                    Group Key Management using IKEv2
                     draft-ietf-ipsecme-g-ikev2-03

Abstract

   This document presents an extension to the Internet Key Exchange
   version 2 (IKEv2) protocol for the purpose of a group key management.
   The protocol is in conformance with the Multicast Security (MSEC) key
   management architecture, which contains two components: member
   registration and group rekeying. Both components require a Group
   Controller/Key Server to download IPsec group security associations
   to authorized members of a group. The group members then exchange IP
   multicast or other group traffic as IPsec packets. This document
   obsoletes RFC 6407.

Status of This Memo

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

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

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

   This Internet-Draft will expire on January 13, 2022.

Copyright Notice

Table of Contents

1. Introduction and Overview

A group key management protocol provides IPsec keys and policy to a
set of IPsec devices which are authorized to communicate using a
Group Security Association (GSA) defined in [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). This document presents an extension to
IKEv2 [RFC7296] called G-IKEv2, that allows to perform a group key
management.

G-IKEv2 conforms to the Multicast Group Security Architecture
[RFC3740], Multicast Extensions to the Security Architecture for the
Internet Protocol [RFC5374] and the Multicast Security (MSEC) Group
Key Management Architecture [RFC4046]. G-IKEv2 replaces GDOI
[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].

A GM begins a "registration" exchange when it first joins the group.
With G-IKEv2, the GCKS authenticates and authorizes GMs, then pushes
policy and keys used by the group to the GM. G-IKEv2 includes two
"registration" exchanges. The first is the GSA_AUTH exchange (
Section 1.4.1), which follows an IKE_SA_INIT exchange. The second is

the GSA_REGISTRATION exchange (Section 1.4.2), which a GM can use
within an established IKE SA. Group rekeys are accomplished using
either the GSA_REKEY pseudo-exchange (a single message distributed to
all GMs, usually as a multicast message), or as a GSA_INBAND_REKEY
exchange delivered individually to group members using existing IKE
SAs).

Large and small groups may use different sets of these protocols.
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. (Note: For clarity, IKE_SA_INIT is omitted from the
figure.)

```
                            +--------+
               +------------>|  GCKS  |<-------------+
               |            +--------+               |
               |             |    ^                  |
               |             |    |                  |
               |             | GSA_AUTH              |
               |             |   or                  |
               |             | GSA_REGISTRATION      |
               |             |    |                  |
         GSA_AUTH            |    |            GSA_AUTH
           or          GSA_REKEY |               or
       GSA_REGISTRATION          |    |       GSA_REGISTRATION
           |                |    |                  |
           |    +-----------+----------------+      |
           |    |           |    |           |      |
           v    v           v    v           v      v
        +-------+        +--------+        +-------+
        |  GM   |  ...   |   GM   |  ...   |  GM   |
        +-------+        +--------+        +-------+
            ^                ^                ^
            |                |                |
        +-------ESP-------+-------ESP------+
```
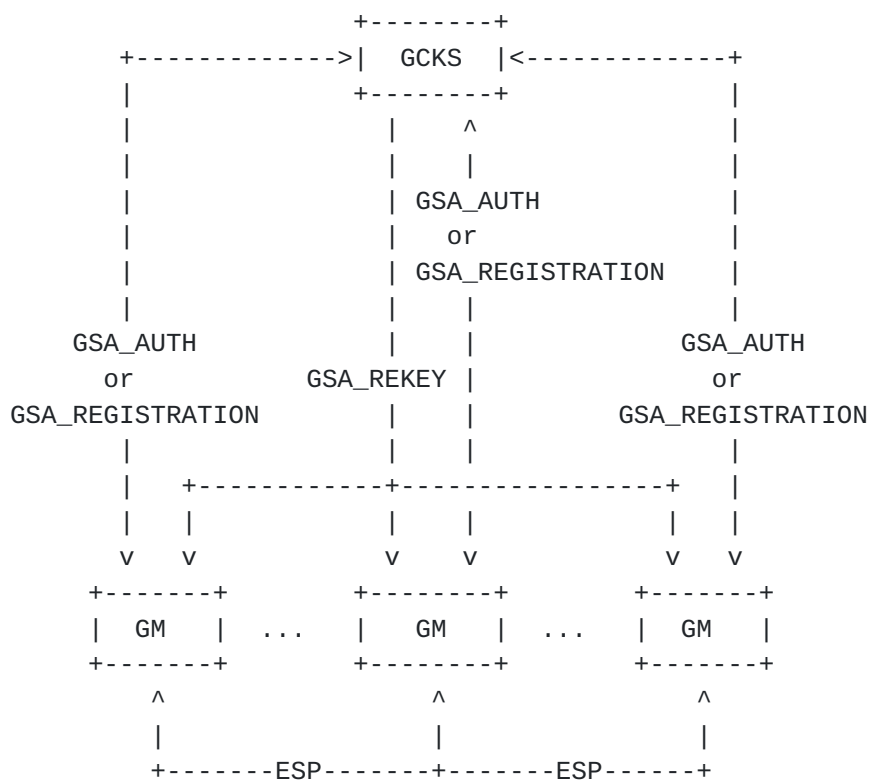
            Figure 1: G-IKEv2 used in large groups

Alternatively, a small group may simply use the GSA_AUTH as a
registration protocol, where the GCKS issues rekeys using the
GSA_INBAND_REKEY within the same IKEv2 SA. The GCKS is also likely
to be a GM in a small group (as shown in Figure 2.)

```
                    GSA_AUTH, GSA_INBAND_REKEY
         +---------------------------------------------------+
         |                                                   |
         |           GSA_AUTH, GSA_INBAND_REKEY              |
         |    +-----------------------------+                |
         |    |                             |                |
         |    | GSA_AUTH, GSA_INBAND_REKEY  |                |
         |    |    +--------+               |                |
         v    v    v        v               v                v
      +---------+    +----+          +----+           +----+
      | GCKS/GM |    | GM |          | GM |           | GM |
      +---------+    +----+          +----+           +----+
          ^            ^               ^                ^
          |            |               |                |
          +----ESP-----+------ESP-------+-----ESP-----+
```
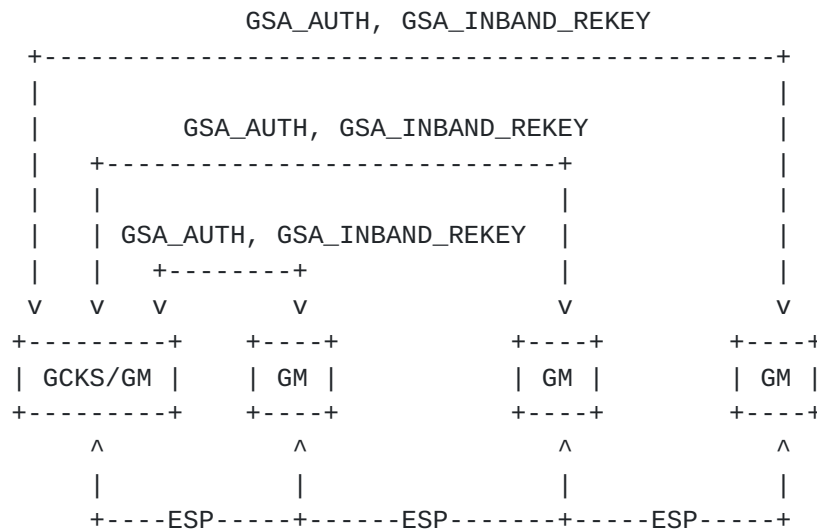
              Figure 2: G-IKEv2 used in small groups

   IKEv2 message semantics are preserved in that all communications
   consists 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 Language

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and
   "OPTIONAL" in this document are to be interpreted as described in BCP
   14 [RFC2119] [RFC8174] when, and only when, they appear in all
   capitals, as shown here.

1.2. G-IKEv2 Integration into IKEv2 Protocol

   G-IKEv2 uses the security mechanisms of IKEv2 (peer authentication,
   confidentiality, message integrity) to ensure that only authenticated
   devices have access to the group policy and keys. The G-IKEv2
   exchange further provides group authorization, and secure policy and
   key download from the GCKS to GMs. Some IKEv2 extensions require
   special handling if used with G-IKEv2. See Section 4 for more
   details.

   It is assumed that readers are familiar with the IKEv2 protocol, so
   this document skips many details that are described in [RFC7296].

1.2.1. **G-IKEv2 Transport and Port**

   G-IKEv2 SHOULD use UDP port 848, the same as GDOI [RFC6407], because
   they serve a similar function. They can use the same ports, just as
   IKEv1 and IKEv2 can share port 500. The version number in the IKE
   header distinguishes the G-IKEv2 protocol from GDOI protocol
   [RFC6407]. G-IKEv2 MAY also use the IKEv2 ports (500, 4500), which
   would provide a better integration with IKEv2. G-IKEv2 MAY also use
   TCP transport for registration (unicast) IKE SA, as defined in
   [RFC8229].

1.2.2. **IKEv2 Header Initialization**

   The Major Version is (2) and Minor Version is (0) according to IKEv2
   [RFC7296], and maintained in this document. The G-IKEv2 IKE_SA_INIT,
   GSA_AUTH, GSA_REGISTRATION and GSA_INBAND_REKEY use the IKE SPI
   according to IKEv2 [RFC7296], section 2.6.

1.3. **G-IKEv2 Protocol**

1.3.1. **G-IKEv2 Payloads**

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

```
     Notation       Payload
    -----------------------------------------------------------
     AUTH           Authentication
     CERT           Certificate
     CERTREQ        Certificate Request
     D              Delete
     GSA            Group Security Association
     HDR            IKEv2 Header
     IDg            Identification - Group
     IDi            Identification - Initiator
     IDr            Identification - Responder
     KD             Key Download
     KE             Key Exchange
     Ni, Nr         Nonce
     N              Notify
     SA             Security Association
     SAg            Security Association - GM Supported Transforms
```

   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:

o IDg (Group ID) - The GM requests the GCKS for membership into the
   group by sending its IDg payload.

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

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

o SAg (Security Association - GM Supported Transforms) - the GM
   sends supported transforms, so that GCKS may select a policy
   appropriate for all members of the group.

The details of the contents of each payload are described in
Section 3.

1.4. G-IKEv2 Member Registration and Secure Channel Establishment

The registration protocol consists of a minimum of two messages
exchanges, IKE_SA_INIT and GSA_AUTH; member registration may have a
few more messages exchanged if the EAP method, cookie challenge (for
DoS protection) or negotiation of Diffie-Hellman group is included.
Each exchange consists of request/response pairs. The first exchange
IKE_SA_INIT is defined in IKEv2 [RFC7296]. This exchange negotiates
cryptographic algorithms, exchanges nonces and does a Diffie-Hellman
exchange between the group member (GM) and the Group Controller/Key
Server (GCKS).

The second exchange GSA_AUTH authenticates the previous messages,
exchanges identities and certificates. These messages are encrypted
and integrity protected with keys established through the IKE_SA_INIT
exchange, so the identities are hidden from eavesdroppers and all
fields in all the messages are authenticated. The GCKS SHOULD
authorize group members to be allowed into the group as part of the
GSA_AUTH exchange. Once the GCKS accepts a group member to join a
group it will download the data security keys (TEKs) and/or group key
encrypting key (KEK) or KEK array as part of the GSA_AUTH response
message.

1.4.1. GSA_AUTH exchange

After the group member and GCKS use the IKE_SA_INIT exchange to
negotiate cryptographic algorithms, exchange nonces, and perform a
Diffie-Hellman exchange as defined in IKEv2 [RFC7296], the GSA_AUTH
exchange MUST complete before any other exchanges can be done. The
security properties of the GSA_AUTH exchange are the same as the
properties of the IKE_AUTH exchange. It is used to authenticate the
IKE_SA_INIT messages, exchange identities and certificates. G-IKEv2

also uses this exchange for group member registration and
authorization. Even though the IKE_AUTH does contain the SA2, TSi,
and TSr payload the GSA_AUTH does not. They are not needed because
policy is not negotiated between the group member and the GCKS, but
instead downloaded from the GCKS to the group member.

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

                     Figure 3: GSA_AUTH Request

After the IKE_SA_INIT exchange completes, the group member 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 emit data packets SHOULD
include a Notify payload status type of SENDER, which enables the
GCKS to provide any additional policy necessary by group senders.

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

                  Figure 4: GSA_AUTH Normal Response

The GCKS responds with IDr, optional CERT, and AUTH material as if it
were an IKE_AUTH. It also informs the group member of the
cryptographic policies of the group in the GSA payload and the key
material in the KD payload. The GCKS can also include a Delete (D)
payload instructing the group member to delete existing SAs it might
have as the result of a previous group member registration. Note,
that since the GCKS generally doesn't know which SAs the GM has, the
SPI field in the Delete payload(s) SHOULD be set to zero in this
case. (See more discussion on the Delete payload in Section 3.6.)

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 3.7, the GSA_AUTH response will not
include the GSA and KD, but will include a Notify payload indicating
errors. If the group member included an SAg payload, and the GCKS
chooses to evaluate it, and it detects that that group member cannot
support the security policy defined for the group, then the GCKS
SHOULD return a NO_PROPOSAL_CHOSEN. Other types of notifications can
be AUTHORIZATION_FAILED or REGISTRATION_FAILED.

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

                     Figure 5: GSA_AUTH Error Response

   If the group member finds the policy sent by the GCKS is
   unacceptable, the member SHOULD initiate GSA_REGISTRATION exchange
   sending IDg and the Notify NO_PROPOSAL_CHOSEN (see Section 1.4.2)).

1.4.2. GSA_REGISTRATION Exchange

   When a secure channel is already established between a GM and the
   GCKS, the GM registration for a group can reuse the established
   secure channel. In this scenario the GM will use the
   GSA_REGISTRATION exchange. Payloads in the exchange are generated
   and processed as defined in Section 1.4.1.

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

                 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 GCKS by evaluation of SAg payload). In this case the
   GCKS returns an appropriate error notification as described in
   Section 1.4.1.

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

                 Figure 7: GSA_REGISTRATION Error Exchange

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

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

        Figure 8: GM Reporting Errors in GSA_REGISTRATION Exchange

1.4.3. GM Registration Operations

   A G-IKEv2 Initiator (GM) requesting registration contacts the GCKS
   using the IKE_SA_INIT exchange and receives the response from the
   GCKS. This exchange is unchanged from the IKE_SA_INIT in IKEv2
   protocol.

   Upon completion of parsing and verifying the IKE_SA_INIT response,
   the GM sends the GSA_AUTH 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 initiator wishes to
   join. An initiator intending to emit data traffic SHOULD send a
   SENDER Notify payload status. The SENDER notification not only
   signifies that it is a sender, but provides the initiator the ability
   to request Sender-ID values, in case the data security SA supports a
   counter mode cipher. Section 1.4.6) includes guidance on requesting
   Sender-ID values.

   A GM may be limited in the types of Transforms that it is able or
   willing to use, and may find it useful to inform the GCKS which
   Transforms it is willing to accept for different security protocols.
   Proposals for Rekey SA (with protocol GIKE_REKEY) and for data
   security (AH [RFC4302] and/or ESP [RFC4303]) SAs may be included into
   SAg. Each Proposal contains a list of Transforms that the GM is able
   to support for that protocol. Valid transform types depend on the
   protocol and are defined in Figure 15. Other transform types SHOULD
   NOT be included. The SPI length of each Proposal in an SAg is set to
   zero, and thus the SPI field is empty. The GCKS MUST ignore SPI
   field in the SAg payload.

   Generally, a single Proposal of each type will suffice, because the
   group member is not negotiating Transform sets, simply alerting the
   GCKS to restrictions it may have. In particular, the restriction
   from Section 3.3 of [RFC7296] that AEAD and non-AEAD transforms must
   not be combined in a single proposal doesn't hold when the SAg
   payload is being formed. However if the GM has restrictions on
   combination of algorithms, this can be expressed by sending several
   proposals.

   Proposal Num field in Proposal substructure is treated specially in
   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 of different types having the same value in
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
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 SA, then only X can be
used in data security SAs, and the same for Y. To indicate this the
GM sends several Proposals marking those of them that must be used in
conjunction by putting the same value in their Proposal Num field.
In the simplest case when no dependency between transforms exists,
all Proposals in SAg payload will have the same value in Proposal Num
field.

Although the SAg payload is optional, it is RECOMMENDED for the GM to
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 inclusion one or
more the IPCOMP_SUPPORTED notifications along with the SAg payload.
The CPI in these notifications is set to zero and MUST be ignored by
the GCKS.

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

The GSA payload contains the security policy and cryptographic
protocols used by the group. This policy describes the Rekey SA
(KEK), Data-security SAs (TEK), and other group policy (GAP). 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 1.4.2). Note, that
this should normally not happen if the GM includes SAg payload in the
GSA_AUTH request and the GCKS takes it into account. Finally the KD
are parsed providing the keying material for the TEK and/or KEK. The
GM interprets the KD key packets, where each key packet includes the
keying material for SAs distributed in the GSA payload. Keying
material is matched by comparing the SPIs in the key packets 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.

The GSA KEK policy MUST include the attribute GSA_INITIAL_MESSAGE_ID
with a first Message ID the GM should expect to receive if it is non-
zero. 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
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 must have a Message ID
greater or equal to the Message ID received in the
GSA_INITIAL_MESSAGE_ID attribute.

Once a GM has received GSA_REKEY policy during a registration the IKE
SA may be closed. However, the GM SHOULD NOT close IKE SA, it is the
GCKS who makes the decision whether to close or keep it, because
depending on the policy the IKE SA may be used for inband rekeying
for small groups.

1.4.4. GCKS Registration Operations

A G-IKEv2 GCKS passively listens for incoming requests from group
members. When the GCKS receives an IKE_SA_INIT request, it selects
an IKE proposal and generates a nonce and DH to include them in the
IKE_SA_INIT response.

Upon receiving the GSA_AUTH request, the GCKS authenticates the group
member using the same procedures as in the IKEv2 IKE_AUTH. The GCKS
then authorizes the group member according to group policy before
preparing to send the GSA_AUTH response. If the GCKS fails to
authorize the GM, it will respond with an AUTHORIZATION_FAILED notify
message.

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, this policy is constructed in the GSA KEK and the
key is constructed in the KD KEK. The GSA KEK MUST include the
GSA_INITIAL_MESSAGE_ID attribute, specifying the starting Message ID
the GCKS will use when sending the GSA_REKEY message to the group
member if this Message ID is non-zero. 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 GAP MAY also be included to provide the ATD and/or DTD
(Section 3.4.3.1) specifying activation and deactivation delays for
SAs generated from the TEKs. If the group member 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 SIDs (see Section 1.4.6).

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.

If a group member includes an SAg in its GSA_AUTH or GSA_REGISTRATION
request, the GCKS MAY evaluate it according to an implementation
specific policy.

o The GCKS could evaluate the list of Transforms and compare it to
   its current policy for the group. If the group member did not
   include all of the ESP or AH Transforms in its current policy,
   then it could return a NO_PROPOSAL_CHOSEN Notification.

o The GCKS could store the list of Transforms, with the goal of
   migrating the group policy to a different Transform when all of
   the group members indicate that they can support that Transform.

o 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 need for the IKE SA
(e.g., it does not plan to initiate an GSA_INBAND_REKEY exchange).
If the GM does not initiate another registration exchange or Notify
(e.g., NO_PROPOSAL_CHOSEN), and also does not close the IKE SA and
the GCKS is not intended to use the SA, then after a short period of
time the GCKS SHOULD close the IKEv2 SA. The delay before closing
provides for receipt of a GM's error notification in the event of
packet loss.

1.4.5. 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 TEKs and KEK, deleting TEKs, and/or excluding group
members. 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 group members.

GSA_REKEY The GSA_REKEY is a pseudo-exchange initiated by the GCKS,
   where the rekey policy is usually delivered to group members using
   IP multicast as a transport. This is not a real IKEv2 exchange,
   since no response messages are sent. This method is valuable for
   large and dynamic groups, and where policy may change frequently
   and a scalable rekeying method is required. When the GSA_REKEY is
   used, the IKEv2 SA protecting the member registration exchanges is
   usually terminated, and group members 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 IKEv2 SA that was setup to protecting 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 the policy the GCKS MAY combine these two methods. For
   example, it may use the GSA_INBAND_REKEY to deliver key to the GMs in
   the group acting as senders (as this would provide reliable keys
   delivery), and the GSA_REKEY for the rest GMs.

1.4.5.1. GSA_REKEY

   The GCKS initiates the G-IKEv2 Rekey securely, usually using IP
   multicast. Since this rekey does not require a response and it sends
   to multiple GMs, G-IKEv2 rekeying MUST NOT support IKE SA windowing.
   The GCKS rekey message replaces the rekey GSA KEK or KEK array, and/
   or creates a new Data-Security GSA TEK. The SID Download attribute
   in the Key Download payload (defined in Section 3.5.3.2) 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:

```
 Members (Responder)           GCKS (Initiator)
 -------------------           ----------------
                         <--  HDR, SK{GSA, KD, [N,] [D,] [AUTH]}
```

                  Figure 9: GSA_REKEY Pseudo-Exchange

   HDR is defined in Section 3.1. The Message ID in this message will
   start with the value the GCKS sent to the group members in the KEK
   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 group members.

   The GSA payload contains the current rekey and data security SAs.
   The GSA may contain a new rekey SA and/or a new data security SA
   Section 3.4.

   The KD payload contains the keys for the policy included in the GSA.
   If the data security SA is 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
   is updated in the KD payload.

A Delete payload MAY be included to instruct the GM to delete
existing SAs.

The AUTH payload MUST be included to authenticate the GSA_REKEY
message if the authentication method is based on public key
signatures or a dedicated shared secret and MUST NOT be included if
authentication is implicit. In a latter case, the fact that a GM can
decrypt the GSA_REKEY message and verify its ICV proves that the
sender of this message knows the current KEK, thus authenticating
that the sender is a member of the group. Shared secret and implicit
authentication don't provide source origin authentication. For this
reason using them as authentication methods for GSA_REKEY is NOT
RECOMMENDED unless source origin authentication is not required (for
example, in a small group of highly trusted GMs). If AUTH payload is
included then the Auth Method field MUST NOT be NULL Authentication.

During group member registration, the GCKS sends the authentication
key in the GSA KEK payload, AUTH_KEY attribute, which the group
member uses to authenticate the key server. Before the current
Authentication Key expires, the GCKS will send a new AUTH_KEY to the
group members in a GSA_REKEY message. The AUTH key that is used in
the rekey message may be not the same as the authentication key used
in GSA_AUTH. If implicit authentication is used, then AUTH_KEY MUST
NOT be sent to GMs.

1.4.5.1.1. GSA_REKEY Messages Authentication

The content of the AUTH payload depends on the authentication method
and is either a digital signature or a result of prf applied to the
content of the not yet encrypted GSA_REKEY message.

The authentication algorithm (prf or digital signing) is applied to
the concatenation of two chunks: A and P. The chunk A lasts from the
first octet of the G-IKEv2 Header (not including prepended four
octets of zeros, if port 4500 is used) to the last octet of the
Encrypted Payload header. The 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 3.14 of [RFC7296] for description of the
Encrypted payload). In other words, the P chunk is the inner
payloads of the Encrypted payload in plaintext form. These inner
payloads must be fully formed and ready for encryption except for the
AUTH payload. Figure 10 illustrates the layout of the P and A chunks
in the GSA_REKEY message.

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 if Digital Signature authentication method is in use,

then the ASN.1 Length and the AlgorithmIdentifier fields must be
properly filled in, see [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).

```
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
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ^ ^
    |                    G-IKEv2 SA Initiator's SPI               | | |
    |                                                            | | |
    +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ I |
    |                    G-IKEv2 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 Authentication Method transform and the key
   provided before in the AUTH_KEY attribute. Depending on the
   authentication method the authentication data is either a digital
   signature or a result of applying prf from the Pseudorandom Function
   transform. 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 SKe/SKa keys.

   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 (that with Fragment Number
equal to 1).

1.4.5.1.2. 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. If the
message is using a new KEK attribute, the Message ID is reset to 0 in
this message. The GSA, KD, N and D payloads follow with the same
characteristics as in the GSA Registration exchange.

The AUTH payload (if present) is created as defined in
Section 1.4.5.1.1.

Because GSA_REKEY messages are not acknowledged and could be
discarded by the network, one or more GMs may not receive the
message. To mitigate such lost messages, during a rekey event the
GCKS may transmit several GSA_REKEY messages with the new policy.
The retransmitted messages MUST be bitwise identical and SHOULD be
sent within a short time interval (a few seconds) to ensure that
time-to-live 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 Sections Section 3.4.2.2.3 for more detail.

1.4.5.1.3. GSA_REKEY GM Operations

When a group member receives the Rekey Message from the GCKS it
decrypts the message using the current KEK, validates its
authenticity using the key retrieved in a previous G-IKEv2 exchange
if AUTH payload is present, verifies the Message ID, and processes
the GSA and KD payloads. The group member then downloads the new
data security SA and/or new rekey SA. The parsing of the payloads is
identical to the parsing done in the registration exchange.

Replay protection is achieved by a group member rejecting a GSA_REKEY
message which 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 or greater than the Message
ID it receives in the GSA_INITIAL_MESSAGE_ID attribute. Note, that
if no this attribute was 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.

If the GSA payload includes a Data-Security SA including a counter-
modes of operation and the receiving group member is a sender for
that SA, the group member uses its current SID value with the Data-
Security SAs to create counter-mode nonces. If it is a sender and
does not hold a current SID value, it MUST NOT install the Data-
Security SAs. It MAY initiate a GSA_REGISTRATION exchange to the
GCKS in order to obtain an SID value (along with current group
policy).

Once a new Rekey SA is installed as a result of GSA_REKEY message,
the current Rekey SA (over which the message was received) MUST be
silently deleted after waiting DEACTIVATION_TIME_DELAY interval
regardless of its expiration time. If the GSA TEK payload includes
GSA_REKEY_SPI attribute then after installing a new Data-Security SA
the old one, identified by the SPI in this attribute, MUST be
silently deleted after waiting 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.

1.4.5.1.4. IKE Fragmentation

IKE 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.

o 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.

o The GCKS must always use IKE fragmentation based on a known
   fragmentation threshold (unspecified in this memo), as there is no
   way to check if fragmentation is needed by first sending
   unfragmented messages and waiting for response.

o PMTU probing cannot be performed due to lack of GSA_REKEY response
   message.

1.4.5.2. GSA_INBAND_REKEY Exchange

When the IKEv2 SA protecting the member registration exchange is
maintained while group member participates in the group, the GCKS can
use the GSA_INBAND_REKEY exchange to individually provide policy
updates to the group member.

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

                  Figure 11: GSA_INBAND_REKEY Exchange

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

1.4.5.2.1. GSA_INBAND_REKEY GCKS Operations

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

1.4.5.2.2. GSA_INBAND_REKEY GM Operations

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

1.4.5.3. Deletion of SAs

There are occasions when the GCKS may want to signal to group members
to delete policy at the end of a broadcast, or if group policy has
changed. Deletion of keys MAY be accomplished by sending the G-IKEv2
Delete Payload [RFC7296], section 3.11 as part of the GSA_REKEY
pseudo-exchange as shown below.

```
 Members (Responder)            GCKS (Initiator)
--------------------           ------------------
                     <--    HDR, SK{[GSA,] [KD,], [N] [D,] [AUTH]}
```

                   Figure 12: SA Deletion in GSA_REKEY

The GSA MAY specify the remaining active time of the remaining policy
by using the DTD attribute in the GSA GAP. If a GCKS has no further
SAs to send to group members, the GSA and KD payloads MUST be omitted
from the message. There may be circumstances where the GCKS may want
to start over with a clean state. If the administrator is no longer
confident in the integrity of the group, the GCKS can signal deletion
of all the policies of a particular TEK protocol by sending a TEK

with a SPI value equal to zero in the delete payload. For example,
if the GCKS wishes to remove all the KEKs and all the TEKs in the
group, the GCKS SHOULD send a Delete payload with a SPI of zero and
Protocol ID of AH or ESP, followed by another Delete payload with a
SPI of zero and Protocol ID of GIKE_REKEY, indicating that the KEK SA
should be deleted.

1.4.6. Counter-based modes of operation

Several new counter-based modes of operation have been specified for
ESP (e.g., AES-CTR [RFC3686], AES-GCM [RFC4106], AES-CCM [RFC4309],
ChaCha20-Poly1305 [RFC7634], 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 Initialization
Vector (IV) using the same cipher key and mode. This requirement is
met in G-IKEv2 when the following requirements are met:

o The GCKS distributes a unique key for each Data-Security SA.

o 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 SID values.

1.4.6.1. Allocation of SIDs

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 SID to each group member acting in the
role of sender on the Data-Security SA. The SID value is used
exclusively by the group member to which it was allocated. The group
member uses the same SID 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 member can choose to request one or
more SID values. Requesting a value of 1 is not necessary since the
GCKS will automatically allocate exactly one to the group member. A
group member MUST request as many SIDs matching the number of
encryption modules in which it will be installing the TEKs in the
outbound direction. Alternatively, a group member MAY request more
than one SID and use them serially. This could be useful when it is
anticipated that the group member will exhaust their range of Data-
Security SA nonces using a single SID 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 SID values, which
ensures that each SID will be allocated to just one group member.

1. A GCKS maintains an SID-counter, which records the SIDs that have
    been allocated. SIDs are allocated sequentially, with zero as
    the first allocated SID.

2. Each time an SID is allocated, the current value of the counter
    is saved and allocated to the group member. The SID-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 member may request a count of SIDs
    during registration in a Notify payload information of type
    SENDER. When the GCKS receives this request, it increments the
    SID-counter once for each requested SID, and distributes each SID
    value to the group member. The GCKS SHOULD have a policy-defined
    upper bound for the number of SIDs that it will return
    irrespective of the number requested by the GM.

4. A GCKS allocates new SID values for each GSA_REGISTRATION
    exchange originated by a sender, regardless of whether a group
    member had previously contacted the GCKS. In this way, the GCKS
    is not required to maintaining a record of which SID values it
    had previously allocated to each group member. More importantly,
    since the GCKS cannot reliably detect whether the group member
    had sent data on the current group Data-Security SAs it does not
    know what Data-Security counter-mode nonce values that a group
    member has used. By distributing new SID values, the key server
    ensures that each time a conforming group member installs a Data-
    Security SA it will use a unique set of counter-based mode
    nonces.

5. When the SID-counter maintained by the GCKS reaches its final SID
    value, no more SID values can be distributed. Before
    distributing any new SID values, the GCKS MUST delete the Data-
    Security SAs for the group, followed by creation of new Data-
    Security SAs, and resetting the SID-counter to its initial value.

6. The GCKS SHOULD send a GSA_REKEY message deleting all Data-
    Security SAs and the Rekey SA for the group. This will result in
    the group members initiating a new GSA_REGISTRATION exchange, in
    which they will receive both new SID values and new Data-Security
    SAs. The new SID values can safely be used because they are only
    used with the new Data-Security SAs. Note that deletion of the
    Rekey SA is necessary to ensure that group members receiving a
    GSA_REKEY message before the re-register do not inadvertently use

their old SIDs with the new Data-Security SAs. Using the method
above, at no time can two group members use the same IV values
with the same Data-Security SA key.

1.4.6.2. GM Usage of SIDs

A GM applies the SID to data security SA as follows.

o The most significant bits NUMBER_OF_SID_BITS of the IV are taken
   to be the SID field of the IV.

o The SID is placed in the least significant bits of the SID field,
   where any unused most significant bits are set to zero. If the
   SID value doesn't fit into the NUMBER_OF_SID_BITS bits, then the
   GM MUST treat this as a fatal error and re-register to the group.

2. 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). An unrelated notion to PFS, "forward
access control" and "backward access control" have been called
"perfect forward security" and "perfect backward security" in the
literature [RFC2627].

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

The Group Key Management Method transform from the GSA policy
specifies how members of the group obtain group keys. This document
specifies a single method for the group key management - Wrapped Key
Download. This method assumes that all group keys are sent to the
GMs by the GCKS encrypted with other keys, called Key Wrap Keys
(KWK).

2.1. Key Wrap Keys

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

The GCKS may also send other keys to GMs that will be used as Key
Wrap Keys for the purpose of building key hierarchy. Each such key

is associated with an encryption algorithm from the Encryption
Algorithm transform used for the SA the key is sent in. The size of
such key MUST be of the size of the key size of this Encryption
Algorithm transform (taking into consideration the Key Length
attribute for this transform if present). This association persists
even if the key is used later in the context of another SA with
possibly different Encryption Algorithm transform.

To have an ability to provide forward access control the GCKS
provides each GM with a personal key at the time of registration.
Besides several intermediate keys that form a key hierarchy and are
shared among several GMs are provided by the GCKS.

2.1.1. Default Key Wrap Key

The default KWK (SK_w) is only used in the context of a single IKE
SA. Every IKE SA (unicast or group rekey) will have its own SK_w.
The SK_w is used with the algorithm from the Encryption Algorithm
transform used for the SA the SK_w is used in. The size of SK_w MUST
be of the key size of this Encryption Algorithm transform (taking
into consideration the Key Length attribute for this transform if
present).

For the unicast IKE SA (used for the GM registration and optionally
for GSA_INBAND_REKEY exchanges) the SK_w is computed as follows:

SK_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 SK_w is provided along with other SA
keys as defined in Section 2.4.

2.2. GCKS Key Management Semantics

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

o A simple key management policy - when the GCKS always sends group
   SA keys encrypted with the SK_w.

o An LKH key management policy - when the GCKS provides each GM with
   an individual key at the time of GM registration (encrypted with
   SK_w). Then the GCKS forms an hierarchy of keys so that the group
   SA keys are encrypted with other keys which are encrypted with
   other keys and so on, tracing back to the individual GMs' keys.

Other key policies may also be employed by the GCKS.

2.2.1. Forward Access Control Requirements

When group membership is altered using a group management algorithm
new GSA TEKs (and their associated keys) are usually also needed.
New GSAs 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, new GSA
TEKs and the associated key packets in the KD payload 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 key
packets in the KD payload are not protected with the new KEK. A
second G-IKEv2 rekey message can deliver the new GSA TEKS and their
associated key packets 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. Group members, which are denied access, will not be
able to access the new KEK, but 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.

2.3. GM Key Management Semantics

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

Each key is identified by a 32-bit number called Key ID. Zero Key ID
has a special meaning - it always contains keying material from which
the group SA keys are taken.

All keys in G-IKEv2 are transmitted in encrypted form, which format
is defined in Section 3.5.1. This format specifies a Key ID (ID of a
key that is encrypted in this attribute) and a KWK ID (ID of a key
that was used to encrypt this attribute). Keys may be encrypted
either with default KWK (SK_w) or with other keys, which the GM has
received in the KEY_WRAP_KEY attributes. If a key was encrypted with
SK_w, then the KWK ID field is set to zero, otherwise the KWK ID
field identifies the key used for encryption.

When a GM receives a message from the GCKS installing new data
security or rekey SA, it will contain a KD payload with a 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 (SK_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 policy employed by the GCKS. If
multiple SA_KEY attributes are present then all of them MUST contain
the same keying material encrypted using different keys. The GM in
general is unaware of the GCKS's key management policy and can always
use the same procedure to get the keys. In particular, the GM's task
is to find a way to decrypt at least one of the SA_KEY attributes
using either the SK_w or the keys from the KEY_WRAP_KEY attributes
that are present in the same message or were receives 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 update it when
needed. When the GSA_REKEY message is received the GM processes the
received SA_KEY attributes one by one trying to construct a new key
path that starts from this attributes and ends with any key in the
working Key Path or with the default KWK (SK_w).

In the simplest case the SA_KEY attribute is encrypted with SK_w so
that the new Key Path is empty. If more complex key management
policies are used then the Key Path will contain intermediate keys,
which will be from the KEY_WRAP_KEY attributes received in the same
messages. If the GM is able to construct a new Key Path, then it is
able to decrypt the SA_KEY attribute and use its content to form the
SA keys. If it is unable to build a new Key Path, then in 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:

o If the new Key Path is empty then no actions are needed. This may
   happen if no KEY_WRAP_KEY attributes from the received message
   were used.

o If the new Key Path is non-empty and it ends up with the default
   KWK (SK_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 it
   with its personal key and the other keys from the key tree that
   are needed for this GM. These keys form an initial working Key
   Path.

o In all other cases the new Key Path will end up where some key
   from the GM's working Key Path was used. In this case the new Key
   Path replaces the part of the GM's 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 policy.

2.4. Group SA Keys

Group SA keys are downloaded to GMs in the form of keying material.
The keys are taken from this keying material as if they were
concatenated to form it.

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 first
bits of the keying material and the integrity key (if any) MUST be
taken from the remaining bits.

For a group rekey SA the following keys are taken from the keying
material:

SK_e | SK_a | SK_w = KEYMAT

where SK_e and SK_a are the keys used for the Encryption Algorithm
and the Integrity Algorithm transforms for the corresponding SA and
SK_w is a default KWK for this SA. Note, that SK_w is also used with
the Encryption Algorithm transform as well as SK_e. Note also, that
if AEAD algorithm is used for encryption, then SK_a key will not be
used (GM can use the formula above assuming the length of SK_a is
zero).

3. 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 and several new payloads are defined: Group Identification
(IDg), Group Security Association (GSA) Key Download (KD). 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 processing
of existing IKEv2 payloads.

3.1. G-IKEv2 Header

G-IKEv2 uses the same IKE header format as specified in [RFC7296]
section 3.1. Major Version is 2 and 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].

3.2. Group Identification Payload

The Group Identification (IDg) payload allows the group member 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, 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 Group Identification payload is fifty (50).

3.3. Security Association - GM Supported Transforms Payload

The Security Association - GM Supported Transforms Payload (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 is identical to the SA Payload Type - thirty-three (33).

3.4. Group Security Association Payload

The Group Security Association (GSA) payload is used by the GCKS to
assert security attributes for both Rekey and Data-security SAs. The
Payload Type for the Group Security Association 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 13: GSA Payload Format

   The Security Association Payload fields are defined as follows:

   o Next Payload, C, RESERVED, Payload Length fields comprise the
      IKEv2 Generic Payload Header and are defined in Section 3.2. of
      [RFC7296].

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

3.4.1. Group Policies

   Croup policies are comprised of two types of policy - Group SA (GSA)
   policy and Group Associated (GA) policy. GSA policy defines
   parameters for the Security Association for the group. Depending on
   the employed security protocol GSA policies may further be classified
   as rekeying SA policy (GSA KEK) and data traffic SA policy (GSA TEK).
   GSA payload may contain zero or one GSA KEK policy, zero or more GSA
   TEK policies, and zero or one GA policy, where either one GSA KEK or
   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 Registration SA.
   Alternatively, group policy might use a Rekey SA but choose to
   download a KEK to the group member only as part of the Registration
   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 protected with an individual security association policy.

   A GAP allows for the distribution of group-wise policy, such as
   instructions for when to activate and de-activate SAs.

Policies are distributed in substructures to the GSA payload. The
format of the substructures is defined below in Section 3.4.2 (for
GSA policy) and in Section 3.4.3 (for GA policy). The first octet of
the substructure unambiguously determines its type - it is zero for
GAP and non-zero (actually, it is the security protocol ID) for GSA
policies.

3.4.2. Group Security Association Policy Substructure

The GSA policy substructure contains parameters for the SA used with
this group. Depending on the security protocol the SA is either a
rekey SA or a data security SA (ESP and AH). It is NOT RECOMMENDED
that the GCKS 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                  ~
  |                                                               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                                                               |
  ~                        <GSA Transforms>                       ~
  |                                                               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                                                               |
  ~                        <GSA Attributes>                       ~
  |                                                               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

                  Figure 14: GSA Policy Substructure Format

The GSA policy fields are defined as follows:

o Protocol (1 octet) - Identifies the security protocol for this
   group SA. The values are defined in the IKEv2 Security Protocol
   Identifiers in [IKEV2-IANA]. The valid values for this field are:

      <TBA> (GIKE_REKEY) for GSA KEK policy and 2 (AH) or 3 (ESP) for
      GSA TEK policy.

   o SPI Size (1 octet) - Size of Security Parameter Index (SPI) for
      the group SA. SPI size depends on the SA protocol. For
      GIKE_REKEY it is 16 octets, while for AH and ESP it is 4 octets.

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

   o 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 case of
      GIKE_REKEY the SPI must be the IKEv2 Header SPI pair where the
      first 8 octets become the "Initiator's SPI" field in the G-IKEv2
      rekey message IKEv2 HDR, and the second 8 octets become the
      "Responder's SPI" in the same HDR. When selecting SPI the GCKS
      MUST make sure that the sole first 8 octets (corresponding to
      "Initiator's SPI" field in the IKEv2 header) uniquely identify the
      Rekey SA.

   o Source & Destination Traffic Selectors - (variable) -
      Substructures describing the source and destination of the network
      identities. The format for these substructures is defined in
      IKEv2 [RFC7296], section 3.13.1. For the group rekey SA (protocol
      GIKE_REKEY) the destination traffic selectors MUST define a single
      IP address, IP protocol and port the GSA_REKEY messages will be
      destined to. The source traffic selector in this case MUST either
      define a single IP address, IP protocol and port the GSA_REKEY
      messages will be originated from or be a wildcard selector. For
      the data security (AH and ESP) SAs the traffic selectors instead
      specify characteristics of the traffic to be protected by the
      corresponding SA.

   o GSA Transforms (variable) - A list of Transform Substructures
      specifies the policy information for the group SA. The format is
      defined in IKEv2 [RFC7296], section 3.3.2. The Last Substruc
      value in each Transform Substructure will be set to 3 except for
      the last one in the list, which is set to 0. Section 3.4.2.1
      describes using IKEv2 transforms in GSA policy substructure.

   o GSA 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 group SA protocol and the group policy. Section 3.4.2.2
      defines attributes used in GSA policy.

3.4.2.1. GSA Transforms

GSA policy is defined by means of transforms in GSA policy
substructure. For this purpose the transforms defined in [RFC7296]
are used. In addition, new transform types are defined for using in
G-IKEv2: Authentication Method (AUTH) and Group Key Management Method
(GKM), see Section 6.

Valid Transform Types depend on group SA protocol and are summarized
in the table below.

```
Protocol     Mandatory Types                 Optional Types
-----------------------------------------------------------
GIKE_REKEY   ENCR, INTEG*, PRF, AUTH, GKM
ESP          ENCR                            INTEG, ESN
AH           INTEG                           ESN
```

Figure 15: Valid Transform Types

(*) If AEAD encryption algorithm is used, then INTEG transform MUST
NOT be specified, otherwise it MUST be specified.

3.4.2.1.1. Authentication Method Transform

The Authentication Method (AUTH) transform is used in the GIKE_REKEY
policy to convey information of how GCKS will authenticate the
GSA_REKEY messages. This values are from the IKEv2 Authentication
Method registry [IKEV2-IANA]. Note, that this registry defines only
256 possible values, so even that Transform ID field in the Transform
substructure allows for 65536 possible values, in case of the
Authentication Method transform the values 257-65535 MUST NOT be
used.

Among the currently defined authentication methods in the IKEv2
Authentication Method registry, only the following are allowed to be
used in the Authentication Method transform: Shared Key Message
Integrity Code, NULL Authentication and Digital Signature. Other
currently defined authentication methods MUST NOT be used. The
following semantics is associated with each of the allowed methods.

   Shared Key Message Integrity Code - GCKS will authenticates the
   GSA_REKEY messages by means of shared secret. In this case the
   GCKS MUST include the AUTH_KEY attribute containing the shared key
   into the KD payload at the time the GM is registered to the group.

   NULL Authentication - No additional 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.

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
Algorithm Identifier (<TBA by IANA>) is defined. This attribute
contains DER-encoded ASN.1 object AlgorithmIdentifier, which would
specify 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], see also [RFC7427] for
the list of common AlgorithmIdentifier values used in IKEv2. In
case of using digital signature the GCKS MUST include the
Algorithm Identifier attribute in the Authentication Method
transform.

The type of the Authentication Method Transform is <TBA by IANA>.

3.4.2.1.2. Group Key Management Method Transform

The Group Key Management Method (GKM) transform is used in the
GIKE_REKEY policy to convey information of how GCKS will manage the
group keys to provide forward and backward access control (i.e., used
to exclude group members). Possible key management methods are
defined in a new IKEv2 registry "Transform Type <TBA> - Group Key
Management Methods" (see Section 6). This document defines one
values for this registry:

Wrapped Key Download (<TBA by IANA>) - Keys are downloaded by GCKS
to the GMs in encrypted form. This algorithm may provide forward
and backward access control if some form of key hierarchy is used
and each GM is provided with a personal key at the time of
registration. Otherwise no access control is provided.

The type of the Group Key Management Method transform is <TBA by
IANA>.

3.4.2.1.3. Extended Sequence Number Transform

Extended Sequence Number (ESN) Transform is defined in [RFC7296] to
allow using 64-bit sequence numbers in ESP and AH. Since both AH
[RFC4302] and ESP [RFC4303] are defined so, that high-order 32 bits
of extended sequence numbers are never transferred on the wire, it
makes using ESN in multicast data security SAs problematic, because
GMs that join 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 [RFC4302] and
[RFC4303] is resource-consuming. For this reason extended sequence
numbers SHOULD NOT be used for multicast data security SAs and thus
the ESN Transform SHOULD NOT be included in the GSA Payload.

3.4.2.2. GSA Attributes

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

This document creates a new IKEv2 IANA registry for the types of the
GSA attributes which is initially filled as described in Section 6.
In particular, the following attributes are initially added.

```
GSA Attributes            Value  Type   Multiple  Used In
------------------------------------------------------------------------
Reserved                  0
GSA_KEY_LIFETIME          1      V      N         (GIKE_REKEY, AH, ESP)
GSA_INITIAL_MESSAGE_ID    2      V      N         (GIKE_REKEY)
GSA_NEXT_SPI              3      V      Y         (GIKE_REKEY, AH, ESP)
```

The attributes must follow the format defined in the IKEv2 [RFC7296]
section 3.3.5. In the table, attributes that are defined as TV are
marked as Basic (B); attributes that are defined as TLV are marked as
Variable (V).

3.4.2.2.1. GSA_KEY_LIFETIME Attribute

The GSA_KEY_LIFETIME attribute specifies the maximum time for which
the group SA is valid. The value is a 4 octet number defining a
valid time period in seconds. A single attribute of this type MUST
be included into any GSA policy substructure.

When the lifetime expires, the group security association and all
associated keys MUST be deleted. The GCKS may delete the SA at any
time before the end of the valid period.

3.4.2.2.2. GSA_INITIAL_MESSAGE_ID Attribute

The GSA_INITIAL_MESSAGE_ID attribute 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 MUST be included into the GSA KEK
   policy substructure if the initial Message ID is non-zero. Note,
   that it is always the case if GMs join the group after some multicast
   rekey operations have already taken place, so in these cases this
   attribute will be included into the GSA policy at the time of GMs'
   registration.

3.4.2.2.3. GSA_NEXT_SPI Attribute

   The optional GSA_NEXT_SPI attribute contains SPI that the GCKS
   reserved for the next group SA replacing this group SA. The length
   of the attribute data is determined by the SPI Size field in the GSA
   Policy substructure the attribute resides in (see Section 3.4.2), and
   the attribute data contains 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 and if later the GM starts receiving
   group SA messages with one of these SPIs without seeing a rekey
   message over the current rekey SA, this 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 passive GMs, i.e. those, that only listen and don't send
   data. There is also no point to include this attribute in the
   GSA_INBAND_REKEY messages, since they use reliable transport. Note
   also, 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 case of
   the GCKS reboot), so the GM must only treat these information as a
   "best effort" made by GCKS to prepare for future rekeys.

3.4.3. Group Associated Policy Substructure

   Group specific policy that does not belong to any SA policy can be
   distributed to all group member using Group Associated Policy (GAP)
   substructure.

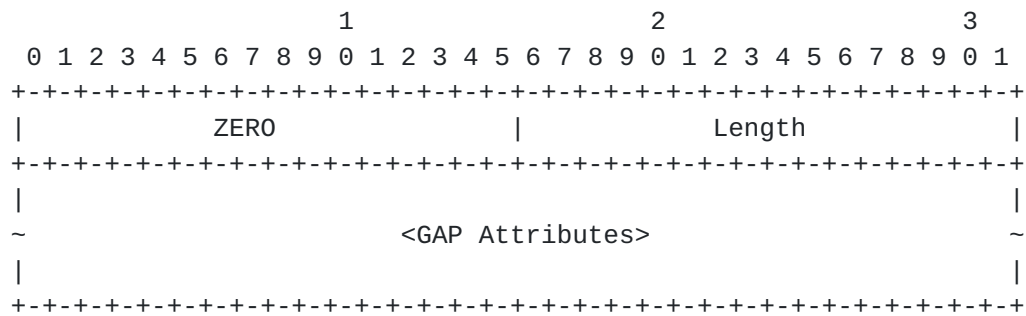   The GAP substructure is defined as follows:

```
                     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
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |              ZERO             |             Length            |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                                                               |
  ~                       <GAP Attributes>                        ~
  |                                                               |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

                    Figure 16: GAP Substructure Format

   The GAP substructure fields are defined as follows:

   o ZERO (2 octets) - MUST be zero.

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

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

   This document creates a new IKEv2 IANA registry for the types of the
   GAP attributes which is initially filled as described in Section 6.
   In particular, the following attributes are initially added.

```
         GAP Attributes                Value   Type    Multiple
         ---------------------------------------------------
         Reserved                        0
         GAP_ATD                         1      B       N
         GAP_DTD                         2      B       N
         GAP_SID_BITS                    3      B       N
```

   The attributes must follow the format defined in the IKEv2 [RFC7296]
   section 3.3.5. In the table, attributes that are defined as TV are
   marked as Basic (B); attributes that are defined as TLV are marked as
   Variable (V).

3.4.3.1. GAP_ATD And GAP_DTD Attributes

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

   The GAP_ATD attribute 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 staring 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 GAP_DTD attribute allows the GCKS to set the Deactivation Time
Delay for previously distributed SAs. The DTD defines how long after
receiving a request to delete data security SAs passive group members
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 GAP_ATD
attribute).

The GAP_ATD and GAP_DTD attributes contain 16 bit unsigned integer in
a network byte order, specifying the delay in seconds. These
attributes are OPTIONAL. If one of them or both are not sent by the
GCKS, the GMs should use default values for activation and
deactivation time delays.

3.4.3.2. GAP_SID_BITS Attribute

The GAP_SID_BITS attribute declares how many bits of the cipher nonce
are taken to represent an SID value. The bits are applied as the
most significant bits of the IV, as shown in Figure 1 of [RFC6054]
and specified in Section 1.4.6.2. Guidance for a GCKS choosing the
NUMBER_OF_SID_BITS is provided in Section 3 of [RFC6054]. This value
is applied to each SID value distributed in the KD payload.

The GCKS MUST include this attribute if there are more than one
sender in the group and any of the data security SAs use counter-
based cipher mode. The number of SID bits is represented as 16 bit
unsigned integer in network byte order.

3.5. Key Download Payload

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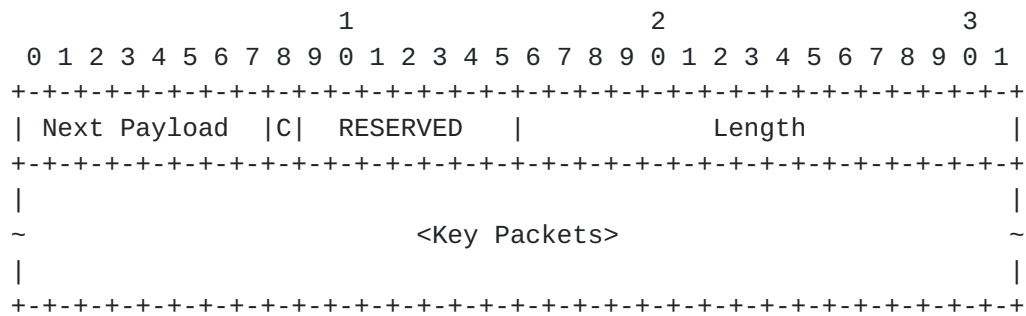payload is fifty-two (52).

```
                         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   |            Length             |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                        <Key Packets>                          ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

                   Figure 17: Key Download Payload Format

   The Key Download payload fields are defined as follows:

   o Next Payload, C, RESERVED, Payload Length fields comprise the
      IKEv2 Generic Payload Header and are defined in Section 3.2. of
      [RFC7296].

   o Key Packets (variable) - Contains Group Key Packet and Member Key
      Packet substructures. Each Key Packet contains keys for a single
      group rekey or data security SA or a keys and security parameters
      for a GM.

   Two types of Key Packets are used - Group Key Packet and Member Key
   Packet.

3.5.1. Wrapped Key Format

   The symmetric keys in G-IKEv2 are never sent in clear. They are
   always encrypted with other keys using the format called Wrapped Key
   that is shown below (Figure 18).

   The keys are encrypted using algorithm that is used to encrypt the
   message the keys are sent in. It means, that in case of unicast IKE
   SA (used for GMs registration and rekeying using GSA_INBAND_REKEY)
   the encryption algorithm will be the one negotiated during the SA
   establishment, while for the GSA_REKEY messages the algorithm will be
   provided by the GCKS in the Encryption Algorithm transform in the GSA
   payload when this multicast SA was being established (not in the same
   GSA_REKEY message).

   If AEAD mode is used for encryption, then for the purpose of key
   encryption the authentication tag MUST NOT be used (both not
   calculated and not verified), since the G-IKEv2 provides
   authentication of all its messages. In addition there is no AAD in
   this case. If encryption algorithm requires padding, then the
   encrypted key MUST be padded before encryption to have the required
   size. If the encryption algorithm doesn't define the padding

content, then the following scheme SHOULD be used: the Padding bytes
are initialized with a series of (unsigned, 1-byte) integer values.
The first padding byte appended to the plaintext is numbered 1, with
subsequent padding bytes making up a monotonically increasing
sequence: 1, 2, 3, ... The length of the padding is not transmitted
and is implicitly determined, since the length of the key is known.

```
                        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                             |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   ~                              IV                               ~
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                                                               |
   ~                         Encrypted Key                         ~
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

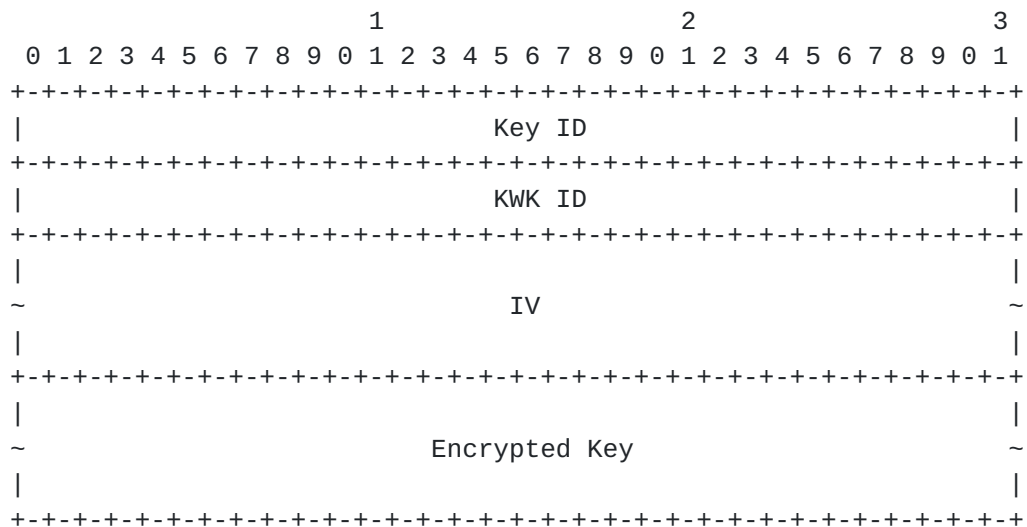   Figure 18: Wrapped Key Format

The Wrapped Key fields are defined as follows:

o Key ID (4 octets) - ID of the encrypted key. The value zero means
   that the encrypted key contains keying material for the group SA,
   otherwise it contains some intermediate key.

o Key Wrap Key (KWK) ID (4 octets) - ID of the key that was used to
   encrypt this key. The value zero means that the default KWK was
   used to encrypt the key, otherwise some other key was used.

o IV (variable) - Initialization Vector used for encryption. The
   size and the content of IV is defined by the encryption algorithm
   employed.

o Encrypted Key (variable) - The encrypted key bits. These bits may
   comprise either a single encrypted key or a result of encryption
   of a concatenation of keys (key material) for several algorithms.

3.5.2. Group Key Packet Substructure

Group Key Packet substructure contains SA key information. This key
information is associated with some group SAs: either with data
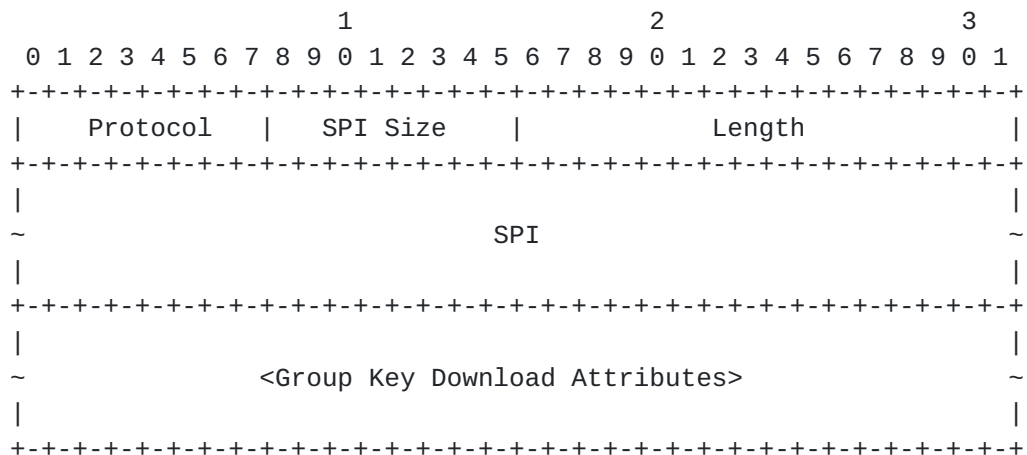security SAs or with group rekey SA.

```
                           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 Download Attributes>                   ~
     |                                                               |
     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

              Figure 19: Group Key Packet Substructure Format

   o Protocol (1 octet) - Identifies the security protocol for this key
      packet. The values are defined in the IKEv2 Security Protocol
      Identifiers in [IKEV2-IANA]. The valid values for this field are:
      <TBA> (GIKE_REKEY) for KEK Key packet and 2 (AH) or 3 (ESP) for
      TEK key packet.

   o SPI Size (1 octet) - Size of Security Parameter Index (SPI) for
      the corresponding SA. SPI size depends on the security protocol.
      For GIKE_REKEY it is 16 octets, while for AH and ESP it is 4
      octets.

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

   o SPI (variable) - Security Parameter Index for the corresponding
      SA. The size of this field is determined by the SPI Size field.
      In case of GIKE_REKEY the SPI must be the IKEv2 Header SPI pair
      where the first 8 octets become the "Initiator's SPI" field in the
      G-IKEv2 rekey message IKEv2 HDR, and the second 8 octets become
      the "Responder's SPI" in the same HDR. When selecting SPI the
      GCKS MUST make sure that the sole first 8 octets (corresponding to
      "Initiator's SPI" field in the IKEv2 header) uniquely identify the
      Rekey SA.

   o Group Key Download Attributes (variable length) - Contains Key
      information for the corresponding SA.

   This document creates a new IKEv2 IANA registry for the types of the
   Group Key Download attributes which is initially filled as described
   in Section 6. In particular, the following attributes are initially
   added.

```
        GKD Attributes       Value   Type    Multiple    Used In
        ------------------------------------------------------------
        Reserved               0
        SA_KEY                 1      V       Y           (GIKE_REKEY)
                                              N           (AH, ESP)
```

The attributes must follow the format defined in the IKEv2 [RFC7296] section 3.3.5. In the table, attributes that are defined as TV are marked as Basic (B); attributes that are defined as TLV are marked as Variable (V).

3.5.2.1. SA_KEY Attribute

The SA_KEY attribute contains a keying material for the corresponding SA. The content of the attribute is formatted according to Section 3.5.1 with a precondition that the Key ID field MUST 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 2.4) for the corresponding SA.

If the Key Packet 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 Packet is for a rekey SA (GIKE_REKEY protocol), then at least one SA_KEY attribute with zero Key ID MUST be present. Depending on GCKS key management policy more SA_KEY attributes MAY be present.

3.5.3. Member Key Packet Substructure

The Member Key Packet substructure contains keys and other parameters that are specific for the member of the group and are not associated with any particular group SA.
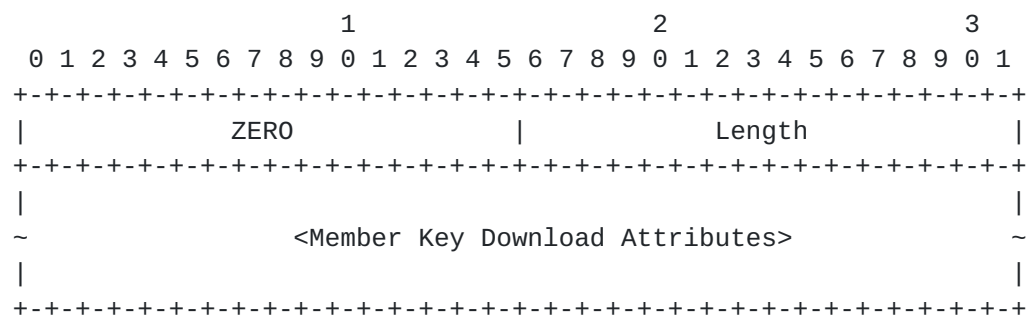
```
                      1                   2                   3
  0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |             ZERO              |             Length             |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 |                                                               |
 ~              <Member Key Download Attributes>                 ~
 |                                                               |
 +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

             Figure 20: Member Key Packet Substructure Format

The Member Key Packet substructure fields are defined as follows:

o ZERO (2 octets) - MUST be zero.

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

o Member Key Download Attributes (variable length) - Contains Key
   information and other parameters exclusively for a particular
   member of the group.

Member Key Packet 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 the
Member Key Download attributes which is initially filled as described
in Section 6. In particular, the following attributes are initially
added.

```
        MKD Attributes            Value   Type     Multiple
        ------------------------------------------------
        Reserved                  0
        KEY_WRAP_KEY              1       V        Y
        GM_SID                    2       V        Y
        AUTH_KEY                  3       V        N
```

The attributes must follow the format defined in the IKEv2 [RFC7296]
section 3.3.5. In the table, attributes that are defined as TV are
marked as Basic (B); attributes that are defined as TLV are marked as
Variable (V).

3.5.3.1. KEY_WRAP_KEY Attribute

The KEY_WRAP_KEY attribute contains a key that is used to encrypt
other keys. One or more the these attributes are sent to GMs if the
GCKS key management policy relies on some key hierarchy (e.g. LKH).

The content of the attribute has a format defined in Section 3.5.1
with a precondition that the Key ID field MUST NOT be zero. The
algorithm associated with the key is from the Encryption Transform
for the SA the KEY_WRAP_KEY attributes was sent in. The size of the
key MUST be equal to the key size for this algorithm.

Multiple instances of the KEY_WRAP_KEY attributes MAY be present in
the key packet.

3.5.3.2. GM_SID Attribute

The GM_SID attribute is used to download one or more Sender-ID (SID)
values for the exclusive use of a group member. One or more of this
attributes MUST be sent by the GCKS if the GM informed the GCKS that
it would be a sender (by inclusion the SENDER notification to the
request) and at least one of the data security SAs included in the
GSA payload uses counter-based mode of encryption.

If the GMs has requested multiple SID values in the SENDER
notification, then the GCKS SHOULD provide it with the requested
number of SIDs by sending multiple instances of the GM_SID attribute.
The GCKS MAY send fewer SIDs than requested by the GM (e.g. if it is
running out of SIDs), but it MUST NOT send more than requested.

3.5.3.3. AUTH_KEY Attribute

The AUTH_KEY attribute 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 Authentication Method
transform in the GSA payload.

o If a shared secret is used for the GSA_REKEY messages
   authentication then the content of the AUTH_KEY attribute is the
   shared secret that MUST be represented in the form of Wrapped Key
   (see Section 3.5.1) with zero KWK ID. The Key ID in this case is
   arbitrary and MUST be ignored by the GM.

o If digital signatures are used for the GSA_REKEY messages
   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 signature algorithm that will use this key was specified in
   the Algorithm Identifier attribute of the Authentication Method
   transform. The key MUST be compatible with this algorithm. An
   RSA public key format is defined in [RFC8017], Section A.1. DSS
   public key format is defined in [RFC3279] Section 2.3.2. For
   ECDSA Public keys, use format described in [RFC5480] Section 2.
   Other algorithms added to the IKEv2 Authentication Method registry
   are also expected to include a format of the SubjectPublicKeyInfo
   object included in the algorithm specification.

Multiple instances of the AUTH_KEY attributes MUST NOT be sent.

3.6. Delete Payload

There are occasions when the GCKS may want to signal to group members
to delete policy at the end of a broadcast, if group policy has
changed, or the GCKS needs to reset the policy and keying material
for the group due to an emergency. Deletion of keys MAY be
accomplished by sending an IKEv2 Delete Payload, section 3.11 of
[RFC7296] as part of a registration or rekey Exchange. Whenever an
SA is to be deleted, the GKCS SHOULD send the Delete Payload in both
registration and rekey exchanges, because GMs with previous group
policy may contact the GCKS using either exchange.

The Protocol ID MUST be GIKE_REKEY (<TBA>) for GSA_REKEY pseudo-
exchange, 2 for AH or 3 for ESP. Note that only one protocol id
value can be defined in a Delete payload. If a TEK and a KEK SA for
GSA_REKEY pseudo-exchange must be deleted, they must be sent in
different Delete payloads. Similarly, if a TEK specifying ESP and a
TEK specifying AH need to be deleted, they must be sent in different
Delete payloads.

There may be circumstances where the GCKS may want to reset the
policy and keying material for the group. The GCKS can signal
deletion of all policy of a particular TEK by sending a TEK with a
SPI value equal to zero in the delete payload. In the event that the
administrator is no longer confident in the integrity of the group
they may wish to remove all KEK and all the TEKs in the group. This
is done by having the GCKS send a delete payload with a SPI of zero
and a Protocol-ID of AH or ESP to delete all TEKs, followed by
another delete payload with a SPI value of zero and Protocol-ID of
KEK SA to delete the KEK SA.

3.7. Notify Payload

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

There are additional Notify Message types introduced by G-IKEv2 to
communicate error conditions and status (see Section 6).

o INVALID_GROUP_ID (45) - error type notification that indicates
   that the group id sent during the registration process is invalid.
   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.

o AUTHORIZATION_FAILED (46) - error type notification that is sent
   in the response to a GSA_AUTH 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.

o REGISTRATION_FAILED (<TBA>) - error type notification that is sent
   by the GCKS when the GM registration request cannot be satisfied.
   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.

o SENDER (16429) - 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 SID values the GM desires. The count MUST be 4 octets
   long and contain the big endian representation of the number of
   requested SIDs. The Protocol ID and SPI Size fields in the Notify
   payload MUST be zero.

o REKEY_IS_NEEDED (<TBA>) - status type notification that is sent in
   the GSA_AUTH response message to indicate that the GM must perform
   an immediate rekey of IKE SA to make it secure against quantum
   computers and then start a registration request over. 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.

3.7.1. USE_TRANSPORT_MODE Notification

This specification uses USE_TRANSPORT_MODE notification defined in
section 3.10.1 of [RFC7296] to specify which mode data security SAs
should be created in. The GCKS MUST include one USE_TRANSPORT_MODE
notification in a message containing the GSA payload for every data
security SAs specified in this payload that is to be created in
transport mode. In other words, there must be as many these
notifications included in the message as many SAs are created in
transport mode. The Protocol ID, SPI Size and SPI fields of the
Notify Payload MUST correctly specify each such SA.

3.8. Authentication Payload

G-IKEv2 uses the same Authentication payload as specified in
[RFC7296], section 3.8, 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 1.4.5.1.1.

4. Interaction with other IKEv2 Protocol Extensions

   A number of IKEv2 extensions is 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 Secure Password
   authentication ([RFC6467]). G-IKEv2 is compatible with and can use
   IKEv2 Session Resumption [RFC5723] except that a GM would include the
   initial ticket request in a GSA_AUTH exchange instead of an IKE_AUTH
   exchange. G-IKEv2 is also compatible with Multiple Key Exchanges in
   IKEv2 framework, defined in [I-D.ietf-ipsecme-ikev2-multiple-ke].

   Some IKEv2 extensions however require special handling if used in
   G-IKEv2.

4.1. Mixing Preshared Keys in IKEv2 for Post-quantum Security

   G-IKEv2 can take advantage of the protection provided by Postquantum
   Preshared Keys (PPK) for IKEv2 [RFC8784]. However, the use of PPK
   leaves the initial IKE SA susceptible to quantum computer (QC)
   attacks. For this reason an alternative approach for using PPK in
   IKEv2 defined in [I-D.smyslov-ipsecme-ikev2-qr-alt] SHOULD be used.

   If the alternative approach is not supported by the peers, then the
   GCKS MUST NOT send GSA and KD payloads in the GSA_AUTH response
   message. Instead, the GCKS MUST return a new notification
   REKEY_IS_NEEDED. Upon receiving this notification in the GSA_AUTH
   response the GM MUST perform an IKE SA rekey and then initiate a new
   GSA_REGISTRATION request for the same group. Below are possible
   scenarios involving using PPK.

   The GM starts the IKE_SA_INIT exchange requesting using PPK, and the
   GCKS responds with agreement to do it, or aborts according to its
   "mandatory_or_not" flag:

```
 Initiator (Member)                  Responder (GCKS)
 --------------------                -----------------
  HDR, SAi1, KEi, Ni, N(USE_PPK)  -->
                               <--  DR, SAr1, KEr, Nr, [CERTREQ],
                                    N(USE_PPK)
```

          Figure 21: IKE_SA_INIT Exchange requesting using PPK

   The GM then starts the GSA_AUTH exchange with the PPK_ID; if using
   PPK is not mandatory for the GM, the NO_PPK_AUTH notification is
   included in the request:

```
 Initiator (Member)                Responder (GCKS)
 --------------------              ------------------
 HDR, SK{IDi, AUTH, IDg,
 N(PPK_IDENTITY), N(NO_PPK_AUTH)}  -->
```

                  Figure 22: GSA_AUTH Request using PPK

If the GCKS has no such PPK and using PPK is not mandatory for it and
the NO_PPK_AUTH is included, then the GCKS continues without PPK; in
this case no rekey is needed:

```
 Initiator (Member)              Responder (GCKS)
 --------------------            ------------------
                         <--  HDR, SK{IDr, AUTH, GSA, KD}
```

                Figure 23: GSA_AUTH Response using no PPK

If the GCKS has no such PPK and either the NO_PPK_AUTH is missing or
using PPK is mandatory for the GCKS, the GCKS aborts the exchange:

```
 Initiator (Member)                Responder (GCKS)
 --------------------              ------------------
                         <--  HDR, SK{N(AUTHENTICATION_FAILED)}
```

                     Figure 24: GSA_AUTH Error Response

Assuming the GCKS has the proper PPK it continues with a request to
the GM to immediately perform a rekey by sending the REKEY_IS_NEEDED
notification:

```
 Initiator (Member)                Responder (GCKS)
 --------------------              ------------------
                         <--  HDR, SK{IDr, AUTH, N(PPK_IDENTITY),
                                       N(REKEY_IS_NEEDED) }
```

                  Figure 25: GSA_AUTH Response using PPK

The GM initiates the CREATE_CHILD_SA exchange to rekey the initial
IKE SA and then makes a new registration request for the same group
over the new IKE SA:

```
   Initiator (Member)                  Responder (GCKS)
   --------------------                ------------------
   HDR, SK{SA, Ni, KEi}  -->
                               <--  HDR, SK{SA, Nr, KEr}
   HDR, SK{IDg} --->
                               <--  HDR, SK{GSA, KD}
```

     Figure 26: Rekeying IKE SA followed by GSA_REGISTRATION Exchange

5. Security Considerations

5.1. GSA Registration and Secure Channel

   G-IKEv2 registration exchange uses IKEv2 IKE_SA_INIT protocols,
   inheriting all the security considerations documented in [RFC7296]
   section 5 Security Considerations, including authentication,
   confidentiality, protection against man-in-the-middle, protection
   against replay/reflection attacks, and denial of service protection.
   The GSA_AUTH and GSA_REGISTRATION exchanges also take advantage of
   those protections. In addition, G-IKEv2 brings in the capability to
   authorize a particular group member regardless of whether they have
   the IKEv2 credentials.

5.2. GSA Maintenance Channel

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

5.2.1. Authentication/Authorization

   Authentication is implicit, the public key of the identity is
   distributed during the registration, and the receiver of the rekey
   message uses that public key and identity to verify the message came
   from the authorized GCKS.

5.2.2. Confidentiality

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

5.2.3. Man-in-the-Middle Attack Protection

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

5.2.4. Replay/Reflection Attack Protection

The GSA_REKEY message includes a monotonically increasing sequence
number to protect against replay and reflection attacks. A group
member 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.

6. IANA Considerations

6.1. New Registries

A new set of registries is created for G-IKEv2 on IKEv2 parameters
page [IKEV2-IANA]. The terms Reserved, Expert Review and Private Use
are to be applied as defined in [RFC8126].

This document creates a new IANA registry "Transform Type <TBA> -
Group Key Management Methods". The initial values of the new
registry are:

```
Value                       Group Key Management Method
-------------------------------------------------------
Reserved                    0
Wrapped Key Download         1
Unassigned                  2-1023
Private Use              1024-65535
```

Changes and additions to the unassigned range of this registry are by
the Expert Review Policy [RFC8126].

This document creates a new IANA registry "GSA Attributes". The
initial values of the new registry are:

```
GSA Attributes          Value  Type   Multiple  Used In
----------------------------------------------------------------------
Reserved                  0
GSA_KEY_LIFETIME          1      V       N        (GIKE_REKEY, AH, ESP)
GSA_INITIAL_MESSAGE_ID    2      V       N        (GIKE_REKEY)
GSA_NEXT_SPI              3      V       Y        (GIKE_REKEY, AH, ESP)
Unassigned              5-16383
Private Use         16384-32767
```

Changes and additions to the unassigned range of this registry are by
the Expert Review Policy [RFC8126].

This document creates a new IANA registry "GAP Attributes". The
initial values of the new registry are:

```
GAP Attributes                Value   Type    Multiple
-----------------------------------------------------
Reserved                        0
GAP_ATD                         1       B       N
GAP_DTD                         2       B       N
GAP_SID_BITS                    3       B       N
Unassigned                    4-16383
Private Use             16384-32767
```

Changes and additions to the unassigned range of this registry are by
the Expert Review Policy [RFC8126].

This document creates a new IANA registry "Group Key Download
Attributes". The initial values of the new registry are:

```
GKD Attributes      Value   Type    Multiple    Used In
------------------------------------------------------------
Reserved              0
SA_KEY                1       V       Y           (GIKE_REKEY)
                                      N           (AH, ESP)
Unassigned          2-16383
Private Use     16384-32767
```

Changes and additions to the unassigned range of this registry are by
the Expert Review Policy [RFC8126].

This document creates a new IANA registry "Member Key Download
Attributes". The initial values of the new registry are:

```
MKD Attributes          Value   Type    Multiple
-------------------------------------------------
Reserved                  0
KEY_WRAP_KEY              1       V       Y
GM_SID                   2       V       Y
AUTH_KEY                 3       V       N
Unassigned             4-16383
Private Use        16384-32767
```

Changes and additions to the unassigned range of this registry are by
the Expert Review Policy [RFC8126].

6.2. Changes in the Existing IKEv2 Registries

This document defines new Exchange Types in the "IKEv2 Exchange
Types" registry:

```
Value          Exchange Type
-----------------------------
39             GSA_AUTH
40             GSA_REGISTRATION
41             GSA_REKEY
<TBA>          GSA_INBAND_REKEY
```

This document defines new Payload Types in the "IKEv2 Payload Types"
registry:

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

This document defines a new Security Protocol Identifier in the
"IKEv2 Security Protocol Identifiers" registry:

```
<TBA>          GIKE_REKEY
```

This document defines new Transform Types in the "Transform Type
Values" registry and changes the "Used In" column for the existing
allocations:

```
Type  Description                          Used In
---------------------------------------------------------------------
1     Encryption Algorithm (ENCR)          (IKE, GIKE_REKEY and ESP)
2     Pseudo-random Function (PRF)         (IKE, GIKE_REKEY)
3     Integrity Algorithm (INTEG)          (IKE, GIKE_REKEY, AH,
                                            optional in ESP)
4     Diffie-Hellman Group (D-H)           (IKE, optional in AH, ESP)
5     Extended Sequence Numbers (ESN)      (AH and ESP)
<TBA> Authentication Method (AUTH)         (GIKE_REKEY)
<TBA> Group Key Management Method (GKM)    (GIKE_REKEY)
```

This document defines a new Attribute Type in the "IKEv2 Transform
Attribute Types" registry:

```
Value          Attribute Type          Format
----------------------------------------------
<TBA>          Algorithm Identifier    TLV
```

This document defines new Notify Message Types in the "Notify Message
Types - Status Types" registry:

```
Value        Notify Messages - Status Types
--------------------------------------------
16429        SENDER
```

The Notify type with the value 16429 was allocated earlier in the
development of G-IKEv2 document with the name SENDER_REQUEST_ID.
This specification changes its name to SENDER.

This document defines new Notify Message Types in the "Notify Message
Types - Error Types" registry:

```
Value        Notify Messages - Error Types
------------------------------------------
45           INVALID_GROUP_ID
46           AUTHORIZATION_FAILED
<TBA>        REGISTRATION_FAILED
```

7. 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 versions of
the document.

8. Contributors

The following individuals made substantial contributions to early
versions of this memo.

```
   Sheela Rowles
   Cisco Systems
   170 W. Tasman Drive
   San Jose, California  95134-1706
   USA

   Phone: +1-408-527-7677
   Email: sheela@cisco.com
```

      Aldous Yeung
      Cisco Systems
      170 W. Tasman Drive
      San Jose, California 95134-1706
      USA

      Phone: +1-408-853-2032
      Email: cyyeung@cisco.com

      Paulina Tran
      Cisco Systems
      170 W. Tasman Drive
      San Jose, California 95134-1706
      USA

      Phone: +1-408-526-8902
      Email: ptran@cisco.com

      Yoav Nir
      Dell EMC
      9 Andrei Sakharov St
      Haifa 3190500
      Israel

      Email: ynir.ietf@gmail.com

9. References

9.1. Normative References

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

   [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>.

9.2. Informative References

   [I-D.ietf-ipsecme-ikev2-multiple-ke]
              Tjhai, C., Tomlinson, M., Bartlett, G., Fluhrer, S.,
              Geest, D. V., Garcia-Morchon, O., and V. Smyslov,
              "Multiple Key Exchanges in IKEv2", draft-ietf-ipsecme-
              ikev2-multiple-ke-02 (work in progress), January 2021.

   [I-D.smyslov-ipsecme-ikev2-qr-alt]
              Smyslov, V., "Alternative Approach for Mixing Preshared
              Keys in IKEv2 for Post-quantum Security", draft-smyslov-
              ipsecme-ikev2-qr-alt-03 (work in progress), February 2021.

   [IKEV2-IANA]
              IANA, "Internet Key Exchange Version 2 (IKEv2)
              Parameters", <http://www.iana.org/assignments/ikev2-
              parameters/ikev2-parameters.xhtml#ikev2-parameters-7>.

[NNL]   Naor, D., Noal, M., and J. Lotspiech, "Revocation and
            Tracing Schemes for Stateless Receivers", Advances in
            Cryptology, Crypto '01, Springer-Verlag LNCS 2139, 2001,
            pp. 41-62, 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", Manuscript,
             submitted to IEEE Transactions on Software Engineering,
            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>.

[RFC3279]   Bassham, L., Polk, W., and R. Housley, "Algorithms and
            Identifiers for the Internet X.509 Public Key
            Infrastructure Certificate and Certificate Revocation List
            (CRL) Profile", RFC 3279, DOI 10.17487/RFC3279, April
            2002, <https://www.rfc-editor.org/info/rfc3279>.

[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>.

[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>.

   [RFC5480] Turner, S., Brown, D., Yiu, K., Housley, R., and T. Polk,
              "Elliptic Curve Cryptography Subject Public Key
              Information", RFC 5480, DOI 10.17487/RFC5480, March 2009,
              <https://www.rfc-editor.org/info/rfc5480>.

   [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>.

   [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>.

   [RFC8017] Moriarty, K., Ed., Kaliski, B., Jonsson, J., and A. Rusch,
              "PKCS #1: RSA Cryptography Specifications Version 2.2",
              RFC 8017, DOI 10.17487/RFC8017, November 2016,
              <https://www.rfc-editor.org/info/rfc8017>.

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

[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>.

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 group members. 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 0{Y} means that
the default wrap key (SK_w) is used to encrypt key Y, and the
notation X{0} means key X is used to encrypt the group SA key. Note,
that 0{0} means that the group SA key is encrypted with default wrap
key.

The content of the KD payload will be shown as a sequence of Key
Packets. The Group Key Packet substructure will be denoted as SAn(),
when n is an SPI for the SA, and The Member Key Packet substructure
will be denoted as GM(). The content of the Key Packets is shown as
SA_KEY and KEY_WRAP_KEY attributes with the notation described above.
Here is the example of KD payload.

                KD(SA1(X{0}),GM(Y{X},Z{Y},0{Z})

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, i.e.
the relation between the keys. In this case the keys had the
following relation: Z{Y}, Y{X}.

A.2. Group Creation

When a GCKS forms a group, it creates a key tree as shown in the
figure below. 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(0), showing that its Key ID is always
zero). 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
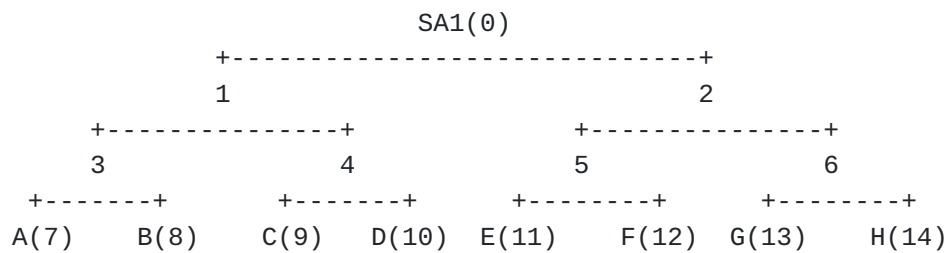group.

```
                              SA1(0)
                 +------------------------------+
                 1                              2
           +---------------+          +---------------+
           3               4          5               6
       +-------+       +-------+      +--------+      +--------+
     A(7)    B(8)    C(9)   D(10)  E(11)   F(12)  G(13)    H(14)
```

                    Figure 27: Initial LKH tree

When GM A joins the group, the GCKS provides it with the keys in the
KEY_WRAP_KEY attributes 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(SA1(1{0}),GM(3{1},7{3},0{7})

                   KD Payload for the Group Member A

From these attributes the GM A will construct the Key Path
0->1->3->7->0 and since it ends up with SK_w, it will use all the
KEY_WRAP_KEY attributes present in the path as its working Key Path:
1->3->7.

Similarly, when other GMs will be joining the group they will be
provided with the corresponding keys, so after all 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

A.3. Simple Group SA Rekey

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

                         KD(SA2(0{0}))

                   KD Payload for the Group Member F

All the GMs will be able to decrypt it and no changes in their
working Key Paths will take place.

A.4. Group Member Exclusion

If the GKCS 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 the key 2 with the key 15
and the key 5 with the key 16. It also a new SA key for a new SA3.
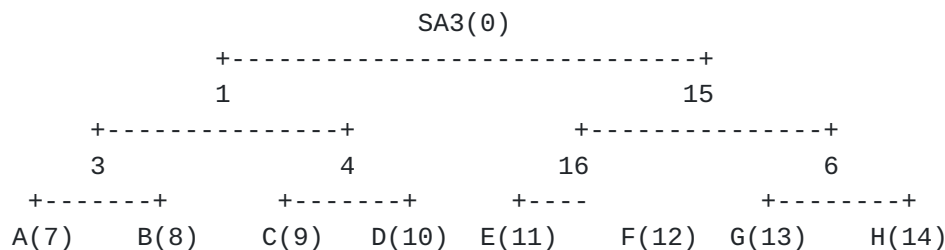
```
                         SA3(0)
                +------------------------------+
                1                              15
           +---------------+           +---------------+
           3               4           16              6
      +-------+       +-------+       +----           +--------+
    A(7)    B(8)    C(9)   D(10)   E(11)    F(12)   G(13)    H(14)
```

Figure 28: LKH tree after F has been excluded

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

KD(SA3(1{0},SA3(15{0})),GM(6{15},16{15},11{16})

KD Payload for the Group Member F

While processing this KD payload:

o GMs A, B, C and D will be able to decrypt the SA_KEY attribute
   1{0} 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.

o GMs G and H will construct new Key Path 15->0 and will be able to
   decrypt the new GM_KEY 15 using the key 6 from their working Key
   Paths. So, they will update their working Key Paths replacing
   their beginnings up to the key 6 with the new Key Path (thus
   replacing the key 2 with the key 15).

o GM E will construct new Key Path 16->15->0 and will be able to
   decrypt the new GM_KEY 16 using the key 11 from its working Key
   Path. So, it will update its working Key Path replacing its
   beginnings up to the key 11 with the new Key Path (thus replacing
   the key 2 with the key 15 and the key 5 with the key 16).

   o GM F won't be able to construct any Key Path leading to any key he
      possesses, so it will be unable to decrypt the new SA key for the
      SA3 and thus it will be excluded from the group once the GCKS
      starts sending TEK keys using SA3.

   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

Authors' Addresses

   Valery Smyslov
   ELVIS-PLUS
   PO Box 81
   Moscow (Zelenograd) 124460
   Russian Federation

   Phone: +7 495 276 0211
   Email: svan@elvis.ru

   Brian Weis
   Independent
   USA

   Email: bew.stds@gmail.com