP_ATD and GWP_DTD Attributes

Section 4.2.1 of [RFC5374] specifies a key rollover method that requires two values be provided to GMs: Activation Time Delay (ATD) and Deactivation Time Delay (DTD).

The GWP_ATD attribute (1) allows a GCKS to set the Activation Time Delay for Data-Security SAs of the group. The ATD defines how long active members of the group (those who sends traffic) should wait after receiving new SAs before sending traffic over them. Note that to achieve smooth rollover, passive members of the group should activate the SAs immediately once they receive them.

The GWP_DTD attribute (2) allows the GCKS to set the DTD for previously distributed SAs. The DTD defines how long after receiving a request to delete Data-Security SAs passive GMs should wait before actually deleting them. Note that active members of the group should stop sending traffic over these old SAs once new replacement SAs are activated (after time specified in the GWP_ATD attribute).

The GWP_ATD and GWP_DTD attributes contain a 16-bit unsigned integer in network byte order, specifying the delay in seconds. These attributes are **OPTIONAL**. If one of them or both are not sent by the GCKS, then no corresponding delay should be employed.

4.4.3.1.2. GWP_SENDER_ID_BITS Attribute

The GWP_SENDER_ID_BITS attribute (3) declares how many bits of the cipher nonce are taken to represent a Sender-ID value. The bits are applied as the most significant bits of the IV, as shown in Figure 1 of Using Counter Modes with ESP and AH to Protect Group Traffic [RFC6054] and as specified in Section 2.5.2. Guidance for a GCKS choosing the value is provided in Section 3 of [RFC6054]. This value is applied to each Sender-ID value distributed in the KD payload.

The GCKS **MUST** include this attribute if there are more than one senders in the group and any of the Data-Security SAs use counter-based cipher mode. The number of Sender-ID bits is represented as a 16-bit unsigned integer in network byte order.

4.5. Key Download Payload

The Key Download (KD) payload contains the group keys for the SAs specified in the GSA payload. The Payload Type for the Key Download payload is fifty-two (52).

```
                     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         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                           <Key Bags>                          ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 17: Key Download Payload Format

The Key Download payload fields are defined as follows:

Next Payload, C, RESERVED, and Payload Length fields:
: Comprise the IKEv2 generic payload header and are defined in Section 3.2 of [RFC7296].

Key Bags (variable):
: A set of key bag substructures.

4.5.1. Key Bags

Keys are distributed in substructures called key bags. Each key bag contains one or more keys that are logically related -- these are keys for either a single SA (Data-Security SA or Rekey SA) or a single GM (in the latter case, besides keys, the key bag may also contain security parameters for this GM).

For this reason, two types of key bags are defined: Group Key Bag and Member Key Bag. The type is unambiguously determined by the first byte of the key bag substructure; for a Member Key Bag, it is zero and for a Group Key Bag, it contains a Security Protocol Identifier, which is always non-zero. For a Group Key Bag, the Protocol along with the SPI (see Figure 18) identify the SA that is associated with the keys in this bag.

4.5.2. Group Key Bag Substructure

The Group Key Bag substructure contains SA key information. This key information is associated with some group SAs: either with Data-Security SAs or with a group Rekey SA.

```
                     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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Protocol   |   SPI Size    |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                              SPI                              ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                  <Group Key Bag Attributes>                   ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 18: Group Key Bag Substructure Format

Protocol (1 octet):
: Identifies the security protocol for this key bag. The values are defined in the "IKEv2 Security Protocol Identifiers" registry in [IKEV2-IANA]. The valid values for this field are: 6 (GIKE_UPDATE) for KEK Key packet and 2 (AH) or 3 (ESP) for TEK key bag.

SPI Size (1 octet):
: Size of the SPI for the corresponding SA. SPI size depends on the security protocol. It is 16 octets for GIKE_UPDATE and 4 octets for AH and ESP.

Length (2 octets, unsigned integer):
: Length of this substructure including the header.

SPI (variable):
Security Parameter Index for the corresponding SA. The size of this field is determined by the SPI Size field. In the case of GIKE_UPDATE, the SPI is the IKEv2 header SPI pair where the first 8 octets become the "IKE SA Initiator's SPI" field in the G-IKEv2 rekey message IKEv2 HDR, and the second 8 octets become the "IKE SA Responder's SPI" in the same HDR.

Group Key Bag Attributes (variable):
Contains key information for the corresponding SA.

This document creates a new IKEv2 IANA registry for the types of Group Key Bag attributes, which has been initially populated as described in Section 9. In particular, the following attributes have been added:

Value	Group Key Bag Attributes	Format	Multi-Valued	Used in Protocol
0	Reserved			
1	SA_KEY	TLV	YES* NO	GIKE_UPDATE AH, ESP

Table 7: Group Key Bag Attributes

Notes:

(*): Multiple SA_KEY attributes may only appear for the GIKE_UPDATE protocol in the GSA_REKEY pseudo-exchange if the GCKS uses the group key management method that allows excluding GMs from the group (like LKH).

The attributes follow the format defined in IKEv2 (Section 3.3.5 of [RFC7296]). The "Format" column defines what attribute format is allowed: Type/Length/Value (TLV) or Type/Value (TV). The "Multi-Valued" column defines whether multiple instances of the attribute can appear. The "Used in Protocol" column lists the security protocols, for which the attribute can be used.

4.5.2.1. SA_KEY Attribute

The SA_KEY attribute (1) contains a keying material for the corresponding SA. The content of the attribute is formatted according to Section 4.5.4 with a precondition that the Key ID field **MUST** always be zero. The size of the keying material **MUST** be equal to the total size of the keys needed to be taken from this keying material (see Section 3.4) for the corresponding SA.

If the key bag is for a Data-Security SA (AH or ESP protocols), then exactly one SA_KEY attribute **MUST** be present with both Key ID and KWK ID fields set to zero.

If the key bag is for a Rekey SA (GIKE_UPDATE protocol), then exactly one SA_KEY attribute **MUST** be present in the GSA_AUTH, GSA_REGISTRATION, and GSA_INBAND_REKEY exchanges. In the GSA_REKEY pseudo-exchange, at least one SA_KEY attribute **MUST** be present, and more attributes **MAY** be present (depending on the key management method employed by the GCKS).

4.5.3. Member Key Bag Substructure

The Member Key Bag substructure contains keys and other parameters that are specific for a member of the group and are not associated with any particular group SA.

```
                     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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Protocol   |   RESERVED    |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                  <Member Key Bag Attributes>                  ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 19: Member Key Bag Substructure Format

The Member Key Bag substructure fields are defined as follows:

Protocol (1 octet):
: **MUST** be zero. This value is reserved (see Section 9) and is never used for any security protocol, so it is used here to indicate that this key bag is not associated with any particular SA.

RESERVED (octet):
: **MUST** be zero on transmission and **MUST** be ignored on receipt.

Length (2 octets, unsigned integer):
: Length of this substructure including the header.

Member Key Bag Attributes (variable):
: Contains key information and other parameters exclusively for a particular member of the group.

The Member Key Bag substructure contains sensitive information for a single GM. For this reason, it **MUST NOT** be sent in GSA_REKEY messages and **MUST** only be sent via unicast SA at the time the GM registers to the group (in either GSA_AUTH or GSA_REGISTRATION exchanges).

This document creates a new IKEv2 IANA registry for the types of Member Key Bag attributes, which has been initially populated as described in Section 9. In particular, the following attributes have been added:

Value	Member Key Bag Attributes	Format	Multi-Valued
0	Reserved		
1	WRAP_KEY	TLV	YES

Value	Member Key Bag Attributes	Format	Multi-Valued
2	AUTH_KEY	TLV	NO
3	GM_SENDER_ID	TLV	YES

Table 8: Member Key Bag Attributes

The attributes follow the format defined in the IKEv2 (Section 3.3.5 of [RFC7296]). The "Format" column defines what attribute format is allowed: Type/Length/Value (TLV) or Type/Value (TV). The "Multi-Valued" column defines whether multiple instances of the attribute can appear.

4.5.3.1. WRAP_KEY Attribute

The WRAP_KEY attribute (1) contains a key that is used to encrypt other keys. One or more of these attributes are sent to GMs if the GCKS key management method relies on some key hierarchy (e.g., LKH). This attribute **MUST NOT** be used if inband rekey (via the GSA_INBAND_REKEY exchange) is employed by the GCKS for the GM.

The content of the attribute has a format defined in Section 4.5.4 with a precondition that the Key ID field **MUST NOT** be zero. The algorithm associated with the key is defined by the Key Wrap Algorithm transform for the SA the WRAP_KEY attributes was sent in. The size of the attribute data **MUST** be equal to the key size for this key wrap algorithm.

Multiple instances of the WRAP_KEY attributes **MAY** be present in the key bag.

4.5.3.2. AUTH_KEY Attribute

The AUTH_KEY attribute (2) contains the key that is used to authenticate the GSA_REKEY messages. The content of the attribute depends on the authentication method the GCKS specified in the Group Controller Authentication Method transform in the GSA payload.

- If digital signatures are used for the GSA_REKEY message authentication, then the content of the AUTH_KEY attribute is a public key used for digital signature authentication. The public key **MUST** be represented as DER-encoded ASN.1 object SubjectPublicKeyInfo, defined in Section 4.1.2.7 of [RFC5280]. The algorithm field inside the SubjectPublicKeyInfo object **MUST** match the content of the Signature Algorithm Identifier attribute in the Group Controller Authentication Method transform. When the id-RSASSA-PSS object identifier appears in the algorithm field of the SubjectPublicKeyInfo object, then the parameters field **MUST** include the RSASSA-PSS-params structure.
- In case of implicit authentication, the AUTH_KEY Attribute is not used and **MUST** be absent (see Section 2.4.1).

Multiple instances of the AUTH_KEY attributes **MUST NOT** be sent.

4.5.3.3. GM_SENDER_ID Attribute

The GM_SENDER_ID attribute (3) is used to download one or more Sender-ID values for the exclusive use of a GM. One or more of these attributes **MUST** be sent by the GCKS if the GM informed the GCKS that it would be a sender (by including the GROUP_SENDER notification to the request) and if at least one of the Data-Security SAs included in the GSA payload uses a counter-based mode of encryption.

If the GMs have requested multiple Sender-ID values in the GROUP_SENDER notification, then the GCKS **SHOULD** provide it with the requested number of Sender-IDs by sending multiple instances of the GM_SENDER_ID attribute. The GCKS **MAY** send fewer values than requested by the GM (e.g., if it is running out of Sender-IDs), but it **MUST NOT** send more than requested.

This attribute **MUST NOT** appear in the rekey operations (in the GSA_REKEY pseudo-exchange or in the GSA_INBAND_REKEY exchange).

4.5.4. Key Wrapping

Symmetric keys in G-IKEv2 are never sent in clear inside G-IKEv2 messages. They are always protected with other symmetric keys. This protection is called key wrapping. Algorithms used for key wrapping are usually based on generic encryption algorithms, but their mode of operation is optimized for protecting short high-entropy data with minimal additional overhead. While key wrap algorithms can be generic in general, they are often tied to the underlying encryption algorithms in practice. For example, AES Key Wrap with Padding Algorithm [RFC5649] defines key wrapping using AES, and Key Wrapping Constructions using SipHash and ChaCha [ARX-KW] define key wrapping using Chacha20.

In G-IKEv2, the key wrap algorithm **MUST** be negotiated in the IKE_SA_INIT exchange so that the GCKS is able to send encrypted keys to the GM in the GSA_AUTH exchange. In addition, if the GCKS plans to use the multicast Rekey SA for group rekey, then it **MUST** specify the key wrap algorithm in the group SA policy for the Rekey SA inside the GSA payload. Note that key wrap algorithms for these cases **MAY** be different. For the unicast SA, the key wrap algorithm is negotiated between the GM and the GCKS, while for the multicast Rekey SA, the key wrap algorithm is provided by the GCKS to the GMs as part of the group policy. If an SAg payload is included in the GSA_AUTH request, then it **MUST** indicate which key wrap algorithms are supported by the GM. In all these cases, the key wrap algorithm is specified in a Key Wrap Algorithm transform (see Section 4.4.2.1.2).

The format of the wrapped key is shown in Figure 20.

```
                     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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Key ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            KWK ID                             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                        Encrypted Key                          ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 20: Wrapped Key Format

The Wrapped Key fields are defined as follows:

Key ID (4 octets):
: ID of the encrypted key. The value zero means that the encrypted key contains SA keys (in the form of keying material; see Section 3.4). Otherwise, it contains some intermediate key.

KWK ID (4 octets):
: ID of the key that was used to encrypt the key with a specified Key ID. The value zero means that the default KWK was used to encrypt the key. Otherwise, some intermediate key was used.

Encrypted Key (variable):
: The encrypted key bits. These bits comprise either a single encrypted key or a result of encryption of a concatenation of keys (key material) for several algorithms. The format of this field is determined by the key wrap algorithm for the SA the wrapped key is sent over.

4.6. Delete Payload

Delete payload is used in G-IKEv2 when the GCKS wants to delete Data-Security and Rekey SAs. The interpretation of the Protocol field in the Delete payload is extended so that zero protocol indicates deletion of whole Group SA (i.e., all Data-Security SAs and the Rekey SA). See Section 2.4.3 for detail.

4.7. Notify Payload

G-IKEv2 uses the same Notify payload as specified in Section 3.10 of [RFC7296].

There are additional Notify message types introduced by G-IKEv2 to communicate error conditions and status (see Section 9).

4.7.1. INVALID_GROUP_ID Notification

INVALID_GROUP_ID (45) is a new error type notification that indicates that the IDg payload sent during the registration process denotes an invalid group. The Protocol ID and SPI Size fields in the Notify payload **MUST** be zero. There is no data associated with this notification and the content of the Notification Data field **MUST** be ignored on receipt.

4.7.2. AUTHORIZATION_FAILED Notification

AUTHORIZATION_FAILED (46) is a new error type notification that is sent in the response to a GSA_AUTH or GSA_REGISTRATION message when authorization failed. The Protocol ID and SPI Size fields in the Notify payload **MUST** be zero. There is no data associated with this notification and the content of the Notification Data field **MUST** be ignored on receipt.

4.7.3. REGISTRATION_FAILED Notification

REGISTRATION_FAILED (49) is a new error type notification that is sent by the GCKS when the GM registration request cannot be satisfied for reasons not related to this particular GM, e.g., if the capacity of the group is exceeded. The Protocol ID and SPI Size fields in the Notify payload **MUST** be zero. There is no data associated with this notification and the content of the Notification Data field **MUST** be ignored on receipt.

4.7.4. GROUP_SENDER Notification

GROUP_SENDER (16429) is a new status type notification that is sent in the GSA_AUTH or the GSA_REGISTRATION exchanges to indicate that the GM intends to be sender of data traffic. The data includes a count of how many Sender-ID values the GM desires. The count **MUST** be 4 octets long and contain the big-endian representation of the number of requested Sender-IDs. The Protocol ID and SPI Size fields in the Notify payload **MUST** be zero.

4.8. Authentication Payload

G-IKEv2 uses the same Authentication payload as specified in Section 3.8 of [RFC7296] to authenticate the rekey message. However, if it is used in the GSA_REKEY messages, the content of the payload is computed differently as described in Section 2.4.1.1.

5. Using G-IKEv2 Attributes

G-IKEv2 defines a number of attributes that are used to convey information from the GCKS to GMs. There are some restrictions on where and when these attributes can appear in G-IKEv2 messages, which are defined when the attributes are introduced. For convenience, these restrictions are summarized in Table 9 (for multicast rekey operations) and Table 10 (for inband rekey operations) below.

The following notations are used:

S A single attribute of this type **MUST** be present.

M Multiple attributes of this type **MAY** be present.

[] Attribute is **OPTIONAL**.

- Attribute **MUST NOT** be present.

Note that the restrictions are defined per a substructure for which corresponding attributes are defined and not per a whole G-IKEv2 message.

Attributes	GSA_AUTH GSA_REGISTRATION	GSA_REKEY	Notes
Group SA Attributes (Section 4.4.2.2)			
GSA_KEY_LIFETIME	S	S	
GSA_INITIAL_MESSAGE_ID	[S]	[S]	
GSA_NEXT_SPI	[M]	[M]	
GW Policy Attributes (Section 4.4.3.1)			
GWP_ATD	[S]	[S]	
GWP_DTD	[S]	[S]	
GWP_SENDER_ID_BITS	S	-	1
Key Bag Attributes (Section 4.5.1)			
SA_KEY	S	S[M]	2
WRAP_KEY	[M]	[M]	3
AUTH_KEY	S	[S]	4
GM_SENDER_ID	S[M]	-	1

Table 9: Attributes in G-IKEv2 Exchanges with Multicast Rekey Operations

Notes:

(1): The GWP_SENDER_ID_BITS attribute **MUST** be present if the GCKS policy includes at least one cipher in counter mode of operation and if the GM included the GROUP_SENDER notify into the registration request. Otherwise, it **MUST NOT** be present. At least one GM_SENDER_ID attribute **MUST** be present in the former case (and more **MAY** be present if the GM requested more Sender-IDs), and it **MUST NOT** be present in the latter case.

(2): For a Data-Security SA, exactly one SA_KEY attribute **MUST** be present. For a Rekey SA, exactly one SA_KEY attribute **MUST** be present in the GSA_AUTH and the GSA_REGISTRATION exchange. In the GSA_REKEY pseudo-exchange, at least one SA_KEY attribute **MUST** be present and more of these attributes **MAY** be present.

(3): The WRAP_KEY attribute **MUST** be present if the GCKS employs a key management method that relies on a key tree (like LKH).

(4): The AUTH_KEY attribute **MUST** be present in the GSA_AUTH and GSA_REGISTRATION exchanges if the GCKS employs an authentication method of rekey operations based on digital signatures and **MUST NOT** be present if implicit authentication is employed. The AUTH_KEY attribute **MUST** be present in the GSA_REKEY pseudo-exchange if the GCKS employs an authentication method based on digital signatures and wants to change the public key for the following multicast rekey operations.

Attributes	GSA_AUTH GSA_REGISTRATION	GSA_INBAND_REKEY	Notes
Group SA Attributes (Section 4.4.2.2)			
GSA_KEY_LIFETIME	[S]	[S]	
GSA_INITIAL_MESSAGE_ID	-	-	
GSA_NEXT_SPI	-	-	
GW Policy Attributes (Section 4.4.3.1)			
GWP_ATD	[S]	[S]	
GWP_DTD	[S]	[S]	
GWP_SENDER_ID_BITS	S	-	1
Key Bag Attributes (Section 4.5.1)			
SA_KEY	S	S	
WRAP_KEY	-	-	
AUTH_KEY	-	-	
GM_SENDER_ID	S[M]	-	1

Table 10: Attributes in G-IKEv2 Exchanges with Inband Rekey Operations

Notes:

(1):

The GWP_SENDER_ID_BITS attribute **MUST** be present if the GCKS policy includes at least one cipher in counter mode of operation and the GM included the GROUP_SENDER notify into the registration request. Otherwise, it **MUST NOT** be present. At least one GM_SENDER_ID attribute **MUST** be present in the former case (and more **MAY** be present if the GM requested more Sender-IDs), and it **MUST NOT** be present in the latter case.

6. Interaction with IKEv2 and ESP Extensions

A number of IKEv2 and ESP extensions are defined that can be used to extend protocol functionality. G-IKEv2 is compatible with most of them. In particular, EAP authentication defined in [RFC7296] can be used to establish registration IKE SA, as well as EAP-only authentication [RFC5998] and secure password authentication [RFC6467]. G-IKEv2 is compatible with and can use IKEv2 Redirect Mechanism [RFC5685] and IKEv2 Session Resumption [RFC5723]. G-IKEv2 is also compatible with Multiple Key Exchanges in the IKEv2 framework, as defined in [RFC9370].

The above list of compatible IKEv2 extensions is not exhaustive. However, some IKEv2 extensions require special handling if used in G-IKEv2.

6.1. Implicit IV for Counter-Based Ciphers in ESP

Using implicit IV for counter-based encryption modes in ESP is defined in [RFC8750]. This extension relies on the uniqueness of ESP sequence numbers. Thus, it cannot be used for multi-sender multicast SAs. However, it is possible to use implicit IV extension for a single-sender multicast ESP SA. Note that while implicit IVs can be used with ESN, using ESN is prohibited in multicast SAs (see Section 4.4.2.1.3).

6.2. Mixing Preshared Keys in IKEv2 for Post-Quantum Security

G-IKEv2 can take advantage of the protection provided by Post-quantum Preshared Keys (PPKs) for IKEv2 [RFC8784]. However, the use of PPKs leaves the initial IKE SA susceptible to quantum computer (QC) attacks. Group SA keys are protected with the default KWK (GSK_w), which is derived from SK_d and thus cannot be broken even by an attacker equipped with a QC. However, other data sent over the initial IKE SA may be susceptible to an attacker equipped with a QC of a sufficient size. Such an attacker can store all the traffic until it obtains such a QC and then decrypt it (i.e., Store Now Decrypt Later attack). See Section 6 of [RFC8784] for details.

While the group keys are protected with PPK and thus are immune to QC, GCKS implementations that care about other data sent over initial IKE SA **MUST** rely on IKEv2 extensions that protect even initial IKE SA against QC (like [RFC9867]).

6.3. Aggregation and Fragmentation Mode for ESP

Aggregation and fragmentation mode for ESP is defined in [RFC9347]. This mode allows IP packets to be split over several ESP packets or several IP packets to be aggregated in a single ESP packet. This mode can only be used with ESP tunnel mode and relies on monotonically increasing sequence numbers in the incoming packets. Thus, it is impossible to use this mode for multi-sender multicast SAs. Since multicast Data-Security SAs are unidirectional, the congestion control feature of aggregation and fragmentation mode cannot be used.

It is possible to use the aggregation and fragmentation mode without congestion control for a single-sender multicast ESP SA created in tunnel mode. GMs supporting this mode can send the USE_AGGFRAG notification in the registration request along with the SAg payload. If the Data-Security SA(s) to be installed on GMs uses the aggregation and fragmentation mode, the GCKS would indicate it by including the USE_AGGFRAG notification along with the GSA payload in its response.

7. GDOI Protocol Extensions

Few extensions were defined for the GDOI protocol [RFC6407], like GDOI Support for IEC 62351 Security Services [RFC8052] or the GDOI GROUPKEY-PUSH Acknowledgement Message [RFC8263]. It is expected that these extensions will be redefined for G-IKEv2 in separate documents, if needed.

8. Security Considerations

When an entity joins the group and becomes a GM, it has to trust that the GCKS only authorized entities that are admitted to the group and has to trust that other GMs will not leak the information shared within the group.

8.1. GSA Registration and Secure Channel

G-IKEv2 registration procedure uses IKEv2 initial exchanges, inheriting all the security considerations documented in Section 5 of [RFC7296], including authentication, confidentiality, on-path attack protection, protection against replay/reflection attacks, and denial- of-service protection. The GSA_REGISTRATION exchange also takes advantage of those protections. In addition, G-IKEv2 brings in the capability to authorize a particular GM regardless of whether they have the IKEv2 credentials.

8.2. GSA Maintenance Channel

The GSA maintenance channel is cryptographically and integrity protected using the cryptographic algorithm and key negotiated in the GSA member registration exchange.

8.2.1. Authentication/Authorization

The authentication key is distributed during the GM registration and the receiver of the rekey message uses that key to verify the message came from the authorized GCKS. An implicit authentication can also be used, in which case, the ability of the GM to decrypt and to verify ICV of incoming messages is used as a proof that the sender knows group keys and therefore is a member of the group. However, implicit authentication doesn't provide source origin authentication, so the GM cannot be sure that the message came from the GCKS. For this reason, using implicit authentication is **NOT RECOMMENDED** unless used with a small group of trusted parties.

8.2.2. Confidentiality

Confidentiality is provided by distributing a confidentiality key as part of the GSA member registration exchange.

8.2.3. On-Path Attack Protection

The GSA maintenance channel is integrity protected by using a digital signature.

8.2.4. Replay/Reflection Attack Protection

The GSA_REKEY message includes a monotonically increasing sequence number to protect against replay and reflection attacks. A GM will recognize a replayed message by comparing the Message ID number to that of the last received rekey message. Any rekey message containing a Message ID number less than or equal to the last received value **MUST** be discarded. Implementations should keep a record of recently received GSA rekey messages for this comparison.

The strict role separation between the GCKS and the GMs and, as a consequence, the limitation for a Rekey SA to be outbound/inbound only, helps to prevent reflection attack.

9. IANA Considerations

9.1. New Registries

Per this document, new registries have been created for G-IKEv2 under the "Internet Key Exchange Version 2 (IKEv2) Parameters" registry group [IKEV2-IANA]. The terms Reserved, Expert Review, and Private Use are as defined in [RFC8126].

1. IANA has created the "Transform Type 13 - Key Wrap Algorithm Transform IDs" registry. The registration policy for this registry is Expert Review [RFC8126]. The initial values of the registry are as follows:

Value	Key Wrap Algorithm
0	Reserved
1	KW_5649_128
2	KW_5649_192
3	KW_5649_256
4	KW_ARX

Value	Key Wrap Algorithm
5-1023	Unassigned
1024-65535	Reserved for Private Use

Table 11

2. IANA has created the "Transform Type 14 - Group Controller Authentication Method Transform IDs" registry. The registration policy for this registry is Expert Review [RFC8126]. The initial values of the registry are as follows:

Value	Group Controller Authentication Method
0	Reserved
1	Implicit
2	Digital Signature
3-1023	Unassigned
1024-65535	Reserved for Private Use

Table 12

3. IANA has created the "Group SA Attributes" registry. The registration policy for this registry is Expert Review [RFC8126]. The initial values of the registry are as follows:

Value	Group SA Attributes	Format	Multi-Valued	Used in Protocol
0	Reserved			
1	GSA_KEY_LIFETIME	TLV	NO	GIKE_UPDATE, AH, ESP
2	GSA_INITIAL_MESSAGE_ID	TLV	NO	GIKE_UPDATE
3	GSA_NEXT_SPI	TLV	YES	GIKE_UPDATE, AH, ESP
4-16383	Unassigned			
16384-32767	Reserved for Private Use			

Table 13

4. IANA has created the "Group-Wide Policy Attributes" registry. The registration policy for this registry is Expert Review [RFC8126]. The initial values of the registry are as follows:

Value	GW Policy Attributes	Format	Multi-Valued
0	Reserved		
1	GWP_ATD	TV	NO
2	GWP_DTD	TV	NO
3	GWP_SENDER_ID_BITS	TV	NO
4-16383	Unassigned		
16384-32767	Reserved for Private Use		

Table 14

5. IANA has created the "Group Key Bag Attributes" registry. The registration policy for this registry is Expert Review [RFC8126]. The initial values of the registry are as follows:

Value	Group Key Bag Attributes	Format	Multi-Valued	Used in Protocol
0	Reserved			
1	SA_KEY	TLV	YES NO	GIKE_UPDATE AH, ESP
2-16383	Unassigned			
16384-32767	Reserved for Private Use			

Table 15

6. IANA has created the "Member Key Bag Attributes" registry. The registration policy for this registry is Expert Review [RFC8126]. The initial values of the registry are as follows:

Value	Member Key Bag Attributes	Format	Multi-Valued
0	Reserved		
1	WRAP_KEY	TLV	YES
2	AUTH_KEY	TLV	NO
3	GM_SENDER_ID	TLV	YES
4-16383	Unassigned		
16384-32767	Reserved for Private Use		

Table 16

9.1.1. Guidance for Designated Experts

In all cases of Expert Review described in this section, the designated expert (DE) is expected to ascertain the existence of suitable documentation (a specification) as described in [RFC8126] and verify that the document is permanently and publicly available. The DE is also expected to check the clarity of purpose and use of the requested code points. Lastly, the DE must verify that any specification produced outside the IETF does not conflict with work that is active or already published within the IETF.

9.2. Changes in the Existing IKEv2 Registries

1. In the "IKEv2 Exchange Types" registry, IANA has updated the references for the following entries to point to this document and has registered "GSA_INBAND_REKEY":

Value	Exchange Type
39	GSA_AUTH
40	GSA_REGISTRATION
41	GSA_REKEY
42	GSA_INBAND_REKEY

Table 17

2. In the "IKEv2 Payload Types" registry, IANA has listed this document as a reference for the following entries:

Value	Next Payload Type	Notation
50	Group Identification	IDg
51	Group Security Association	GSA
52	Key Download	KD

Table 18

3. In the "IKEv2 Payload Types" registry, IANA has updated the definition of Payload Type 33 and added a reference to this document as follows:

Value	Next Payload Type	Notation	Reference
33	Security Association	SA	[RFC7296]
	Security Association - GM Supported Transforms	SAg	RFC 9838

Table 19

4. In the "Transform Type Values" registry, IANA has made the following changes:
    - Registered "Key Wrap Algorithm (KWA)" and "Group Controller Authentication Method (GCAUTH)".
    - Updated the "Used In" column for values 1 and 3 and listed this document as an additional reference.

Type	Description	Used In
1	Encryption Algorithm (ENCR)	(IKE, GIKE_UPDATE, ESP)
3	Integrity Algorithm (INTEG)	(IKE, GIKE_UPDATE, AH, optional in ESP)
13	Key Wrap Algorithm (KWA)	(IKE, GIKE_UPDATE)
14	Group Controller Authentication Method (GCAUTH)	(GIKE_UPDATE)

Table 20

5. In the "IKEv2 Transform Attribute Types" registry, IANA has added the following entry:

Value	Attribute Type	Format
18	Signature Algorithm Identifier	TLV

Table 21

6. In the "Transform Type 5 - Sequence Numbers Transform IDs" registry, IANA has added the following entry:

Number	Name
2	32-bit Unspecified Numbers

Table 22

7. In the "IKEv2 Notify Message Error Types" registry, IANA has made the following changes:
    - Registered "REGISTRATION_FAILED".
    - Updated the references for "INVALID_GROUP_ID" and "AUTHORIZATION_FAILED" to point to this document.

Value	Notify Message Error Type
45	INVALID_GROUP_ID
46	AUTHORIZATION_FAILED

Value	Notify Message Error Type
49	REGISTRATION_FAILED

Table 23

8. An earlier draft of this document [G-IKEV2] registered the Notify type 16429 in the "IKEv2 Notify Message Status Types" registry with the name SENDER_REQUEST_ID. Per this document, IANA has renamed it as follows:

Value	Notify Message Status Type
16429	GROUP_SENDER

Table 24

9. In the "IKEv2 Security Protocol Identifiers" registry, IANA has added the following entry:

Protocol ID	Protocol
6	GIKE_UPDATE

Table 25

10. References

10.1. 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>.

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

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

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

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008, <https://www.rfc-editor.org/info/rfc5280>.

[RFC6054] McGrew, D. and B. Weis, "Using Counter Modes with Encapsulating Security Payload (ESP) and Authentication Header (AH) to Protect Group Traffic", RFC 6054, DOI 10.17487/RFC6054, November 2010, <https://www.rfc-editor.org/info/rfc6054>.

[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>.

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

[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>.

[RFC9827] Smyslov, V., "Renaming the Extended Sequence Numbers (ESN) Transform Type in the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 9827, DOI 10.17487/RFC9827, September 2025, <https://www.rfc-editor.org/info/rfc9827>.

10.2. Informative References

[ARX-KW] Shinichi, S., "ARX-KW, a family of key wrapping constructions using SipHash and ChaCha", Cryptology ePrint Archive, Paper 2020/059, January 2020, <https://eprint.iacr.org/2020/059.pdf>.

[G-IKEV2] Rowles, S., Yeung, A., Tran, P., and Y. Nir, "Group Key Management using IKEv2", Work in Progress, Internet-Draft, draft-yeung-g-ikev2-07, 5 November 2013, <https://datatracker.ietf.org/doc/html/draft-yeung-g-ikev2-07>.

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

[NNL] Naor, D., Naor, M., and J. Lotspiech, "Revocation and Tracing Schemes for Stateless Receivers", Advances in Cryptology - CRYPTO 2001, Lecture Notes in Computer Science, vol. 2139, pp. 41-62, DOI 10.1007/3-540-44647-8_3, 2001, <http://www.wisdom.weizmann.ac.il/~naor/PAPERS/2nl.pdf>.

[OFT] McGrew, D. and A. Sherman, "Key Establishment in Large Dynamic Groups Using One-Way Function Trees", IEEE Transactions on Software Engineering, vol. 29, no. 5, pp. 444-458, DOI 10.1109/TSE.2003.1199073, May 1998, <https://pdfs.semanticscholar.org/d24c/7b41f7bcc2b6690e1b4d80eaf8c3e1cc5ee5.pdf>.

[RFC2409] Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, DOI 10.17487/RFC2409, November 1998, <https://www.rfc-editor.org/info/rfc2409>.

[RFC2627] Wallner, D., Harder, E., and R. Agee, "Key Management for Multicast: Issues and Architectures", RFC 2627, DOI 10.17487/RFC2627, June 1999, <https://www.rfc-editor.org/info/rfc2627>.

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

[RFC3740] Hardjono, T. and B. Weis, "The Multicast Group Security Architecture", RFC 3740, DOI 10.17487/RFC3740, March 2004, <https://www.rfc-editor.org/info/rfc3740>.

[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, <https://www.rfc-editor.org/info/rfc3948>.

[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>.

[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, <https://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, <https://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, <https://www.rfc-editor.org/info/rfc4543>.

[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, <https://www.rfc-editor.org/info/rfc5374>.

[RFC5649] Housley, R. and M. Dworkin, "Advanced Encryption Standard (AES) Key Wrap with Padding Algorithm", RFC 5649, DOI 10.17487/RFC5649, September 2009, <https://www.rfc-editor.org/info/rfc5649>.

[RFC5685] Devarapalli, V. and K. Weniger, "Redirect Mechanism for the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5685, DOI 10.17487/RFC5685, November 2009, <https://www.rfc-editor.org/info/rfc5685>.

[RFC5723] Sheffer, Y. and H. Tschofenig, "Internet Key Exchange Protocol Version 2 (IKEv2) Session Resumption", RFC 5723, DOI 10.17487/RFC5723, January 2010, <https://www.rfc-editor.org/info/rfc5723>.

[RFC5998] Eronen, P., Tschofenig, H., and Y. Sheffer, "An Extension for EAP-Only Authentication in IKEv2", RFC 5998, DOI 10.17487/RFC5998, September 2010, <https://www.rfc-editor.org/info/rfc5998>.

[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>.

[RFC6467] Kivinen, T., "Secure Password Framework for Internet Key Exchange Version 2 (IKEv2)", RFC 6467, DOI 10.17487/RFC6467, December 2011, <https://www.rfc-editor.org/info/rfc6467>.

[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>.

[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, <https://www.rfc-editor.org/info/rfc7634>.

[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>.

[RFC8263] Weis, B., Mangla, U., Karl, T., and N. Maheshwari, "Group Domain of Interpretation (GDOI) GROUPKEY-PUSH Acknowledgement Message", RFC 8263, DOI 10.17487/RFC8263, November 2017, <https://www.rfc-editor.org/info/rfc8263>.

[RFC8750] Migault, D., Guggemos, T., and Y. Nir, "Implicit Initialization Vector (IV) for Counter-Based Ciphers in Encapsulating Security Payload (ESP)", RFC 8750, DOI 10.17487/RFC8750, March 2020, <https://www.rfc-editor.org/info/rfc8750>.

[RFC8784] Fluhrer, S., Kampanakis, P., McGrew, D., and V. Smyslov, "Mixing Preshared Keys in the Internet Key Exchange Protocol Version 2 (IKEv2) for Post-quantum Security", RFC 8784, DOI 10.17487/RFC8784, June 2020, <https://www.rfc-editor.org/info/rfc8784>.

[RFC9242] Smyslov, V., "Intermediate Exchange in the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 9242, DOI 10.17487/RFC9242, May 2022, <https://www.rfc-editor.org/info/rfc9242>.

[RFC9329] Pauly, T. and V. Smyslov, "TCP Encapsulation of Internet Key Exchange Protocol (IKE) and IPsec Packets", RFC 9329, DOI 10.17487/RFC9329, November 2022, <https://www.rfc-editor.org/info/rfc9329>.

[RFC9347] Hopps, C., "Aggregation and Fragmentation Mode for Encapsulating Security Payload (ESP) and Its Use for IP Traffic Flow Security (IP-TFS)", RFC 9347, DOI 10.17487/RFC9347, January 2023, <https://www.rfc-editor.org/info/rfc9347>.

[RFC9370] Tjhai, CJ., Tomlinson, M., Bartlett, G., Fluhrer, S., Van Geest, D., Garcia-Morchon, O., and V. Smyslov, "Multiple Key Exchanges in the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 9370, DOI 10.17487/RFC9370, May 2023, <https://www.rfc-editor.org/info/rfc9370>.

[RFC9867] Smyslov, V., "Mixing Preshared Keys in the IKE_INTERMEDIATE and CREATE_CHILD_SA Exchanges of the Internet Key Exchange Protocol Version 2 (IKEv2) for Post-Quantum Security", RFC 9867, DOI 10.17487/RFC9867, October 2025, <https://www.rfc-editor.org/info/rfc9867>.

Appendix A. Use of LKH in G-IKEv2

Section 5.4 of [RFC2627] describes the LKH architecture and how a GCKS uses LKH to exclude GMs. This section clarifies how the LKH architecture is used with G-IKEv2.

A.1. Notation

In this section, we will use the notation X{Y}, where a key with ID Y is encrypted with the key with ID X. The notation GSK_w{Y} means that the default wrap key GSK_w (with zero KWK ID)is used to encrypt key Y, and the notation X{K_sa} means key X is used to encrypt the SA key K_sa (which always has a Key ID of zero). Note that GSK_w{K_sa} means that the SA key is encrypted with the default wrap key, in which case, both KWK ID and Key ID are zero.

The content of the KD payload will be shown as a sequence of key bags. The Group Key Bag substructure will be denoted as GP(SAn)() when n is an SPI for the SA and the Member Key Bag substructure will be denoted as MP(). The content of the key bags is shown as SA_KEY and WRAP_KEY attributes with the notation described above. For simplicity, the type of the attribute will not be shown because it is implicitly defined by the type of key bag.

Below is the example of a KD payload:

```
KD(GP(SA1)(X{K_sa}),MP(Y{X},Z{Y},GSK_w{Z})
```

Figure 21: Example of a KD Payload

For simplicity, any other attributes in the KD payload are omitted.

We will also use the notation X->Y->Z to describe the Key Path. In this case, key Y is needed to decrypt key X and key Z is needed to decrypt key Y. In the example above, the keys had the following relation: K_sa->X->Y->Z->GSK_w.

A.2. Group Creation

When a GCKS forms a group, it creates a key tree as shown in Figure 22. The key tree contains logical keys (which are represented as the values of their Key IDs in the figure) and a private key shared with only a single GM (the GMs are represented as letters followed by the corresponding key ID in parentheses in the figure). The root of the tree contains the multicast Rekey SA key (which is represented as SAn(K_san). The figure below assumes that the Key IDs are assigned sequentially; this is not a requirement and only used for illustrative purposes. The GCKS may create a complete tree as shown or a partial tree, which is created on demand as members join the group.

```
                          SA1(K_sa1)
             +------------------------------+
             1                              2
     +---------------+              +---------------+
     3               4              5               6
 +-------+       +-------+      +--------+      +--------+
A(7)    B(8)    C(9)   D(10)  E(11)    F(12)  G(13)    H(14)
```

Figure 22: Initial LKH Tree

When GM A joins the group, the GCKS provides it with the keys in the KD payload of the GSA_AUTH or GSA_REGISTRATION exchange. Given the tree shown in figure above, the KD payload will be:

```
KD(GP(SA1)(1{K_sa1}),MP(3{1},7{3},GSK_w{7})
```

Figure 23: KD Payload for the Group Member A

From these attributes, the GM A will construct the Key Path K_sa1->1->3->7->GSK_w. Since it ends up with GSK_w, it will use all the WRAP_KEY attributes present in the path as its Working Key Path: 1->3->7.

Similarly, when other GMs join the group, they will be provided with the corresponding keys and thus the GMs will have the following Working Key Paths:

```
A: 1->3->7      B: 1->3->8      C: 1->4->9,     D: 1->4->10
E: 2->5->11     F: 2->5->12     G: 2->6->13     H: 2->6->14
```

Figure 24: Key Paths for all GMs

A.3. Simple Group SA Rekey

If the GCKS performs a simple SA rekey without changing group membership, it will only send a Group Key Bag in the KD payload with a new SA key encrypted with the default KWK.

```
KD(GP(SA2)(GSK_w{K_sa2}))
```

Figure 25: KD Payload for the Simple Group SA Rekey

All the GMs will be able to decrypt it and no changes in their Working Key Paths will happen.

A.4. Group Member Exclusion

If the GCKS has reason to believe that a GM should be excluded, then it can do so by sending a GSA_REKEY message that includes a set of GM_KEY attributes, which would allow all GMs, except for the excluded one, to get a new SA key.

In the example below, the GCKS excludes GM F. For this purpose, it changes the key tree as follows, replacing key 2 with key 15 and key 5 with key 16. It also generates a new SA key for a new SA3.

```
                          SA3(K_sa3)
             +------------------------------+
             1                             15
     +---------------+              +---------------+
     3               4             16               6
 +-------+       +-------+      +----           +--------+
A(7)    B(8)    C(9)   D(10)  E(11)    F(12)  G(13)    H(14)
```

Figure 26: LKH Tree after F Has Been Excluded

Then it sends the following KD payload for the new Rekey SA3:

```
KD(GP(SA3)(1{K_sa3},15{K_sa3}),MP(6{15},16{15},11{16})
```

Figure 27: KD Payload for the Group Member F

While processing this KD payload:

- GMs A, B, C, and D will be able to decrypt the SA_KEY attribute 1{K_sa3} by using the "1" key from their key path. Since no new GM_KEY attributes are in the new Key Path, they won't update their Working Key Paths.
- GMs G and H will construct new Key Path 15->6 and will be able to decrypt the intermediate key 15 using key 6 from their Working Key Paths. So, they will update their Working Key Paths replacing their beginnings up to key 6 with the new Key Path (thus replacing the key 2 with the key 15).
- GM E will construct a new Key Path 16->15->11 and will be able to decrypt the intermediate key 16 using key 11 from its Working Key Path. So, it will update its Working Key Path replacing its beginnings up to key 11 with the new Key Path (thus replacing key 2 with key 15 and key 5 with key 16).
- GM F won't be able to construct any Key Path leading to any key it possesses, so it will be unable to decrypt the new SA key for the SA3. Thus, it will be excluded from the group once the SA3 is used.

Finally, the GMs will have the following Working Key Paths:

```
A: 1->3->7      B: 1->3->8      C: 1->4->9,     D: 1->4->10
E: 15->16->11   F: excluded     G: 15->6->13    H: 15->6->14
```

Figure 28: Key Paths for all GMs after Exclusion of a GM

Acknowledgements

The authors thank Lakshminath Dondeti and Jing Xiang for first exploring the use of IKEv2 for group key management and providing the basis behind the protocol. Mike Sullenberger and Amjad Inamdar were instrumental in helping resolve many issues in several draft versions of the document.

The authors are grateful to Tero Kivinen, Daniel Migault, Gorry Fairhurst, Robert Sparks, Russ Housley, and Paul Wouters for their careful reviews and valuable proposals for improving the document quality.

Contributors

The following individuals made substantial contributions to earlier draft versions of this document.

Sheela Rowles
Cisco Systems

Aldous Yeung
Cisco Systems
Email: cyyeung@cisco.com

Paulina Tran
Cisco Systems

Yoav Nir
Dell EMC
Email: ynir.ietf@gmail.com

Authors' Addresses

Valery Smyslov
ELVIS-PLUS
Russian Federation
Email: svan@elvis.ru

Brian Weis
Independent
United States of America
Email: bew.stds@gmail.com