Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 21, 2011

S. Hartman
Painless Security
D. Zhang
Huawei

October 18, 2010

Multicast Router Key Management Protocol (MRKMP)
draft-hartman-karp-mrkmp-00.txt

Abstract

Several routing protocols engage in one-to-many communication. In order to authenticate these communications using symmetric cryptography, a group key needs to be established. This specification defines a group protocol for establishing and managing such keys.

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 http://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 April 21, 2011.

Copyright Notice

described in the Simplified BSD License.

Table of Contents

1. Introduction

Many routing protocols such as OSPF and IS-Is use a one-to-many or multicast model of communications. The same message is sent to a number of recipients.

These protocols have cryptographic authentication mechanisms that use a key shared among all members of a communicating group in order to protect messages sent within that group. From a security standpoint, all routers in a group are considered equal. Protecting against a misbehaving router that is part of the group is out of scope for this protocol.

Routers need to be provisioned with some credentials for a one-to-one authentication protocol. Preshared keys or asymmetric keys and an authorization list are expected to be common deployments.

The members of a group elect a Group Controller/Key Server (GCKS). Potentially any member of the group may act as a GCKS. Since protecting against misbehaving routers is out of scope, there is no need to protect against a node that is not currently the GCKS impersonating the GCKS.

To prove membership in the group, a router authenticates using its provisioned credentials to the current GCKS. If successful, the router is given the current key material for the group. Group size is relatively small and need for forced eviction of members is rare. If a GCKS needs to evict a member, then it can simply re-authenticate with the existing members and provide them new key material.

1.1. Terminology

One key terminology question to answer is the definition of group. It appears that as used in this document, the term group corresponds to a routing protocol instance on a single link. However, this needs to be confirmed with TE routing protocols and with PIM. If that works out then a more precise term than group should be used in this document.

1.2. Relationship to IKEv2

IKEv2 [RFC5996]provides a protocol for authenticating IPsec security associations between two peers. It currently provides no group keying. IKev2 is attractive as a basis for this protocol because while it is much simpler than IKE, it provides all the needed flexibility in one-to-one authentication.

Unlike IKE, IKEv2 is explicitly designed for IPsec. The document

does not separate handling of aspects of the protocol that would be needed for IPsec from those that apply to general key management. IPsec specific rules are combined with more general requirements. While concepts and protocol payloads can be used in a different key management protocol, the current structure of IKEv2 does not provide a mechanism for applying IKEv2 to a domain of interpretation other than IPsec. In addition, the complexity required in the IKE specification when compared to IKEv2 suggests that the generality of IKE may not be worth the complexity cost.

For these reasons, this protocol borrows concepts and payloads from IKEv2 but does not normatively depend on the IKEv2 specification.

1.3. Relationship to GDOI

The IPsec Group Domain of Interpretation (GDOI) [RFC3547] provides a protocol that is structurally very similar to this one. As specified, IKE can be used to provide phase 1 authentication to a GCKS. After that, GDOI provides phase 2 messages to establish key-encryption keys and traffic keys. Key management operations can be accomplished via GDOI messages sent to the group after the phase 2 exchange.

GDOI is defined for IKE not for IKEv2. In addition, GDOI’s phase 2 uses its own hashing mechanism and nonce mechanism to provide integrity protection and replay protection. Like IKE, GDOI has significant complexity to support phase 2 identities that are different than the phase 1 identity. GDOI requires a GCKS to have a signature key used to sign GDOI messages when the rekey protocol is used. Since attacks caused by members of the group masquerading as the GCKS are out of scope, this is significant unnecessary complexity in the protocol.

This protocol can be thought of as a simplified GDOI based on IKEv2 rather than IKE. However, integrity and replay mechanisms are taken from IKEv2. Support for phase 2 identities is removed as unneeded complexity. Security for the group key management messages is provided using symmetric primitives rather than asymmetric signatures. Phase 1 authentication will often still involve asymmetric signatures.

2. Overview

MRKMP is composed of several parts. There is an initial exchange used to establish a shared key with a GCKS and authenticate the identities of both parties. Unicast key management exchanges provide the ability to join a group or request updates to the group; group joins can also be combined with the initial exchange. There is an election protocol used by routers to determine which router will act as the GCKS; this protocol is not integrity protected, but a GCKS confirms its role when a member uses the unicast exchange to join the group. Finally, a GCKS uses multicast exchanges to update parameters of the group. This section briefly describes each of these parts of MRKMP. The later sections in the document describe the details of the protocols.

2.1. Types of Keys

MRKMP manipulates several different types of symmetric keys:

preshared: Preshared keys are one mechanism for authenticating one router to another during the initial exchange. These keys are configured by some mechanism such as manual configuration or a management application outside of the scope of MRKMP. A single preshared key can be used for all members of a group. Alternatively each pair of routers can have a different preshared key.

peer key management key: Routers share a key with the GCKS that is a result of the mrkmp_init exchange.

KEK: A Key encryption Key (KEK) is a key used to encrypt group key management messages to the current members of a group. A KEK is learned as the product of establishing an MRKMP association or through a group key management message encrypted in a previous KEK. A KEK has an explicit expiration but may also be retired by a message encrypted in the KEK sent by the GCKS.

protocol master key: A protocol master key is the key exported by MRKMP for use by a routing protocol such as OSPF or IS-IS. The Protocol master key is the key that would be manually configured if a routing protocol is used without key management.

transport key: The transport key is the key used to integrity protect routing messages in a protocol such as IS-IS or OSPF. In today's routing protocol cryptographic authentication mechanisms the transport key is the same as the protocol master key. A disadvantage of this approach is that replay prevention is challenging with this architecture. Ideally some key derivation step would be used to establish a fresh transport key among all the participants in the group.

2.1.1. Key Encryption Key

When a router wishes to join a group, the router performs the mrkmp_init and mrkmp_auth exchange with a GCKS. During this process the router can establish an association with a specific group. Part of that association will be delivery of a KEK and associated parameters.

Group key management messages are sent to a group address not unicast to an individual peer. The group key management messages are protected using the KEK. The group key management messages need to provide both integrity and confidentiality protection using the KEK.

As part of establishing the association, the router joining the group is given an expiration time for the KEK. A group key management message may establish a new KEK with new parameters.

From time to time, a GCKS may wish to either force early expiration of a KEK or allow a KEK to expire. Protocol master keys are permitted to be valid for somewhat longer than the KEK that created them so as to avoid disrupting routing when this happens. When a KEK is retired or expires without being replaced by a new KEK announced in the old KEK, group members need to perform a new initial exchange to the GCKS. This is useful for example if a router is no longer authorized to be part of the group.

Other mechanisms such as LKH (section 5.4 [RFC2627]) could be used to permit removal of a group member while avoiding new initial authentications. However these mechanisms come at a complexity cost that is not justified for a small number of routers participating in a single multicast link.

2.1.2. Protocol Keys

Current routing protocols directly use the protocol master key to integrity protect messages. One advantage for this approach is that the initial hello messages used for discovery and capability exchange can be protected using the same mechanism as other messages. Typically a sequence number is used for replay detection. Without

changing the key, the existing protocols are vulnerable to a number of serious denial of service attacks from replays.

The MRKMP can solve this replay problem by changing the protocol master key whenever a peer is about to exhaust its sequence number space or whenever a peer loses information about what sequence numbers it used. This could potentially involve changing the protocol master key whenever a router reboots that was part of the group using the current protocol master key. Since key changes will not disrupt active adjacencies and can be accomplished relatively quickly, this is not expected to be a huge problem. Note that after one key change, others routers can boot without causing additional key changes; a flurry of key changes would not be required if several routers reboot near each other.

Another approach would be to separate the protocol master key from the transport keys. For example the transport key used by a given peer could be a fresh key derived from the protocol master key and nonces announced by that peer. Some mechanism would need to make sure that the peer's announcement of its nonce was fresh; this mechanism would almost certainly involve some form of interaction with the router wishing to guarantee freshness. There are two key advantages of this separation between transport keys and protocol master keys. The first is that the interaction between the MRKMP and routing protocol can be simplified significantly. The second is that even when manually configured protocol master keys are used, replay and adequate DOS protection can be achieved.

2.2. GCKS Election

Before a MRKMP system actually starts working, the routers in the multicast group need to select a GCKS so that they can obtain cryptographic keys to secure subsequent exchanges of routing information. MRKMP specifies an election protocol that dynamically assigns the responsibility of key management to one of the group members. Note that there are already announcer-electing mechanisms provided in some routing protocols (e.g., OSPF and IS-IS). However, much involvement between a MRKMP system and a routing protocol implementation will be introduced if the MRKMP system reuses the announcer-electing mechanism for the election of the GCKS. The state machine of the routing protocol also has to be modified. For instance, in OSPF, after a DR has been elected, routers need to halt their OSPF executions, and carry out the initial exchange to authenticate the DR and collect the keys for subsequent communications. After this step, the routers need to re-start their OSPF state machines so as to exchange routing information. As a consequence of such cases, an individual GCKS electing solution within MRKMP is preferable.

Each router has a GCKS priority. Higher priorities are more preferred GCKSes. As discussed in Section 8, the routing protocol can influence the GCKS election protocol by manipulating the priority so that it is likely that the same router will be the announcer for the routing protocol and the GCKS. Even if two different routers are elected as the announcer and GCKS, then the routing protocol and MRKMP will function correctly.

2.3. Initial Exchange

The initial exchange is based on IKEv2's IKE_SA_INIT and IKE_SA_AUTH exchanges. During this exchange, an initiating router attempts to authenticate to the router it believes is a GCKS for a group that the initiating router wants to join. Messages are unicast from the initiator to the responding GCKS. Unicast MRKMP P messages form a request/response protocol; the party sending the messages is responsible for retransmissions.

The initial exchange provides capability negotiation, specifically including supported cryptographic suites for the key management protocol. Identification of the initiator and responder is also exchanged. A symmetric key is established to integrity protect and encrypt key management messages. While routing security does not typically require confidentiality, the key management protocol does because keys are exchanged and these must be protected.

Then the identities of each party are cryptographically verified. This can be done using a preshared key or symmetric keys. Other mechanisms may be added as a future extension.

The authentication exchange also provides an opportunity to join a group as part of the initial exchange. In the typical case, a router can obtain the needed key material for a group in two round-trips.

2.4. Group Join Exchange

The primary purpose of the unicast MRKMP messages is to get an initiator the information it needs to join a group and participate in a routing protocol. The initiator indicates what group it wants to join. XXX we need to discuss group naming--if MRKMP is limited to a subnet this may be as simple as saying that initiator wants to join the OSPF group or the IS-IS group.

The responder performs several checks. First, the responder confirms that the responder is currently acting as GCKS for the group in question. Then, the responder confirms that the initiator is permitted to join the group. If these checks pass, then the responder provides a key download payload to the initiator encrypted

in the peer key management key. As discussed in Section 2.1.2, the GCKS MUST change the protocol master key if a router was part of the group under the current protocol master key and reboots. In this case, the GCKS SHOULD provide the new and old protocol master key to the initiator, setting the validity times for the old key to permit reception but not transmission. The GCKS MUST use the mechanism in the next section to flood the new key to the rest of the group.

A group association created by this exchange may last beyond the unicast MRKMP association used to create it. Once membership in a group is established, resources are not required to maintain the unicast association with the GCKS.

A member of a group can also use the unicast exchange to request a GCKS to change the protocol master key because that group has exhausted its available sequence space. For protocols where the protocol master key is the same as the transport key, it is critical that no two messages be sent by the same router with the same sequence number and protocol master key. The sequence number space is finite. So if a router is running low on available sequence space it needs to request a new protocol master key be generated.

2.5. Group Key Management

The GCKS shares a KEK with all members of a group. The GCKS can send a multicast message to the group to update the set of protocol master keys, update the KEK, or retire the KEK and request new group join exchanges.

Typically the protocol master key is changed only when needed to provide replay protection or when the KEK changes. The KEK changes whenever a new GCKS is elected or whenever it is administratively desirable to change the keys. For example if an employee leaves an organization it might be desirable to change the KEKs. A KEK is retired whenever forward security is desired: whenever the authorization of who is permitted to be in a group changes and the GCKS needs to make sure that the router is no longer participating. Most authorization changes such as removing a router from service do not require forward security in practical deployments.

3. GCKS Election

The GCKS election process selects a single router on a link to act as GCKS for a group.Similar with other popular announcer electing mechanisms (e.g., VRRP, HSRP), in MRKMP, only GCKSes use multicast to periodically send Advertisement messages. Such advertisements can be used as heart beat packets to indicate the aliveness of GCKSes. In addition, a state machine with three states (Initial, GCKS, and Member) is specified for GCKS election. When a router is initially connected to a multicast network, its state is set as Initial. The router then sends a multicast initial advertisement, if a GCKS is working on the network, it will reply the router with an advertisement using unicast. After receiving the advertisement from the GCKS, the router will try to register with the GCKS using the initial exchange, and then the state of the router is transferred to Member. Note that when the router receives the advertisement it does not have the traffic distributed in the group. Thus, the integrity of the unicast advertisement does not have to be protected. After a certain period, if the router still does not receive any advertisement from a GCKS or other group members, the router then believe there is no other group member on the network and set its state as GCKS. If during the period the router does not receive any advertisement from a GCKS but receives advertisements from other routers on the network, router believes that the group is involved in a GCKS election process. Apart from the initialization of a multicast network, the fail-over of a GCKS can also trigger an election process. For instance, if a router does not receive the heart beat advertisement for a certain period, it will transfer its state to Initial and try to elect a new one. In a GCKS electing process, a router has to stay in the Initial state until a new GCKS is allocated. Particularly, the router first sends its initial advertisement with its priority and waits for a certain period. During the period, if a router receives an initial advertisement which consists of a lower priority, the router then sends the advertisement again with a limited rate. After period, if the router does not find any router with a higher priority, it announces itself as the GCKS. If two routers have the same priority, the one with the lowest IP source address used for messages on the link will be the GCKS. After a router transfer its state to GCKS, it will reply to the initial advertisements from other routers with GCKS advertisements, even when the initial advertisements consist of properties priorities than its priority. This approach guarantees that a GCKS will not be changed frequently after it has been elected. After receiving the GCKS advertisement of the new elected GCKS, other routers transfer their states to Member. However, if a GCKS G1 receives a GCKS advertisement from another router G2 and G2 is a more preferred GCKS, G1 follows the procedure in Section 3.2.

If a node in state member fails to perform an initial exchange with the router it believes to be GCKS, it resets its state to initial but ignores advertisements from that router. This way an attacker cannot disrupt communications indefinitely by masquerading as a GCKS.

If a node transitions to GCKS state, it performs the procedure in Section 3.1.

3.1. A new GCKS is Elected

3.2. Merging Partitioned Networks

Whenever a GCKS finds that a more preferred router is also acting as a GCKS for the same group, then the group is partitioned. Typically if there is already an active GCKS for a group, even if a more preferred GCKS joins, the GCKS will not change. Two situations can result in multiple GCKSes active for a group. The first is that members of the group do not share common authentication credentials. The second is that the group was previously partitioned so that some nodes could not see election messages from other nodes. After the problem resulting in the partition is fixed, then both active GCKSes will see each others election announcements. The group needs to merge.

The less preferred GCKS performs a unicast mrkmp_merge_sa unicast key management message to the more preferred GCKS. In this message the less preferred GCKS includes its key download payload, so the more preferred GCKS learns the protocol master keys of the less preferred GCKS.

The more preferred GCKS generates a new key download payload including a KEK and the union of all the protocol master keys. The GCKS SHOULD mark the existing protocol master keys as expiring for usage in transmitted packets in a relatively short time. The GCKS SHOULD introduce a new protocol master key. This key download payload is returned to the less preferred GCKS and is sent out in the current KEK using a group key management message.

The less preferred GCKS sends the received key download payload encrypted in its existing KEK. XXX how many retransmits. After all retransmissions of this payload the less preferred GCKS sets its state to member.

As a result of this procedure, members learn the protocol master keys of both GCKSes and converge on a single KEK and GCKS. Changing the protocol master keys during a merge is important for protocols that use the protocol master key as a transport key. The new GCKS does not know which routers have joined the group with the other GCKS.

Therefore, it could not correctly detect one of these routers rebooting and change the protocol master key at that point. If the key is changed as part of the merge, replays are handled.

4. Key Download Payload

What all is actually in the message you get at the end of phase 2 and that is sent out periodically during group key management

For the KEK, this needs to include the key itself, the algorithm (presumably drawn from the IKEv2 symmetric algorithms), key ID, group ID and the four lifetimes.

The protocol master keys include the key, an algorithm ID, the key ID and the four lifetimes.

By four lifetimes we mean receive start, send start, send end and receive end. It’s important that a key can be flooded out to all potential receivers before it is used for sending.

5. Initial Exchange Details

6. Group Management Unicast Exchanges

6.1. Group Join Exchange

If a router receives a group join exchange for a group for which it is not the GCKS, it MUST return a notification. If it knows the GCKS for the group then it returns MRKMP_WRONG_GCKS including the address of the GCKS in the notification payload. The initiator tries the group join exchange (probably with a new initial exchange) with the indicated router. If the responder does not know the GCKS for the group, either because it is not a member of the group or because its GCKS election state is initial, it returns the MRKMP_GCKS_UNKNOWN notification. If the responder is not trying to be a member of this group or has seen a more preferred GCKS advertisement in the election process then the potential_candidate bit is clear, otherwise it is set. The initiator sets its GCKS election state to initial when receiving this notification. If the potential candidate bit is set in the notification then the initiator will accept GCKS election advertisements from the responder. If the potential candidate bit is clear, then the initiator will discard GCKS election advertisements from the responder until BLACKLIST_TIMEOUT seconds have elapsed or until the initiator successfully joins the group.

7. Group Key Management Operation

Group key management messages are multicast from the GCKS to the group. The message contains the key identifier of a KEK, as well as encrypted/integrity-protected payloads. Inside the encrypted/ integrity-protected payloads is a monotonically increasing sequence number, and payloads specific to the message being sent. Group members MUST ignore a message with a sequence number that is the same or less than the sequence number of the most recent message they have received.

7.1. General operation

Periodically the GCKS will send out an update message encrypted in the current KEK including the current group key download payload and parameters. If a new KEK is about to be valid for receiving messages, this is included. Any protocol master keys that are valid for sending or receiving SHOULD be included.

If a previous KEK is still valid for sending, then an update message is sent encrypted in the old KEK. This message MUST include the new KEK. This message SHOULD include the protocol master keys.

7.2. Out of Sequence Space

7.3. Changing the Active GCKS

8. Interface to Routing Protocol

This section describes signaling between MRKMP and the routing protocol. The primary communication between these protocols is that MRKMP populates rows in the key table making protocol master keys available to the routing protocol. However additional signaling is also required from the routing protocol to MRKMP. This section discusses that signaling. All required communication from MRKMP to the routing protocol can be accomplished by manipulating the key table. However an implementation MAY wish to signal MRKMP failures to the routing protocol in order to provide consistent management feedback.

8.1. Joining a Group

When a routing protocol instance wishes to begin communicating on a multicast group, it signals a group join event to MRKMP. This event includes the identity of the group as well as this router's priority for being a GCKS for the group. When MRKMP receives this event, it starts MRKMP for this group and attempts to find a GCKS.

8.2. Priority Adjustment

It is desirable that the GCKS function track the functions within a routing protocol. For example for protocols such as OSPF that designate a router on a link to manage adjacencies for that link, it would be desirable for the GCKS role to be assigned to that router. The routing protocol provides a priority input to the GCKS election process. Initially the routing protocol should map any priority mechanism within the routing protocol to the GCKS election procedure so that routers favored as announcer for a link will also be favored as a GCKS.

However, the routing protocol SHOULD also dynamically manipulate the GCKS election priority based on what happens within the routing protocol. The router actually elected as the announcer SHOULD have a GCKS election priority higher than any other group member. Typically, by the time the routing protocol is able to elect an announcer, a GCKS will already be chosen. However, if a GCKS election is triggered when the routing protocol is already operational, then the election can choose the routing protocol's announcer.

8.3. Leaving a Group

If a routing protocol terminates on an interface, MRKMP needs to be notified that group is no longer joined. MRKMP MUST stop participating in the GCKS election process, stop monitoring for key

management messages and if the current router is a GCKS, stop acting in that role.

9. Security Considerations

An attacker who can suppress packets sent to the group can create a denial of service condition. One attack is to suppress GCKS election packets and cause two routers to believe they are both the GCKS for the group. If the least preferred router never hears the GCKS advertisement from the more preferred router, then the group will remain partitioned. Such an attacker is likely to be able to mount more direct denial of service, for example suppressing the actual routing protocol packets.

The security of the system as a whole depends on the pair-wise security between the router currently in the GCKS role and the other routers in the group. Since any router can potentially act as GCKS, the pair-wise security between all members of the group is critical to the security of the system. In practical deployments, information used by the router acting as GCKS to authorize a member joining the group will be configured by some management application. In these deployments, the security of the system depends on the management application correctly maintaining this information on all routers potentially in the group.

10. Acknowledgements

This draft is the result of a design discussion held after the IETF 78 KARMP meeting. The authors, David Mcgrew, Brian Weis and Gregory Lebovitz all contributed to the design meeting.

11. Informative References

[RFC2627] Wallner, D., Harder, E., and R. Agee, "Key Management for Multicast: Issues and Architectures", RFC 2627, June 1999.

[RFC3547] Baugher, M., Weis, B., Hardjono, T., and H. Harney, "The Group Domain of Interpretation", RFC 3547, July 2003.

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

Authors' Addresses

Sam Hartman
Painless Security

Email: hartmans-ietf@mit.edu

Dacheng Zhang
Huawei

Email: zhangdacheng@huawei.com

MSEC Working Group
Internet-Draft
Intended status: Standards Track
Expires: April 28, 2011

B. Weis
S. Rowles
Cisco Systems
T. Hardjono
MIT

October 25, 2010

The Group Domain of Interpretation
draft-ietf-msec-gdoi-update-07

Abstract

This document describes an updated version of the Group Domain of Interpretation (GDOI) protocol specified in RFC 3547. The GDOI provides group key management to support secure group communications according to the architecture specified in RFC 4046. The GDOI manages group security associations, which are used by IPsec and potentially other data security protocols.

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 http://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 April 28, 2011.

Table of Contents

1. Introduction

Secure group and multicast applications require a method by which each group member shares common security policy and keying material. This document describes the Group Domain of Interpretation (GDOI), which is an ISAMKP [RFC2408] Domain of Interpretation (DOI), a group key management system. The GDOI distributes security associations (SAs) for IPsec AH [RFC4302] and ESP [RFC4303] protocols and potentially other data security protocols used in group applications. The GDOI uses the group key management model defined in [RFC4046], and described more generally by the The Multicast Group Security Architecture [RFC3740].

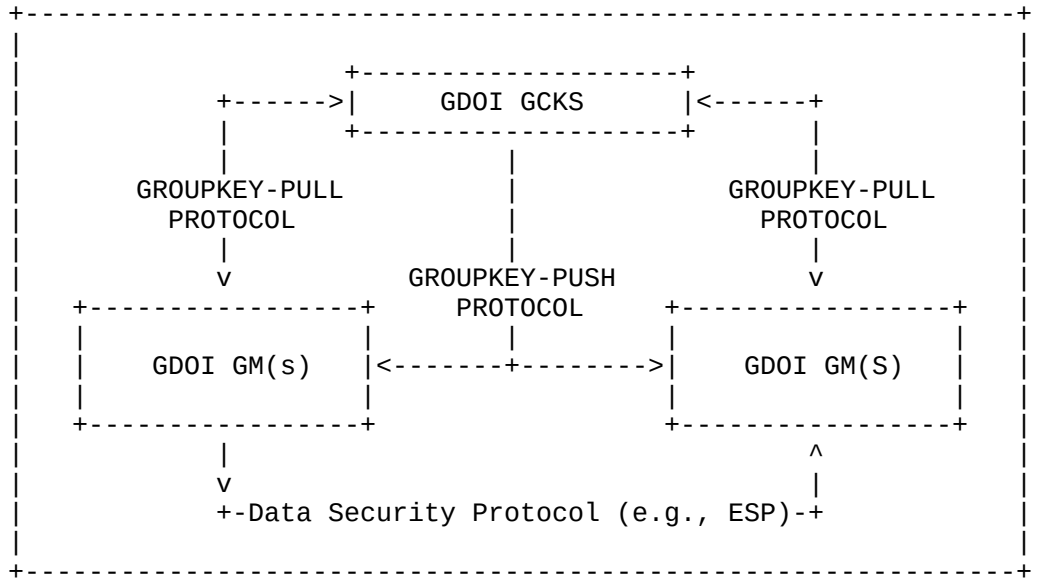
In this group key management model, the GDOI protocol participants are a "group controller/key server" (GCKS) and a group member (GM). A group member contacts ("registers with") a GCKS to join the group. During the registration mutual authentication and authorization are achieved, after which the GCKS distributes current group policy and keying material to the group member over an authenticated and encrypted session. The GCKS may also initiate contact ("rekeys") group members to provide updates to group policy.

ISAKMP defines two "phases" of negotiation (p.16 of [RFC2408]). A Phase 1 security association provides mutual authentication and authorization, and a security association that is used by the protocol participants to execute a phase 2 exchange. This document incorporates (i.e., uses but does not re-define) the Phase 1 security association definition from the Internet DOI [RFC2407], [RFC2409]. Phase 1 security association types other than ISAKMP are possible, and are noted in Appendix A. Requirements of those phase 1 security associations are specified in Section 2. The GDOI includes two new phase 2 ISAKMP exchanges (protocols), as well as necessary new payload definitions to the ISAKMP standard (p. 14 of [RFC2408]). These two new protocols are:

1. The GROUPKEY-PULL registration protocol exchange. This exchange uses "pull" behavior since the member initiates the retrieval of these SAs from a GCKS. It is protected by an ISAKMP phase 1 protocol, as described above. At the culmination of a GROUPKEY-PULL exchange, an authorized group member has received and installed a set of SAs that represent group policy, and it is ready to participate in secure group communications.

2. The GROUPKEY-PUSH rekey protocol exchange. The rekey protocol is a datagram initiated ("pushed") by the GCKS, usually delivered to group members using a IP multicast address. The rekey protool is an ISAKMP protocol, where cryptographic policy and keying material ("Re-key SA") is included in the group policy

distributed by the GCKS in the GROUPKEY-PULL exchange. At the culmination of a GROUPKEY-PUSH exchange the key server has sent group policy to all authorized group members, allowing receiving group members to participate in secure group communications. If a group management method is included in group policy (as described in Section 1.5.1), at the conclusion of the GROUPKEY-PUSH exchange some members of the group may have been de-authorized and no longer able to participate in the secure group communications.



Although the GROUPKEY-PUSH protocol specified by this document can be used to refresh the Re-key SA protecting the GROUPKEY-PUSH protocol, the most common use of GROUPKEY-PUSH is to establish keying material and policy for a data security protocol.

In summary, GDOI is a group security association management protocol: All GDOI messages are used to create, maintain, or delete security associations for a group. As described above, these security associations protect one or more data security protocal SAs, a Re-key SA, and/or other data shared by group members for multicast and groups security applications.

1.1. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

1.2. Terminology

The following key terms are used throughout this document.

Data-Security SA. The security policy distributed by a GDOI GCKS describing traffic that is expected to be protected by group members. This document described the distribution of IPsec AH and ESP Data-Security SAs.

GCKS. A Group Controller/Key Server [RFC3740] that defines group policy and distributes keys for that policy.

Group Member. An authorized member of a secure group, sending and/or receiving IP packets related to the group.

GROUPKEY-PULL. A protocol used by a GDOI Group Member to requst group policy and keying material.

GROUPKEY-PUSH. A protocol used by a GDOI GCKS to distribute updates of group policy and keying material to authorized group members.

Key Encrypting Key (KEK). The symmetric cipher key used to protect the GROUPKEY-PUSH message.

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

Re-key SA. The security policy protecting a GROUPKEY-PUSH protocol.

Traffic Encryption Key (TEK). The symmetric cipher key used to protect a data security protocol (e.g., IPsec ESP).

1.3. GDOI Applications

GDOI can be used to distribute keys for several secure multicast applications, where different appliacations have different key management requirements. This section outlines two example ways that GDOI can be used. Other examples can be found in Section 10 of [HD03].

A simple application is secure delivery of periodic multicast content over an organization’s IP network, perhaps a multicast video broadcast. Assuming the content delivery timeframe is bounded and the group membership is not expected to change over time, there is no need for group policy to include a GROUPKEY-PUSH exchange, and there’s no need for the GCKS to distribute a Re-key SA. Thus, the GDOI GCKS may only need to distribute a single set of Data-Security SAs to protect the time-bounded broadcast.

In contrast, a persistent IP multicast application (e.g., stock-ticker delivery service) may have many group members, where the group membership changes over time. A periodic change of Data-security SAs may be desirable, and the potential for change in group membership requires the use of a group management method enabling de-authorization of group members. The GDOI GCKS will distribute the current set of Data-Security SAs and a Re-key SA to registering group members. It will then deliver regularly-scheduled GROUPKEY-PUSH protocol delivering the new SAs for the group. Additionally, the group membership on the GCKS may be frequently adjusted, which will result in GROUPKEY-PUSH exchange delivering a new Rekey SAs protected by a group management method. Each GROUPKEY-PUSH may include Data-security SAs and/or a Rekey SA.

In each example the relevant policy is defined on the GCKS and relayed to group members using the GROUPKEY-PULL and/or GROUPKEY-PUSH protocols. Specific policy choices configured by the GCKS administrator depends on each application.

1.4. Extending GDOI

Not all secure multicast or multimedia applications can use IPsec ESP or AH. Many Real Time Transport Protocol applications, for example, require security above the IP layer to preserve RTP header compression efficiencies and transport-independence [RFC3550]. Alternatively, GDOI can distribute message authentication code (MAC) policy and keys for legacy applications that have defined their own security associations [I-D.weis-gdoi-mac-tek].

In order to add a new data security protocol, a new RFC MUST specify the data-security SA parameters conveyed by GDOI for that security protocol; these parameters are listed in Section 5.5.2 of this document.

Data security protocol SAs MUST protect group traffic. GDOI provides no restriction on whether that group traffic is transmitted as unicast or multicast packets.

1.5. Forward and Backward Access Control

Through GROUPKEY-PUSH, the GDOI supports group management methods such as LKH (section 5.4 of [RFC2627]) 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 GDOI, but are not specified as a part of this document.

1.5.1. Forward Access Control Requirements

When group membership is altered using a group management algorithm new Data-security SAs are usually also needed. New SAs 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 Data-security SAs MUST NOT be included in a GROUPKEY-PUSH message which changes group membership. This is required because the new Data-security SAs are not protected with the new KEK. Instead, two sequential GROUPKEY-PUSH messages must be sent by the GCKS; the first changing the KEK, and the second (protected with the new KEK) distributing the new Data-security SAs.

Note that in the above sequence, although the new KEK can effectively deny access to the group to some group members they will be able to view the new KEK policy. If forward access control policy for the group includes keeping the KEK policy secret as well as the KEK itself secret, then two GROUPKEY-PUSH messages changing the KEK must occur before the new Data-security SAs are transmitted.

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

2. GDOI Phase 1 protocol

The GDOI GROUPKEY-PULL exchange is a "phase 2" protocol which MUST be protected by a "phase 1" protocol. The "phase 1" protocol can be any protocol which provides for the following protections:

- Peer Authentication
- Confidentiality
- Message Integrity

The following sections describe one such "phase 1" protocol. Other protocols which may be potential "phase 1" protocols are described in Appendix A. However, the use of the protocols listed there are not considered part of this document.

2.1. ISAKMP Phase 1 protocol

This document defines how the ISAKMP phase 1 exchanges as defined in [RFC2409] can be used a "phase 1" protocol for GDOI. The following sections define characteristics of the ISAKMP phase 1 protocols that are unique for these exchanges when used for GDOI.

Section 7.1 describes how the ISAKMP Phase 1 protocols meet the requirements of a GDOI "phase 1" protocol.

2.1.1. DOI value

The Phase 1 SA payload has a DOI value. That value MUST be the GDOI DOI value as defined later in this document.

2.1.2. UDP port

IANA has assigned port 848 for the use of GDOI, which allows for an implementation to use separate ISAKMP implementations to service GDOI and IKEv1 [RFC2409]. A GCKS SHOULD listen on this port for GROUPKEY-PULL exchanges, and the GCKS MAY use this port to distribute GROUPKEY-PUSH messages. An ISAKMP phase 1 exchange implementation supporting NAT Traversal [RFC3947] may move to port 4500 to process the GROUPKEY-PULL exchange.

3. GROUPKEY-PULL Exchange

The goal of the GROUPKEY-PULL exchange is to establish a Re-key and/or Data-security SAs at the member for a particular group. A Phase 1 SA protects the GROUPKEY-PULL; there MAY be multiple GROUPKEY-PULL exchanges for a given Phase 1 SA. The GROUPKEY-PULL exchange downloads the data security keys (TEKs) and/or group key encrypting key (KEK) or KEK array under the protection of the Phase 1 SA.

3.1. Authorization

The Phase 1 identity SHOULD be used by a GCKS to authorize the Phase 2 (GROUPKEY-PULL) request for a group key. A group member MUST ensure that the Phase 1 identity of the GCKS is an authorized GCKS. When no authorization is performed, it is possible for a rogue GDOI participant to perpetrate a man-in-the-middle attack between a group member and a GCKS [MP04].

3.2. Messages

The GROUPKEY-PULL is a Phase 2 exchange. Phase 1 computes SKEYID_a which is the "key" in the keyed hash used in the GROUPKEY-PULL HASH payloads. When using the Phase 1 defined in this document, SKEYID_a is derived according to [RFC2409]. As with the IKEv1 HASH payload generation (Section 5.5 of [RFC2409], each GROUPKEY-PULL message hashes a uniquely defined set of values. Nonces permute the HASH and provide some protection against replay attacks. Replay protection is important to protect the GCKS from attacks that a key management server will attract.

The GROUPKEY-PULL uses nonces to guarantee "liveness", or against replay of a recent GROUPKEY-PULL message. The replay attack is only useful in the context of the current Phase 1. If a GROUPKEY-PULL message is replayed based on a previous Phase 1, the HASH calculation will fail due to a wrong SKEYID_a. The message will fail processing before the nonce is ever evaluated. In order for either peer to get the benefit of the replay protection, it must postpone as much processing as possible until it receives the message in the protocol that proves the peer is live. For example, the Responder MUST NOT adjust its internal state (e.g., keeping a record of the Initiator) until it receives a message with Nr included properly in the HASH payload.

Nonces require an additional message in the protocol exchange to ensure that the GCKS does not add a group member until it proves liveliness. The GROUPKEY-PULL member-initiator expects to find its nonce, Ni, in the HASH of a returned message. And the GROUPKEY-PULL

GCKS responder expects to see its nonce, Nr, in the HASH of a returned message before providing group-keying material as in the following exchange.

```
          Initiator (Member)                  Responder (GCKS)
          ------------------                  ----------------
          HDR*, HASH(1), Ni, ID     -->
                                    <--     HDR*, HASH(2), Nr, SA
          HDR*, HASH(3)             -->
                                    <--     HDR*, HASH(4), [SEQ,] KD
```

Hashes are computed as follows:

```
HASH(1) = prf(SKEYID_a, M-ID | Ni | ID)
HASH(2) = prf(SKEYID_a, M-ID | Ni_b | Nr | SA)
HASH(3) = prf(SKEYID_a, M-ID | Ni_b | Nr_b
HASH(4) = prf(SKEYID_a, M-ID | Ni_b | Nr_b [ | SEQ | ] KD)
```

* Protected by the Phase 1 SA, encryption occurs after HDR

HDR is an ISAKMP header payload that uses the Phase 1 cookies and a message identifier (M-ID) as in IKEv1. Note that nonces are included in the first two messages, with the GCKS returning only the SA policy payload before liveliness is proven. The HASH payloads [RFC2409] prove that the peer has the Phase 1 secret (SKEYID_a) and the nonce for the exchange identified by message id, M-ID. Once liveliness is established, the last message completes the real processing of downloading the KD payload.

In addition to the Nonce and HASH payloads, the member-initiator identifies the group it wishes to join through the ISAKMP ID payload. The GCKS responder informs the member of the current value of the sequence number in the SEQ payload; the sequence number provides anti-replay state associated with a KEK. The SEQ payload has no other use, and is omitted from the GROUPKEY_PULL exchange when a KEK attribute is not included in the SA payload.

When a SEQ payload is included in the GROUPKEY-PULL exchange, it includes the most recently used sequence number for the group. At the conclusion of a GROUPKEY-PULL exchange, the initiating group member MUST NOT accept any rekey message with both the KEK attribute SPI value and a sequence number less than or equal to the one received during the GROUPKEY-PULL. When the first group member initiates a GROUPKEY-PULL exchange, the GCKS provides a Sequence Number of zero, since no GROUPKEY-PUSH messages have yet been sent. Note the sequence number increments only with GROUPKEY-PUSH messages.

The GROUPKEY-PULL exchange distributes the current sequence number to the group member.

The sequence number resets to a value of one with the usage of a new KEK attribute. Thus the first packet sent for a given Rekey SA will have a Sequence Number of 1. The sequence number increments with each successive rekey.

The GCKS responder informs the member of the cryptographic policies of the group in the SA payload, which describes the DOI, KEK and/or TEK keying material, and authentication transforms. The SPIs are also determined by the GCKS and downloaded in the SA payload chain (see Section 5.2). The SA KEK attribute contains the ISAKMP cookie pair for the Re-key SA, which is not negotiated but downloaded. The SA TEK attribute contains a SPI as defined in Section 5.5 of this document. The second message downloads this SA payload. If a Re-key SA is defined in the SA payload, then KD will contain the KEK; if one or more Data-security SAs are defined in the SA payload, KD will contain the TEKs. This is useful if there is an initial set of TEKs for the particular group and can obviate the need for future TEK GROUPKEY-PUSH messages (described in section 4).

3.2.1. ISAKMP Header Initialization

Cookies are used in the ISAKMP header to identify a particular GDOI session. The GDOI GROUPKEY-PULL exchange uses cookies according to ISAKMP [RFC2408].

Next Payload identifies an ISAKMP or GDOI payload (see Section 5.0).

Major Version is 1 and Minor Version is 0 according to ISAKMP (Section 3.1 of [RFC2408]).

The Exchange Type has value 32 for the GDOI GROUPKEY-PULL exchange.

Flags, Message ID, and Length are according to ISAKMP (Section 3.1 of [RFC2408]).

3.3. Initiator Operations

Before a group member (GDOI initiator) contacts the GCKS, it must determine the group identifier and acceptable Phase 1 policy via an out-of-band method. Phase 1 is initiated using the GDOI DOI in the SA payload. Once Phase 1 is complete, the initiator state machine moves to the GDOI protocol.

To construct the first GDOI message the initiator chooses Ni and creates a nonce payload, builds an identity payload including the

group identifier, and generates HASH(1).

Upon receipt of the second GDOI message, the initiator validates HASH(2), extracts the nonce Nr, and interprets the SA payload. The SA payload contains policy describing the security protocol and cryptographic protocols used by the group. This policy describes the Re-key SA (if present), Data-security SAs, and other group policy. If the policy in the SA payload is acceptable to the initiator, it continues the protocol.

The initiator constructs the third GDOI message by creating HASH(3).

Upon receipt of the fourth GDOI message, the initiator validates HASH(4).

If SEQ payload is present, the sequence number in the SEQ payload must be checked against any previously received sequence number for this group. If it is less than the previously received number, it should be considered stale and ignored. This could happen if two GROUPKEY-PULL messages happened in parallel, and the sequence number changed between the times the results of two GROUPKEY-PULL messages were returned from the GCKS.

The initiator interprets the KD key packets, where each key packet includes the keying material for SAs distributed in the SA payload. Keying material is matched by comparing the SPIs in the key packets to SPIs previously sent in the SA payloads. Once TEK keys and policy are matched, the initiator provides them to the data security subsystem, and it is ready to send or receive packets matching the TEK policy. If this group has a KEK, the KEK policy and keys are marked as ready for use, and the initiator knows to expect the sequence number reset to 1 with the next Rekey SA, which will be encrypted with the new KEK attribute. The initiator is now ready to receive GROUPKEY-PUSH messages.

If the KD payload included an LKH array of keys, the initiator takes the last key in the array as the group KEK. The array is then stored without further processing.

3.4. Receiver Operations

The GCKS (responder) passively listens for incoming requests from group members. The Phase 1 authenticates the group member and sets up the secure session with them.

Upon receipt of the first GDOI message the GCKS validates HASH(1), extracts the Ni and group identifier in the ID payload. It verifies that its database contains the group information for the group

identifier.

The GCKS constructs the second GDOI message, including a nonce Nr, and the policy for the group in an SA payload, followed by SA GAP, SA KEK, and/or SA TEK payloads according to the GCKS policy. (See Section 5.2.1 for details on how the GCKS chooses which payloads to send.)

Upon receipt of the third GDOI message the GCKS validates HASH(3).

The GCKS constructs the fourth GDOI message, including the SEQ payload (if the GCKS sends rekey messages), and the KD payload containing keys corresponding to policy previously sent in the SA TEK and SA KEK payloads. If a group management algorithm is defined as part of group policy, the GCKS will first insert the group member into the group management structure (e.g., a leaf in the LKH tree), and then create an LKH array of keys and include it in the KD payload. The first key in the array is associated with the group member leaf node, followed by each LKH node above it in the tree, culminating with the root node (which is also the KEK).

4. GROUPKEY-PUSH Message

GDOI sends control information securely using group communications. Typically this will be using IP multicast distribution of a GROUPKEY-PUSH message but it can also be "pushed" using unicast delivery if IP multicast is not possible. The GROUPKEY-PUSH message replaces a Re-key SA KEK or KEK array, and/or creates a new Data-security SA.

```
          Member                                GCKS or Delegate
          ------                                ----------------
                          <---- HDR*, SEQ, [D,] SA, KD, SIG
```

* Protected by the Re-key SA KEK; encryption occurs after HDR

HDR is defined below. The SEQ payload is defined in the Payloads section. One or more D (Delete) payloads (further described in Section 5.9) optionally specify the deletion of existing group policy. The SA defines the policy (e.g., protection suite) and attributes (e.g., SPI) for a replacement Re-key SA and/or Data-security SAs. KD is the key download payload as described in the Payloads section.

The SIG payload includes a signature of a hash of the entire GROUPKEY-PUSH message (excepting the SIG payload bytes) before it has been encrypted. The HASH is taken over the string 'rekey', the GROUPKEY-PUSH HDR, followed by all payloads preceding the SIG payload. The prefixed string ensures that the signature of the Rekey datagram cannot be used for any other purpose in the GDOI protocol. After the SIG payload is created using the signature of the above hash, with the receiver verifying the signature using a public key retrieved in policy received by a GROUPKEY-PULL exchange, or distributed in an earlier GROUPKEY-PUSH message. The current KEK encryption key (previously distributed in a GROUPKEY-PULL exchange or GROUPKEY-PUSH message) encrypts all the payloads following the GROUPKEY-PUSH HDR. Note: The rationale for this order of operations is given in Section 7.3.5.

If the SA defines the use of a single KEK or an LKH KEK array, KD MUST contain a corresponding KEK or KEK array for a new Re-key SA, which has a new cookie pair. When the KD payload carries a new SA KEK attribute (section 5.3), a Re-key SA is replaced with a new SA having the same group identifier (ID specified in message 1 of section 3.2) and incrementing the same sequence counter, which is initialized in message 4 of section 3.2. Note the first packet for the given Rekey SA encrypted with the new KEK attribute will have a Sequence number of 1. If the SA defines an SA TEK payload, this informs the member that a new Data-security SA has been created, with keying material carried in KD (Section 5.6).

If the SA defines a large LKH KEK array (e.g., during group initialization and batched rekeying), parts of the array MAY be sent in different unique GROUPKEY-PUSH datagrams. However, each of the GROUPKEY-PUSH datagrams MUST be a fully formed GROUPKEY-PUSH datagram. This results in each datagram containing a sequence number and the policy in the SA payload, which corresponds to the KEK array portion sent in the KD payload.

4.1. Use of signature keys

A signing key should not be used in more than one context (e.g., used for host authentication and also for message authentication). Thus, the GCKS SHOULD NOT use the same key to sign the SIG payload in the GROUPKEY-PUSH message as was used for authentication in the GROUPKEY-PULL exchange.

4.2. ISAKMP Header Initialization

Unlike ISAKMP or IKEv1, the cookie pair is completely determined by the GCKS. The cookie pair in the GDOI ISAKMP header identifies the Re- key SA to differentiate the secure groups managed by a GCKS. Thus, GDOI uses the cookie fields as an SPI.

Next Payload identifies an ISAKMP or GDOI payload (see Section 5.0).

Major Version is 1 and Minor Version is 0 according to ISAKMP (Section 3.1 of [RFC2408]).

The Exchange Type has value 33 for the GDOI GROUPKEY-PUSH message.

Flags MUST have the Encryption bit set according to [RFC2008, Section 3.1]. All other bits MUST be set to zero.

Message ID MUST be set to zero.

Length is according to ISAKMP (Section 3.1 of [RFC2408]).

4.3. GCKS Operations

GCKS or its delegate may initiate a Rekey message for one of several reasons, e.g., the group membership has changed or keys are due to expire.

To begin the rekey datagram the GCKS builds an ISAKMP HDR with the correct cookie pair, and a SEQ payload that includes a sequence number which is one greater than the previous rekey datagram. If the message is using the new KEK attribute for the first time, the SEQ is reset to 1 in this message.

An SA payload is then added. This is identical in structure and meaning to a SA payload sent in a GROUPKEY-PULL exchange. If there are changes to the KEK (including due to group members being excluded, in the case of LKH), an SA_KEK attribute is added to the SA. If there are one or more new TEKs then SA_TEK attributes are added to describe that policy.

A KD payload is then added. This is identical in structure and meaning to a KD payload sent in a GROUPKEY-PULL exchange. If an SA_KEK attribute was included in the SA payload then corresponding KEK keys (or a KEK update array) is included. A KEK update array is created by first determining which group members have been excluded, and then generating new keys as necessary and distribute LKH update arrays sufficient to provide the new KEK to remaining group members (see Section 5.4.1 of [RFC2627] for details). TEK keys are also sent for each SA_TEK attribute included in the SA payload.

In the penultimate step, the initiator hashes the string "rekey" followed by the key management message already formed. The hash is signed, placed in a SIG payload and added to the datagram.

Lastly, the payloads following the HDR are encrypted using the current KEK encryption key. The datagram can now be sent.

4.4. Group Member Operations

A group member receiving the GROUPKEY-PUSH datagram matches the cookie pair in the ISAKMP HDR to an existing SA. The message is decrypted, and the form of the datagram is validated. This weeds out obvious ill-formed messages (which may be sent as part of a Denial of Service attack on the group).

The sequence number in the SEQ payload is validated to ensure that it is greater than the previously received sequence number, and that it fits within a window of acceptable values. The SIG payload is then validated. If the signature fails, the message is discarded.

The SA and KD payloads are processed which results in a new GDOI Rekey SA (if the SA payload included an SA_KEK attribute) and/or new IPsec SAs being added to the system. If the KD payload includes an LKH update array, the group member compares the LKH ID in each key update packet to the LKH IDs that it holds. If it finds a match, it decrypts the key using the key prior to it in the key array and stores the new key in the LKH key array that it holds. The final decryption yields the new group KEK.

5. Payloads and Defined Values

This document specifies use of several ISAKMP payloads, which are defined in accordance with RFC 2408. The following payloads are extended or further specified.

Next Payload Type	Value
Security Association (SA)	1
Identification (ID)	5
Nonce (N)	10

Several payload formats specific to the group security exchanges are required.

Next Payload Type	Value
SA KEK Payload (SAK)	15
SA TEK Payload (SAT)	16
Key Download (KD)	17
Sequence Number (SEQ)	18
SA Group Associated Policy (GAP)	TBD-1

5.1. Identification Payload

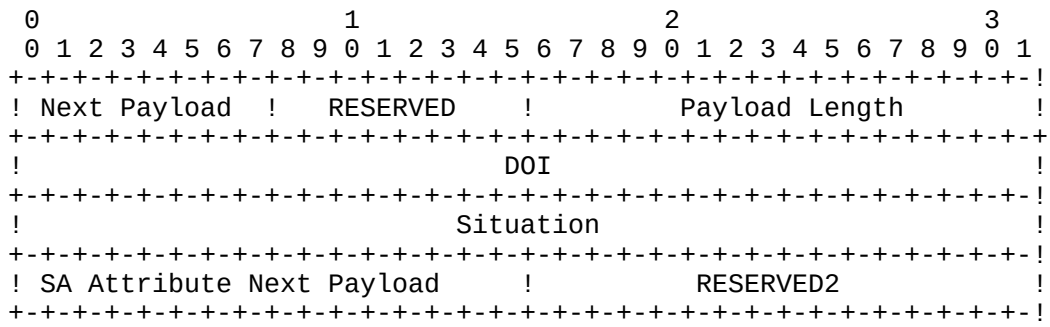
The Identification Payload is defined in RFC 2408. For the GDOI, it is used to identify a group identity that will later be associated with Security Associations for the group. A group identity may map to a specific IP multicast group, or may specify a more general identifier, such as one that represents a set of related multicast streams.

When used with the GDOI, the DOI Specific ID Data field MUST be set to 0.

When used with the GDOI, the ID_KEY_ID ID Type MUST be supported by a conforming implementation, and MUST specify a four (4)-octet group identifier as its value. Implementations MAY also support other ID Types.

5.2. Security Association Payload

The Security Association payload is defined in RFC 2408. For the GDOI, it is used by the GCKS to assert security attributes for both Re-key and Data-security SAs.



The Security Association Payload fields are defined as follows:

o Next Payload (1 octet) -- Identifies the next payload for the GROUPKEY-PULL or the GROUPKEY-PUSH message as defined above. The next payload MUST NOT be a SAK Payload or SAT Payload type, but the next non-Security Association type payload.

o RESERVED (1 octet) -- Must be zero.

o Payload Length (2 octets) -- Is the octet length of the current payload including the generic header and all TEK and KEK payloads.

o DOI (4 octets) -- Is the GDOI, which is value 2.

o Situation (4 octets) -- Must be zero.

o SA Attribute Next Payload (1 octet) -- Must be either a SAK Payload or a SAT Payload. See section 5.2.1 for a description of which circumstances are required for each payload type to be present.

o RESERVED (2 octets) -- Must be zero.

5.2.1. Payloads following the SA payload

Payloads that define specific security association attributes for the KEK and/or TEKs used by the group MUST follow the SA payload. How many of each payload is dependent upon the group policy. There may be zero or one SAK Payloads, zero or more GAP Payloads, and zero or more SAT Payloads, where either one SAK or SAT payload MUST be present. When present, the order of the SA Attributes payloads must be: KEK, GAP, and TEKs.

This latitude allows various group policies to be accommodated. For example if the group policy does not require the use of a Re-key SA, the GCKS would not need to send an SA KEK attribute to the group member since all SA updates would be performed using the Registration

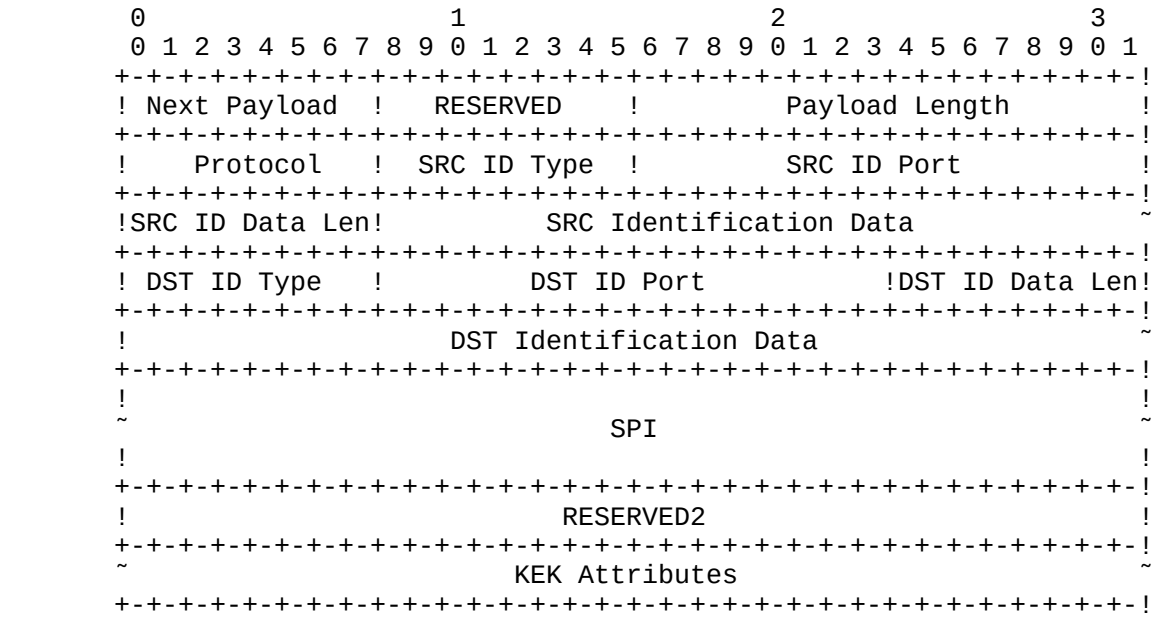
SA. Alternatively, group policy might use a Re-key SA but choose to download a KEK to the group member only as part of the Registration SA. Therefore, the KEK policy (in the SA KEK attribute) would not be necessary as part of the Re-key SA message SA payload.

Specifying multiple SATs allows multiple sessions to be part of the same group and multiple streams to be associated with a session (e.g., video, audio, and text) but each with individual security association policy.

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

5.3. SA KEK payload

The SA KEK (SAK) payload contains security attributes for the KEK method for a group and parameters specific to the GROUPKEY-PULL operation. The source and destination identities describe the identities used for the GROUPKEY-PULL datagram.



The SAK Payload fields are defined as follows:

o Next Payload (1 octet) -- Identifies the next payload for the GROUPKEY-PULL or the GROUPKEY-PUSH message. The only valid next payload types for this message are a GAP Payload, SAT Payload or zero

to indicate that no SA Attribute payloads follow.

o RESERVED (1 octet) -- Must be zero.

o Payload Length (2 octets) -- Length of this payload, including the KEK attributes.

o Protocol (1 octet) -- Value describing an IP protocol ID (e.g., UDP/TCP) for the rekey datagram.

o SRC ID Type (1 octet) -- Value describing the identity information found in the SRC Identification Data field. Defined values are specified by the IPsec Identification Type section in the IANA isakmpd-registry [ISAKMP-REG].

o SRC ID Port (2 octets) -- Value specifying a port associated with the source Id. A value of zero means that the SRC ID Port field should be ignored.

o SRC ID Data Len (1 octet) -- Value specifying the length of the SRC Identification Data field.

o SRC Identification Data (variable length) -- Value, as indicated by the SRC ID Type.

o DST ID Type (1 octet) -- Value describing the identity information found in the DST Identification Data field. Defined values are specified by the IPsec Identification Type section in the IANA isakmpd-registry [ISAKMP-REG].

o DST ID Prot (1 octet) -- Value describing an IP protocol ID (e.g., UDP/TCP).

o DST ID Port (2 octets) -- Value specifying a port associated with the source Id.

o DST ID Data Len (1 octet) -- Value specifying the length of the DST Identification Data field.

o DST Identification Data (variable length) -- Value, as indicated by the DST ID Type.

o SPI (16 octets) -- Security Parameter Index for the KEK. The SPI must be the ISAKMP Header cookie pair where the first 8 octets become the "Initiator Cookie" field of the GROUPKEY-PUSH message ISAKMP HDR, and the second 8 octets become the "Responder Cookie" in the same HDR. As described above, these cookies are assigned by the GCKS.

o RESERVED2 (4 octets) -- Must be zero.

o KEK Attributes -- Contains KEK policy attributes associated with the group. The following sections describe the possible attributes. Any or all attributes may be optional, depending on the group policy.

5.3.1. KEK Attributes

The following attributes may be present in a SAK Payload. The attributes must follow the format defined in ISAKMP (Section 3.3 of [RFC2408]). 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).

ID Class	Value	Type
RESERVED	0	
KEK_MANAGEMENT_ALGORITHM	1	B
KEK_ALGORITHM	2	B
KEK_KEY_LENGTH	3	B
KEK_KEY_LIFETIME	4	V
SIG_HASH_ALGORITHM	5	B
SIG_ALGORITHM	6	B
SIG_KEY_LENGTH	7	B

The KEK_MANAGEMENT_ALGORITHM attribute may only be included in a GROUPKEY-PULL message.

5.3.2. KEK_MANAGEMENT_ALGORITHM

The KEK_MANAGEMENT_ALGORITHM class specifies the group KEK management algorithm used to provide forward or backward access control (i.e., used to exclude group members). Defined values are specified in the following table.

KEK Management Type	Value
RESERVED	0
LKH	1
RESERVED	2-127
Private Use	128-255

5.3.2.1. LKH

This type indicates the group management method described in Section 5.4 of [RFC2627]. A general discussion of LKH operations can also be found in Section 6.3 of Multicast and Group Security [HD03]

5.3.3. KEK_ALGORITHM

The KEK_ALGORITHM class specifies the encryption algorithm using with the KEK. Defined values are specified in the following table. An GDOI implementation MUST abort if it encounters an attribute or capability that it does not understand.

Algorithm Type	Value
RESERVED	0
KEK_ALG_DES	1
KEK_ALG_3DES	2
KEK_ALG_AES	3
RESERVED	4-127
Private Use	128-255

If a KEK_MANAGEMENT_ALGORITHM is defined which defines multiple keys (e.g., LKH), and if the management algorithm does not specify the algorithm for those keys, then the algorithm defined by the KEK_ALGORITHM attribute MUST be used for all keys which are included as part of the management.

5.3.3.1. KEK_ALG_DES

This type specifies DES using the Cipher Block Chaining (CBC) mode as described in [FIPS81].

5.3.3.2. KEK_ALG_3DES

This type specifies 3DES using three independent keys as described in "Keying Option 1" in [FIPS46-3].

5.3.3.3. KEK_ALG_AES

This type specifies AES as described in [FIPS197]. The mode of operation for AES is Cipher Block Chaining (CBC) as recommended in [SP.800-38A].

5.3.4. KEK_KEY_LENGTH

The KEK_KEY_LENGTH class specifies the KEK Algorithm key length (in bits). The Group Controller/Key Server (GCKS) adds the KEK_KEY_LEN attribute to the SA payload when distributing KEK policy to group members. The group member verifies whether or not it has the capability of using a cipher key of that size. If the cipher definition includes a fixed key length (e.g., KEK_ALG_3DES), the group member can make its decision solely using the KEK_ALGORITHM attribute and does not need the KEK_KEY_LEN attribute. Sending the KEK_KEY_LEN attribute in the SA payload is OPTIONAL if the KEK cipher has a fixed key length. Also, note that the KEK_KEY_LEN includes only the actual length of the cipher key (the IV length is not included in this attribute).

5.3.5. KEK_KEY_LIFETIME

The KEK_KEY_LIFETIME class specifies the maximum time for which the KEK is valid. The GCKS may refresh the KEK at any time before the end of the valid period. The value is a four (4) octet number defining a valid time period in seconds.

5.3.6. SIG_HASH_ALGORITHM

SIG_HASH_ALGORITHM specifies the SIG payload hash algorithm. The following table defines the algorithms for SIG_HASH_ALGORITHM.

Algorithm Type	Value
RESERVED	0
SIG_HASH_MD5	1
SIG_HASH_SHA1	2
SIG_HASH_SHA256	TBD-2
SIG_HASH_SHA384	TBD-3
SIG_HASH_SHA512	TBD-4
RESERVED	6-127
Private Use	128-255

The SHA hash algorithms are defined in the Secure Hash Standard[FIPS.180-2.2002].

If the SIG_ALGORITHM is SIG_ALG_ECDSA-256, SIG_ALG_ECDSA-384, or SIG_ALG_ECDSA-521 the hash algorithm is implicit in the definition, and SIG_HASH_ALGORITHM is not required to be present in a SAK Payload.

5.3.7. SIG_ALGORITHM

The SIG_ALGORITHM class specifies the SIG payload signature algorithm. Defined values are specified in the following table.

Algorithm Type	Value
RESERVED	0
SIG_ALG_RSA	1
SIG_ALG_DSS	2
SIG_ALG_ECDSS	3
SIG_ALG_RSA_PSS	TBD-6
SIG_ALG_ECDSA-256	TBD-7
SIG_ALG_ECDSA-384	TBD-8
SIG_ALG_ECDSA-521	TBD-9
RESERVED	8-127
Private Use	128-255

5.3.7.1. SIG_ALG_RSA

This algorithm specifies the RSA digital signature algorithm using the EMSA-PKCS1-v1_5 encoding method, as described in [RFC3447].

5.3.7.2. SIG_ALG_DSS

This algorithm specifies the DSS digital signature algorithm as described in Section 4 of [FIPS186-3].

5.3.7.3. SIG_ALG_ECDSS

This algorithm specifies the Elliptic Curve digital signature algorithm as described in Section 5 of [FIPS186-3]. This definition is deprecated in favor of the SIG_ALG_ECDSA family of algorithms.

5.3.7.4. SIG_ALG_RSA_PSS

This algorithm specifies the RSA digital signature algorithm using the EMSA-PSS encoding method, as described in [RFC3447].

5.3.7.5. SIG_ALG_ECDSA-256

This algorithm specifies the 256-bit Random ECP Group, as described in [RFC5903]. The format of the signature in the SIG payload MUST be as specified in [RFC4754].

5.3.7.6. SIG_ALG_ECDSA-384

This algorithm specifies the 384-bit Random ECP Group, as described in [RFC5903]. The format of the signature in the SIG payload MUST be as specified in [RFC4754].

5.3.7.7. SIG_ALG_ECDSA-521

This algorithm specifies the 521-bit Random ECP Group, as described in [RFC5903]. The format of the signature in the SIG payload MUST be as specified in [RFC4754].

5.3.8. SIG_KEY_LENGTH

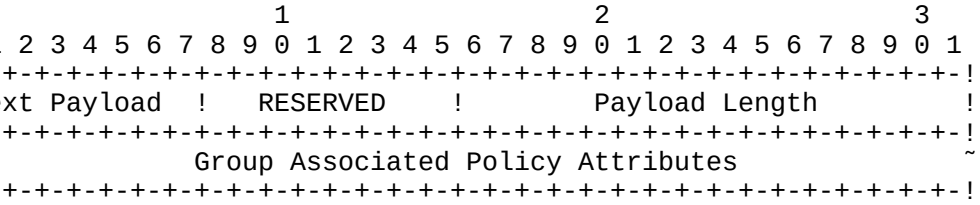
The SIG_KEY_LENGTH class specifies the length of the SIG payload key in bits.

5.4. Group Associated Policy

RFC 3547 provides for the distribution of policy in the GROUPKEY-PULL exchange in an SA payload. Policy can define GROUPKEY-PUSH policy (SA KEK) or traffic encryption policy (SA TEK) such as IPsec policy. There is a need to distribute group policy that fits into neither category. Some of this policy is generic to the group, and some is sender-specific policy for a particular group member.

GDOI distributes this associated group policy in a new payload called the SA Group Associated Policy (SA GAP). The SA GAP payload follows any SA KEK payload, and is placed before any SA TEK payloads. In the case that group policy does not include an SA KEK, the SA Attribute Next Payload field in the SA payload MAY indicate the SA GAP payload.

The SA GAP payload is defined as follows:



The SA GAP payload fields are defined as follows:

- Next Payload (1 octet) -- Identifies the next payload present in the GROUPKEY-PULL or the GROUPKEY-PUSH message. The only valid next payload type for this message is an SA TEK or zero to indicate there are no more security association attributes.

- o RESERVED (1 octet) -- Must be zero.

- o Payload Length (2 octets) -- Length of this payload, including the SA GAP header and Attributes.

- o Group Associated Policy Attributes (variable) -- Contains attributes following the format defined in Section 3.3 of RFC 2408.

Several group associated policy attributes are defined in this memo. An GDOI implementation MUST abort if it encounters and attribute or capability that it does not understand.

5.4.1. ACTIVATION_TIME_DELAY

This attribute allows a GCKS to set the Activation Time Delay for SAs generated from TEKs. The value is in seconds.

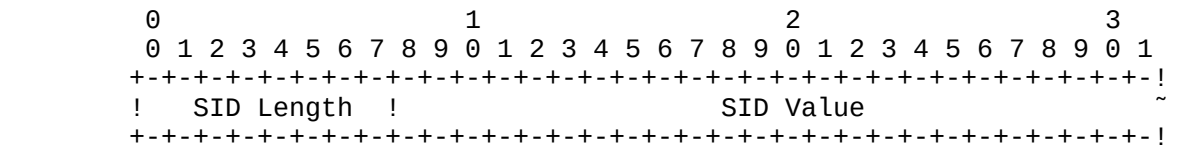
5.4.2. DEACTIVATION_TIME_DELAY

This attribute allows a GCKS to set the Deactivation Time Delay for SAs generated from TEKs. The value is in seconds. The values of Activation Time Delay and Deactivation Time Delay are independant. However, the Deactivation Time Delay vaue should be larger, which allows new SAs to be activated before older SAs are deactivated. Such a policy ensures that protected group traffic will always flow without interruption.

5.4.3. SENDER_ID

Several new AES counter-based modes of operation have been specified for ESP [RFC3686],[RFC4106],[RFC4309],[RFC4543] and AH [RFC4543]. These AES counter-based modes require that no two senders in the group ever send a packet with the same IV. This requirement is met using the method described in [I-D.ietf-msec-ipsec-group-counter-modes], which requires each sender to be allocated a unique Sender ID (SID). The SENDER_ID attribute is used to distribute an SID to a group member during the GROUPKEY-PULL message. Other algorithms with the same need may be defined in the future; the sender MUST use the IV construction method described above with those algorithms as well.

The SENDER_ID attribute value contains the following fields.



- SID Length (1 octet) -- A natural number defining the number of bits to be used in the SID field of the counter mode transform nonce. The length of the SID is chosen by the GCKS administrator based on the counter-based mode specification and the number of expected group members. In accordance with [I-D.ietf-msec-ipsec-group-counter-modes] an implementation MUST support values of 8, 12, and 16.

- SID Value (variable) -- The Sender ID value allocated to the group member.

5.4.3.1. GCKS Semantics

The GCKS maintains a SID counter (SIDC). It is incremented each time a SENDER_ID attribute is distributed to a group member. The first group member to register is given the SID of 1.

Any group member registering will be given a new SID value, which allows group members to act as a group sender when an older SID value becomes unusable (as described in the next section).

A GCKS MAY allocate multiple SID values in one SA GAP payload. Allocating several SID values at the same time to a group member expected to send at a high rate would obviate the need for the group member to re-register as frequently.

If a GCKS allocates all SID values, it can no longer respond to GDOI registrations and must re-initialize the entire group. This is done by including Delete Payload for all ESP and AH SAs in a GDOI rekey message, resetting the SIDC to zero, and creating new ESP and AH SAs that match the group policy. When group members re-register, the SIDs are allocated again beginning with the value 1 as described above. Each re-registering group member will be given a new SID and the new group policy.

The SENDER_ID attribute MUST NOT be sent as part of a GROUPKEY-PUSH message, because distributing the same sender-specific policy to more than one group member may reduce the security of the group.

5.4.3.2. Group Member Semantics

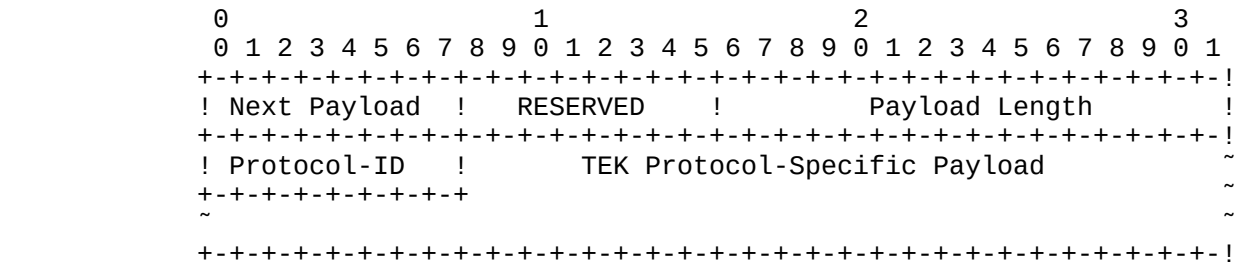
The SENDER_ID attribute value distributed to the group member MUST be used by that group member as the Sender Identifier (SID) field portion of the IV. The SID is used for all counter mode SAs distributed by the GCKS to be used for communications sent as a part of this group.

When the Sender-Specific IV (SSIV) field for any IPsec SA is exhausted, the group member MUST no longer act as a sender using its active SID. The group member SHOULD re-register, during which time the GCKS will issue a new SID to the group member. The new SID replaces the existing SID used by this group member, and also resets the SSIV value to its starting value. A group member MAY re-register prior to the actual exhaustion of the SSIV field to avoid dropping data packets due to the exhaustion of available SSIV values combined with a particular SID value.

A group member MUST NOT process a SENDER_ID attribute present in a GROUPKEY-PUSH message.

5.5. SA TEK Payload

The SA TEK (SAT) payload contains security attributes for a single TEK associated with a group.



The SAT Payload fields are defined as follows:

o Next Payload (1 octet) -- Identifies the next payload for the GROUPKEY-PULL or the GROUPKEY-PUSH message. The only valid next payload types for this message are another SAT Payload or zero to indicate there are no more security association attributes.

o RESERVED (1 octet) -- Must be zero.

o Payload Length (2 octets) -- Length of this payload, including the TEK Protocol-Specific Payload.

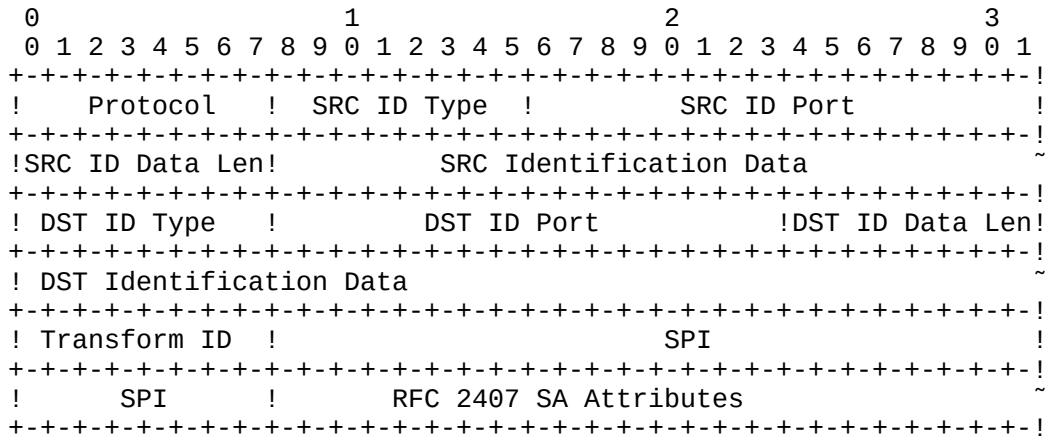
o Protocol-ID (1 octet) -- Value specifying the Security Protocol. The following table defines values for the Security Protocol

Protocol ID	Value
RESERVED	0
GDOI_PROTO_IPSEC_ESP	1
GDOI_PROTO_IPSEC_AH	TBD-5
RESERVED	3-127
Private Use	128-255

o TEK Protocol-Specific Payload (variable) -- Payload which describes the attributes specific for the Protocol-ID.

5.5.1. GDOI_PROTO_IPSEC_ESP/GDOI_PROTO_IPSEC_AH

The TEK Protocol-Specific payload for ESP and AH is as follows:



The SAT Payload fields are defined as follows:

o Protocol (1 octet) -- Value describing an IP protocol ID (e.g., UDP/TCP). A value of zero means that the Protocol field should be ignored.

o SRC ID Type (1 octet) -- Value describing the identity information found in the SRC Identification Data field. Defined values are specified by the IPsec Identification Type section in the IANA isakmpd-registry [ISAKMP-REG].

o SRC ID Port (2 octets) -- Value specifying a port associated with

the source Id. A value of zero means that the SRC ID Port field should be ignored.

o SRC ID Data Len (1 octet) -- Value specifying the length of the SRC Identification Data field.

o SRC Identification Data (variable length) -- Value, as indicated by the SRC ID Type. Set to three bytes of zero for multiple-source multicast groups that use a common TEK for all senders.

o DST ID Type (1 octet) -- Value describing the identity information found in the DST Identification Data field. Defined values are specified by the IPsec Identification Type section in the IANA isakmpd-registry [ISAKMP-REG].

o DST ID Prot (1 octet) -- Value describing an IP protocol ID (e.g., UDP/TCP). A value of zero means that the DST Id Prot field should be ignored.

o DST ID Port (2 octets) -- Value specifying a port associated with the source Id. A value of zero means that the DST ID Port field should be ignored.

o DST ID Data Len (1 octet) -- Value specifying the length of the DST Identification Data field.

o DST Identification Data (variable length) -- Value, as indicated by the DST ID Type.

o Transform ID (1 octet) -- Value specifying which ESP or AH transform is to be used. The list of valid values is defined in the IPsec ESP or IPsec AH Transform Identifiers section of the IANA isakmpd-registry [ISAKMP-REG].

o SPI (4 octets) -- Security Parameter Index for ESP.

o RFC 2407 Attributes -- ESP and AH Attributes from RFC 2407 Section 4.5. The GDOI supports all IPsec DOI SA Attributes for GDOI_PROTO_IPSEC_ESP and GDOI_PROTO_IPSEC_AH excluding the Group Description (section 4.5 of [RFC2407], which MUST NOT be sent by a GDOI implementation and is ignored by a GDOI implementation if received. The following attributes MUST be supported by an implementation supporting ESP and AH: SA Life Type, SA Life Duration, Encapsulation Mode. An implementation supporting ESP must also support the Authentication Algorithm attribute if the ESP transform includes authentication/ The Authentication Algorithm attribute of the IPsec DOI is group authentication in GDOI.

5.5.1.1. Harmonization with RFC 5374

The Multicast Extensions to the Security Architecture for the Internet Protocol (RFC 5374) introduces new requirements for a group key management system distributing IPsec policy. The following sections describe new GDOI requirements that result from harmonizing with that document.

5.5.1.1.1. Group Security Policy Database Attributes

RFC 5374 describes new attributes as part of the Group Security Policy Database (GSPD). These attributes describe policy that a group key management system must convey to a group member in order to support those extensions. The GDOI SA TEK payload distributes IPsec policy using IPsec security association attributes defined in [ISAKMP-REG]. This section defines how GDOI can convey the new attributes as IPsec Security Association Attributes.

5.5.1.1.2. Address Preservation

Applications use the extensions in RFC 5374 to copy the IP addresses into the outer IP header when encapsulating an IP packet as an IPsec tunnel mode packet. This allows an IP multicast packet to continue to be routed as a IP multiast packet. In order for the GDOI group member to appropriately set up the GSPD, the GCKS must provide that policy to the group member.

Depending on group policy, several address preservation methods are possible: no address preservation ("None"), preservation of the original source address ("Source-Only"), preservation of the original destination address ("Destination-Only"), or both addresses ("Source-And-Destination"). This memo adds the "Address Preservation" security association attribute. If this attribute is not included in a GDOI SA TEK payload provided by a GCKS, then Source-And-Destination address preservation has been defined for the SA TEK.

5.5.1.1.3. SA Direction

Depending on group policy, an IPsec SA created from an SA TEK payload may be required in one or both directions. SA TEK policy used by multiple senders is required to be installed in both the sending and receiving direction ("Symmetric"), whereas SA TEK for a single sender should only be installed in the receiving direction by receivers ("Receiver-Only") and in the sending direction by the sender ("Sender-Only"). This memo adds the "SA Direction" security association attribute. If the attribute is not included in a GDOI SA TEK payload, then the IPsec SA is treated as a Symmetric IPsec SA.

5.5.1.1.4. Re-key rollover

Section 4.2.1 of RFC 5374 specifies a key rollover method that requires two values be given it from the group key management protocol. The Activation Time Delay (ATD) attribute allows the GCKS to specify how long after the start of a re-key event that a group member is to activate new TEKs. The Deactivation Time Delay (DTD) attribute allows the GCKS to specify how long after the start of a re-key event that a group member is to deactivate existing TEKs.

This memo adds new attributes by which a GCKS can relay these values to group members as part of the Group Associated Policy described in Section 5.

5.5.2. Other Security Protocols

Besides ESP and AH, GDOI should serve to establish SAs for secure groups needed by other Security Protocols that operate at the transport, application, and internetwork layers. These other Security Protocols, however, are in the process of being developed or do not yet exist.

The following information needs to be provided for a Security Protocol to the GDOI.

- The Protocol-ID for the particular Security Protocol
- The SPI Size
- The method of SPI generation
- The transforms, attributes and keys needed by the Security Protocol

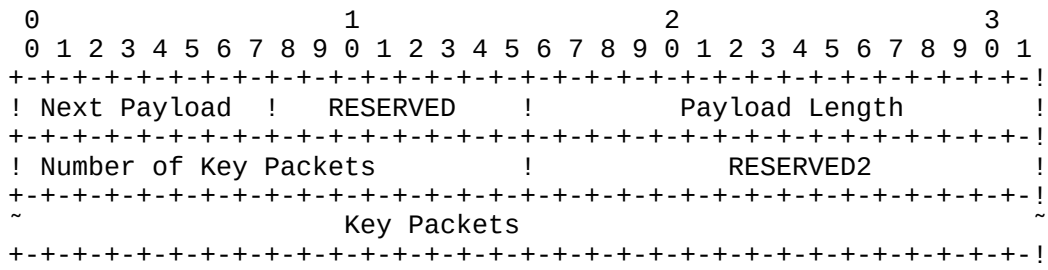
All Security Protocols must provide the information in the bulleted list above to guide the GDOI specification for that protocol. Definitions for the support of those Security Protocols in GDOI will be specified in separate documents.

A Security Protocol MAY protect traffic at any level of the network stack. However, in all cases applications of the Security Protocol MUST protect traffic which MAY be shared by more than two entities.

5.6. Key Download Payload

The Key Download Payload contains group keys for the group specified in the SA Payload. These key download payloads can have several security attributes applied to them based upon the security policy of

the group as defined by the associated SA Payload.



The Key Download Payload fields are defined as follows:

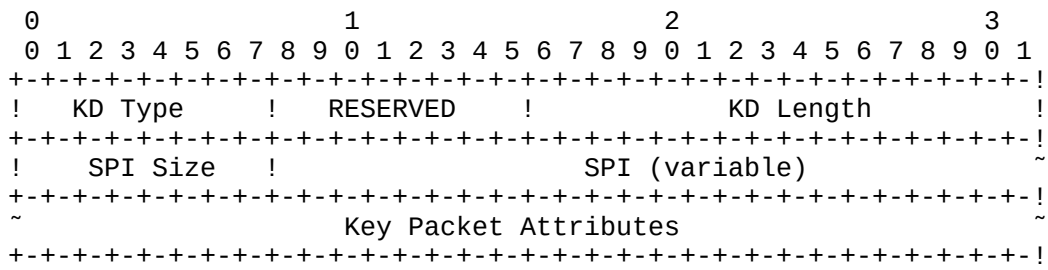
o Next Payload (1 octet) -- Identifier for the payload type of the next payload in the message. If the current payload is the last in the message, then this field will be zero.

o RESERVED (1 octet) -- Unused, set to zero.

o Payload Length (2 octets) -- Length in octets of the current payload, including the generic payload header.

o Number of Key Packets (2 octets) -- Contains the total number of both TEK and Rekey arrays being passed in this data block.

o Key Packets Several types of key packets are defined. Each Key Packet has the following format.



o Key Download (KD) Type (1 octet) -- Identifier for the Key Data field of this Key Packet.

Key Download Type	Value
RESERVED	0
TEK	1
KEK	2
LKH	3
RESERVED	4-127
Private Use	128-255

"KEK" is a single key whereas LKH is an array of key-encrypting keys.

o RESERVED (1 octet) -- Unused, set to zero.

o Key Download Length (2 octets) -- Length in octets of the Key Packet data, including the Key Packet header.

o SPI Size (1 octet) -- Value specifying the length in octets of the SPI as defined by the Protocol-Id.

o SPI (variable length) -- Security Parameter Index which matches a SPI previously sent in an SAK or SAT Payload.

o Key Packet Attributes (variable length) -- Contains Key information. The format of this field is specific to the value of the KD Type field. The following sections describe the format of each KD Type.

5.6.1. TEK Download Type

The following attributes may be present in a TEK Download Type. Exactly one attribute matching each type sent in the SAT payload MUST be present. The attributes must follow the format defined in ISAKMP (Section 3.3 of [RFC2408]). In the table, attributes defined as TV are marked as Basic (B); attributes defined as TLV are marked as Variable (V).

TEK Class	Value	Type
RESERVED	0	
TEK_ALGORITHM_KEY	1	V
TEK_INTEGRITY_KEY	2	V
TEK_SOURCE_AUTH_KEY	3	V

If no TEK key packets are included in a Registration KD payload, the group member can expect to receive the TEK as part of a Re-key SA. At least one TEK must be included in each Re-key KD payload. Multiple TEKs may be included if multiple streams associated with the

SA are to be rekeyed.

When an algorithm specification specifies the format of the keying material, the value transported in the KD payload for that key is passed according to that specification. The keying material may contain information besides a key. For example, The Use of Galois/ Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP) [RFC4106] defines a salt value as part of KEYMAT.

5.6.1.1. TEK_ALGORITHM_KEY

The TEK_ALGORITHM_KEY class declares that the encryption key for this SPI is contained as the Key Packet Attribute. The encryption algorithm that will use this key was specified in the SAT payload.

In the case that the algorithm requires multiple keys (e.g., 3DES), all keys will be included in one attribute.

DES keys will consist of 64 bits (the 56 key bits with parity bit). Triple DES keys will be specified as a single 192 bit attribute (including parity bits) in the order that the keys are to be used for encryption (e.g., DES_KEY1, DES_KEY2, DES_KEY3).

5.6.1.2. TEK_INTEGRITY_KEY

The TEK_INTEGRITY_KEY class declares that the integrity key for this SPI is contained as the Key Packet Attribute. The integrity algorithm that will use this key was specified in the SAT payload. Thus, GDOI assumes that both the symmetric encryption and integrity keys are pushed to the member. HMAC-SHA1 keys will consist of 160 bits[RFC2404], HMAC-MD5 keys will consist of 128 bits[RFC2403]. HMAC-SHA2 and AES-GMAC keys will have a key length equal to the output length of the hash functions [RFC4868][RFC4543].

5.6.1.3. TEK_SOURCE_AUTH_KEY

The TEK_SOURCE_AUTH_KEY class declares that the source authentication key for this SPI is contained in the Key Packet Attribute. The source authentication algorithm that will use this key was specified in the SAT payload.

5.6.2. KEK Download Type

The following attributes may be present in a KEK Download Type. Exactly one attribute matching each type sent in the SAK payload MUST be present. The attributes must follow the format defined in ISAKMP (Section 3.3 of [RFC2408]). In the table, attributes defined as TV are marked as Basic (B); attributes defined as TLV are marked as

Variable (V).

KEK Class	Value	Type
RESERVED	0	
KEK_ALGORITHM_KEY	1	V
SIG_ALGORITHM_KEY	2	V

If the KEK key packet is included, there MUST be only one present in the KD payload.

5.6.2.1. KEK_ALGORITHM_KEY

The KEK_ALGORITHM_KEY class declares the encryption key for this SPI is contained in the Key Packet Attribute. The encryption algorithm that will use this key was specified in the SAK payload.

If the mode of operation for the algorithm requires an Initialization Vector (IV), an explicit IV MUST be included in the KEK_ALGORITHM_KEY before the actual key.

5.6.2.2. SIG_ALGORITHM_KEY

The SIG_ALGORITHM_KEY class declares that the public key for this SPI is contained in the Key Packet Attribute, which may be useful when no public key infrastructure is available. The signature algorithm that will use this key was specified in the SAK payload.

5.6.3. LKH Download Type

The LKH key packet is comprised of attributes representing different nodes in the LKH key tree.

The following attributes are used to pass an LKH KEK array in the KD payload. The attributes must follow the format defined in ISAKMP (Section 3.3 of [RFC2408]). In the table, attributes defined as TV are marked as Basic (B); attributes defined as TLV are marked as Variable (V).

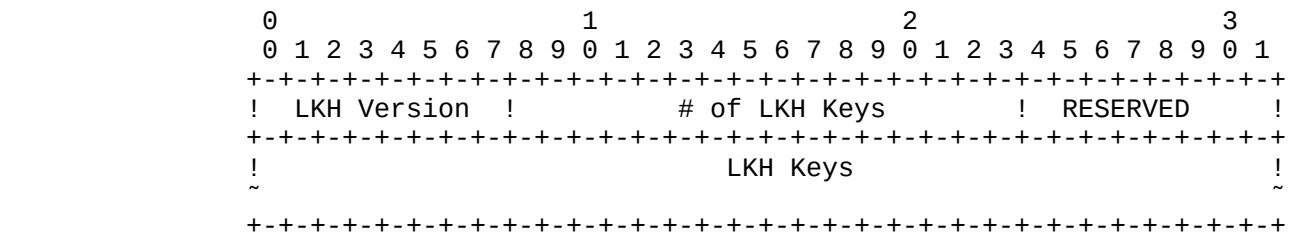
KEK Class	Value	Type
RESERVED	0	
LKH_DOWNLOAD_ARRAY	1	V
LKH_UPDATE_ARRAY	2	V
SIG_ALGORITHM_KEY	3	V
RESERVED	4-127	
Private Use	128-255	

If an LKH key packet is included in the KD payload, there must be only one present.

5.6.3.1. LKH_DOWNLOAD_ARRAY

This attribute is used to download a set of keys to a group member. It MUST NOT be included in a GROUPKEY-PUSH message KD payload if the GROUPKEY-PUSH is sent to more than the group member. If an LKH_DOWNLOAD_ARRAY attribute is included in a KD payload, there must be only one present.

This attribute consists of a header block, followed by one or more LKH keys.

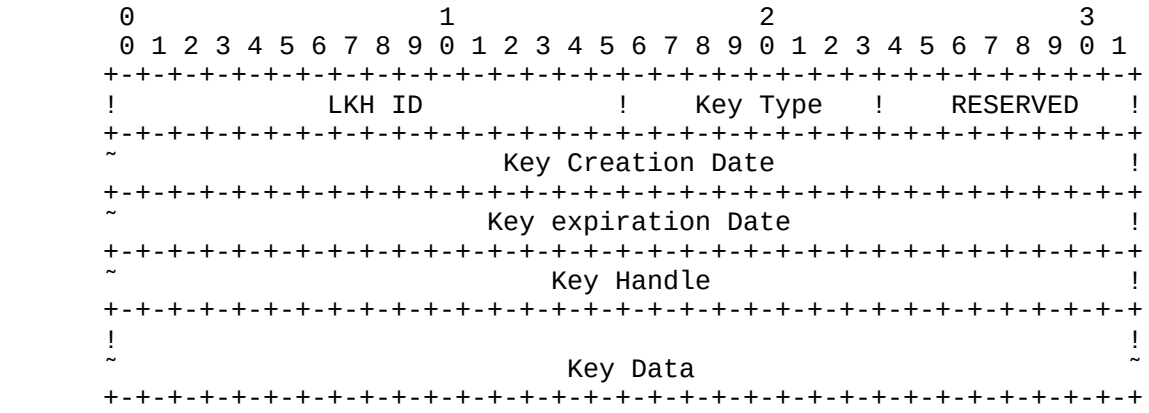


The KEK_LKH attribute fields are defined as follows:

o LKH version (1 octet) -- Version of the LKH data format. Must be one.

o Number of LKH Keys (2 octets) -- This value is the number of distinct LKH keys in this sequence.

o RESERVED (1 octet) -- Unused, set to zero. Each LKH Key is defined as follows:



o LKH ID (2 octets) -- Identity of the LKH node. A GCKS is free to choose the ID in an implementation-specific manner (e.g., the position of this key in a binary tree structure used by LKH).

o Key Type (1 octet) -- Encryption algorithm for which this key data is to be used. This value is specified in Section 5.3.3.

o RESERVED (1 octet) -- Unused, set to zero.

o Key Creation Date (4 octets) -- Time value of when this key data was originally generated. A time value of zero indicates that there is no time before which this key is not valid.

o Key Expiration Date (4 octets) -- Time value of when this key is no longer valid for use. A time value of zero indicates that this key does not have an expiration time.

o Key Handle (4 octets) -- Value assigned by the GCKS to uniquely identify a key within an LKH ID. Each new key distributed by the GCKS for this node will have a key handle identity distinct from previous or successive key handles specified for this node.

o Key Data (variable length) -- Key data, which is dependent on the Key Type algorithm for its format. If the mode of operation for the algorithm requires an Initialization Vector (IV), an explicit IV MUST be included in the Key Data field prepended to the actual key.

The Key Creation Date and Key expiration Dates MAY be zero. This is necessary in the case where time synchronization within the group is not possible.

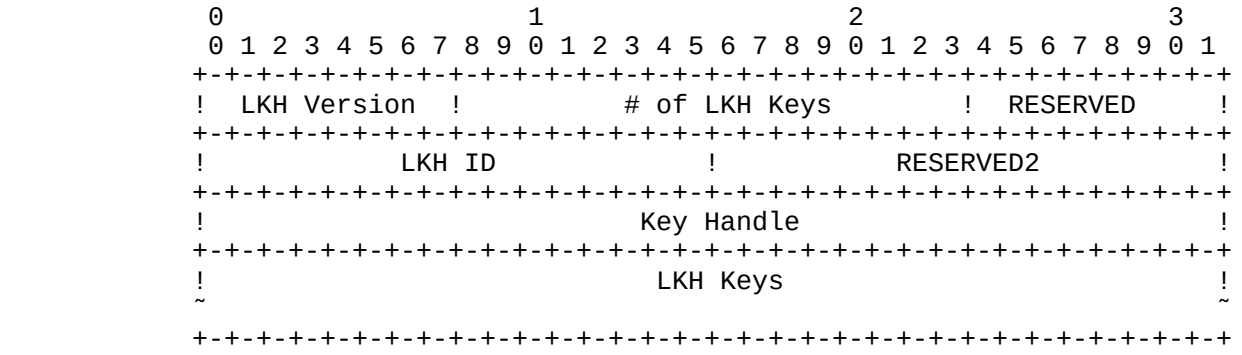
The first LKH Key structure in an LKH_DOWNLOAD_ARRAY attribute contains the Leaf identifier and key for the group member. The rest

of the LKH Key structures contain keys along the path of the key tree in order from the leaf, culminating in the group KEK.

5.6.3.2. LKH_UPDATE_ARRAY

This attribute is used to update the keys for a group. It is most likely to be included in a GROUPKEY-PUSH message KD payload to rekey the entire group. This attribute consists of a header block, followed by one or more LKH keys, as defined in the previous section.

There may be any number of UPDATE_ARRAY attributes included in a KD payload.



o LKH version (1 octet) -- Version of the LKH data format. Must be one.

o Number of LKH Keys (2 octets) -- Number of distinct LKH keys in this sequence.

o RESERVED (1 octet) -- Unused, set to zero.

o LKH ID (2 octets) -- Node identifier associated with the key used to encrypt the first LKH Key.

o RESERVED2 (2 octets) -- Unused, set to zero.

o Key Handle (4 octets) -- Value assigned by the GCKS to uniquely identify the key within the LKH ID used to encrypt the first LKH Key.

The LKH Keys are as defined in the previous section. The LKH Key structures contain keys along the path of the key tree in order from the LKH ID found in the LKH_UPDATE_ARRAY header, culminating in the group KEK. The Key Data field of each LKH Key is encrypted with the LKH key preceding it in the LKH_UPDATE_ARRAY attribute. The first

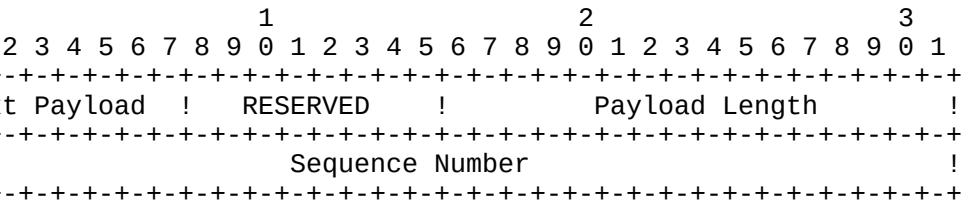
LKH Key is encrypted under the key defined by the LKH ID and Key Handle found in the LKH_UPDATE_ARRAY header.

5.6.3.3. SIG_ALGORITHM_KEY

The SIG_ALGORITHM_KEY class declares that the public key for this SPI is contained in the Key Packet Attribute, which may be useful when no public key infrastructure is available. The signature algorithm that will use this key was specified in the SAK payload.

5.7. Sequence Number Payload

The Sequence Number Payload (SEQ) provides an anti-replay protection for GROUPKEY-PUSH messages. Its use is similar to the Sequence Number field defined in the IPsec ESP protocol [RFC4303].



The Sequence Number Payload fields are defined as follows:

o Next Payload (1 octet) -- Identifier for the payload type of the next payload in the message. If the current payload is the last in the message, then this field will be zero.

o RESERVED (1 octet) -- Unused, set to zero.

o Payload Length (2 octets) -- Length in octets of the current payload, including the generic payload header.

o Sequence Number (4 octets) -- This field contains a monotonically increasing counter value for the group. It is initialized to zero by the GCKS, and incremented in each subsequently-transmitted message. Thus the first packet sent for a given Rekey SA will have a Sequence Number of 1. The GDOI implementation keeps a sequence counter as an attribute for the Rekey SA and increments the counter upon receipt of a GROUPKEY-PUSH message. The current value of the sequence number must be transmitted to group members as a part of the Registration SA payload. A group member must keep a sliding receive window. The window must be treated as in the ESP protocol [RFC4303] Section 3.4.3.

5.8. Nonce

The data portion of the Nonce payload (i.e., Ni_b and Nr_b included in the HASHs) MUST be a value between 8 and 128 bytes.

5.9. Delete

There are times the GCKS may want to signal to receivers to delete SAs, for example at the end of a broadcast. Deletion of keys may be accomplished by sending an ISAKMP Delete payload (Section 3.15 of [RFC2408]) as part of a GDOI GROUPKEY-PUSH message.

One or more Delete payloads MAY be placed following the SEQ payload in a GROUPKEY-PUSH message. If a GCKS has no further SAs to send to group members, the SA and KD payloads MUST be omitted from the message.

The following fields of the Delete Payload are further defined as follows:

o The Domain of Interpretation field contains the GDOI DOI.

o The Protocol-Id field contains TEK protocol id values defined in Section 5.5 of this document. To delete a KEK SA, the value of zero MUST be used as the protocol id. Note that only one protocol id value can be defined in a Delete payload. If a TEK SA and a KEK SA must be deleted, they must be sent in different Delete payloads.

There may be circumstances where the GCKS may want to start over with a clean slate. If the administrator is no longer confident in the integrity of the group, the GCKS can signal deletion of all policy 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 a protocol_id of a TEK protocol_id value, followed by another delete payload with a spi of zero and protocol_id of zero, indicating that the KEK SA should be deleted.

6. Algorithm Selection

For GDOI implementations to interoperate, they must support one or more security algorithms in common. This section specifies the security algorithm implementation requirements for standards-conformant GDOI implementations. In all cases the choices are intended to maintain at least 112 bits of security [SP.800-131].

Algorithms not referenced in this section MAY be used.

6.1. KEK

These tables list the algorithm selections for values related to the KEK.

Requirement	KEK Management Algorithm
SHOULD	LKH

Requirement	KEK Algorithm (notes)
MUST	KEK_ALG_AES with 128-bit keys
SHOULD NOT	KEK_ALG_DES (1)

Requirement	KEK Signature Hash Algorithm (notes)
MUST	SIG_HASH_SHA256
SHOULD	SIG_HASH_SHA1 (2)
SHOULD NOT	SIG_HASH_MD5 (3)

Requirement	KEK Signature Algorithm (notes)
MUST	SIG_ALG_RSA with 2048-bit keys

Notes:

(1) DES, with its small key size and corresponding security strength is of questionable security for general use

(2) The use of SIG_HASH_SHA1 as a signature hash algorithm used with GROUPKEY-PUSH messages remains safe at the time of this writing, and is a widely deployed signature hash algorithm.

(3) Although a real weakness with second preimage resistance with MD5 has not been found at the time of this writing, the security strength of MD5 has been shown to be rapidly declining over time and it’s use should be understood and carefully weighed.

6.2. TEK

The following table lists the requirements for Security Protocol support for an implementation.

Requirement	KEK Management Algorithm
MUST	GDOI_PROTO_IPSEC_ESP

7. Security Considerations

GDOI is a security association (SA) management protocol for groups of senders and receivers. Unlike a data security protocol, SA management includes a key establishment protocol to securely establish keys at communication endpoints. This protocol performs entity authentication of the GDOI member or Group Controller/Key Server (GCKS), it provides confidentiality of key management messages, and it provides source authentication of those messages. This protocol also uses best-known practices for defense against man-in-middle, connection hijacking, replay, reflection, and denial-of-service (DOS) attacks on unsecured networks [STS], [RFC2522], [SKEME]. GDOI assumes the network is not secure and may be under the complete control of an attacker.

GDOI assumes that the host computer is secure even though the network is insecure. GDOI ultimately establishes keys among members of a group, which MUST be trusted to use those keys in an authorized manner according to group policy. The security of GDOI, therefore, is as good as the degree to which group members can be trusted to protect authenticators, encryption keys, decryption keys, and message authentication keys.

There are three phases of GDOI as described in this document: an ISAKMP Phase 1 protocol, a new exchange called GROUPKEY-PULL which is protected by the ISAKMP Phase 1 protocol, and a new message called GROUPKEY-PUSH. Each phase is considered separately below.

7.1. ISAKMP Phase 1

As described in this document, GDOI uses the Phase 1 exchanges defined in [RFC2409] to protect the GROUPKEY-PULL exchange. Therefore all security properties and considerations of those exchanges (as noted in [RFC2409]) are relevant for GDOI.

GDOI may inherit the problems of its ancestor protocols [FS00], such as identity exposure, absence of unidirectional authentication, or stateful cookies [PK01]. GDOI could benefit, however, from improvements to its ancestor protocols just as it benefits from years of experience and work embodied in those protocols. To reap the benefits of future IKE improvements, however, GDOI would need to be revised in a future standards-track RFC, which is beyond the scope of this specification.

7.1.1. Authentication

Authentication is provided via the mechanisms defined in [RFC2409], namely Pre-Shared Keys or Public Key encryption.

7.1.2. Confidentiality

Confidentiality is achieved in Phase 1 through a Diffie-Hellman exchange that provides keying material, and through negotiation of encryption transforms.

The Phase 1 protocol will be protecting encryption and integrity keys sent in the GROUPKEY-PULL protocol. The strength of the encryption used for Phase 1 SHOULD exceed that of the keys send in the GROUPKEY-PULL protocol.

7.1.3. Man-in-the-Middle Attack Protection

A successful man-in-the-middle or connection-hijacking attack foils entity authentication of one or more of the communicating entities during key establishment. GDOI relies on Phase 1 authentication to defeat man-in-the-middle attacks.

7.1.4. Replay/Reflection Attack Protection

In a replay/reflection attack, an attacker captures messages between GDOI entities and subsequently forwards them to a GDOI entity. Replay and reflection attacks seek to gain information from a subsequent GDOI message response or seek to disrupt the operation of a GDOI member or GCKS entity. GDOI relies on the Phase 1 nonce mechanism in combination with a hash-based message authentication code to protect against the replay or reflection of previous key management messages.

7.1.5. Denial of Service Protection

A denial of service attacker sends messages to a GDOI entity to cause that entity to perform unneeded message authentication operations. GDOI uses the Phase 1 cookie mechanism to identify spurious messages prior to cryptographic hash processing. This is a "weak" form of denial of service protection in that the GDOI entity must check for good cookies, which can be successfully imitated by a sophisticated attacker. The Phase 1 cookie mechanism is stateful, and commits memory resources for cookies, but stateless cookies are a better defense against denial of service attacks.

7.2. GROUPKEY-PULL Exchange

The GROUPKEY-PULL exchange allows a group member to request SAs and keys from a GCKS. It runs as a "phase 2" protocol under protection of the Phase 1 security association.

7.2.1. Authentication

Peer authentication is not required in the GROUPKEY-PULL protocol. It is running in the context of the Phase 1 protocol, which has previously authenticated the identity of the peer.

Message authentication is provided by HASH payloads in each message, where the HASH is defined to be over SKEYID_a (derived in the Phase 1 exchange), the ISAKMP Message-ID, and all payloads in the message. Because only the two endpoints of the exchange know the SKEYID_a value, this provides confidence that the peer sent the message.

7.2.2. Confidentiality

Confidentiality is provided by the Phase 1 security association, after the manner described in [RFC2409].

7.2.3. Man-in-the-Middle Attack Protection

Message authentication (described above) includes a secret known only to the group member and GCKS when constructing a HASH payload. This prevents man-in-the-middle and connection-hijacking attacks because an attacker would not be able to change the message undetected.

7.2.4. Replay/Reflection Attack Protection

Nonces provide freshness of the GROUPKEY-PULL exchange. The group member and GCKS exchange nonce values first two messages. These nonces are included in subsequent HASH payload calculations. The Group member and GCKS MUST NOT perform any computationally expensive tasks before receiving a HASH with its own nonce included. The GCKS MUST NOT update the group management state (e.g., LKH key tree) until it receives the third message in the exchange with a valid HASH payload including its own nonce.

Implementations SHOULD keep a record of recently received GROUPKEY-PULL messages and reject messages that have already been processed. This enables an early discard of the replayed messages.

7.2.5. Denial of Service Protection

A GROUPKEY-PULL message identifies its messages using a cookie pair from the Phase 1 exchange that precedes it. The cookies provide a weak form of denial of service protection as described above, in the sense that a GROUPKEY-PULL message with invalid cookies will be discarded.

The replay protection mechanisms described above provide the basis

for denial of service protection.

7.2.6. Authorization

A GCKS implementation should maintain an authorization list of authorized group members. Group members will specifically list each authorized GCKS in its Group Peer Authorization Database (GPAD) [RFC5374].

7.3. GROUPKEY-PUSH Exchange

The GROUPKEY-PUSH exchange is a single message that allows a GCKS to send SAs and keys to group members. This is likely to be sent to all members using an IP multicast group. This provides an efficient rekey and group membership adjustment capability.

7.3.1. Authentication

The GROUPKEY-PULL exchange identifies a public key that is used for message authentication. The GROUPKEY-PUSH message is digitally signed using the corresponding private key held by the GCKS or its delegate. This digital signature provides source authentication for the message. Thus, GDOI protects the GCKS from impersonation in group environments.

7.3.2. Confidentiality

The GCKS encrypts the GROUPKEY-PUSH message with an encryption key that was established by the GROUPKEY-PULL exchange.

7.3.3. Man-in-the-Middle Attack Protection

This combination of confidentiality and message authentication services protects the GROUPKEY-PUSH message from man-in-middle and connection-hijacking attacks.

7.3.4. Replay/Reflection Attack Protection

The GROUPKEY-PUSH 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 sequence number to a sliding window, in the same manner as the ESP protocol uses sequence numbers.

Implementations SHOULD keep a record of recently received GROUPKEY-PUSH messages and reject duplicate messages. This enables an early discard of the replayed messages.

7.3.5. Denial of Service Protection

A cookie pair identifies the security association for the GROUPKEY-PUSH message. The cookies thus serve as a weak form of denial-of-service protection for the GROUPKEY-PUSH message.

The digital signature used for message authentication has a much greater computational cost than a message authentication code and could amplify the effects of a denial of service attack on GDOI members who process GROUPKEY-PUSH messages. The added cost of digital signatures is justified by the need to prevent GCKS impersonation: If a shared symmetric key were used for GROUPKEY-PUSH message authentication, then GCKS source authentication would be impossible and any member would be capable of GCKS impersonation.

The potential of the digital signature amplifying a denial of service attack is mitigated by the order of operations a group member takes, where the least expensive cryptographic operation is performed first. The group member first decrypts the message using a symmetric cipher. If it is a validly formed message then the sequence number is checked against the replay window. Only if the sequence number is valid is the digital signature verified. Thus in order for a denial of service attack to be mounted, an attacker would need to know both the symmetric encryption key used for confidentiality, and a valid sequence number. Generally speaking this means only current group members can effectively deploy a denial of service attack.

7.3.6. Forward Access Control

If a group management algorithm (such as LKH) is used, forward access control may not be ensured in some cases. This can happen if some group members are denied access to the group in the same GROUPKEY-PUSH message as new policy and TEKs are delivered to the group. As discussed in Section 1.5.1, forward access control can be maintained by sending multiple GROUPKEY-PUSH messages, where the group membership changes are sent from the GCKS separate from the new policy and TEKs.

8. IANA Considerations

This memo requests IANA to make several additions to existing registries, and to add sever new GDOI registries. When the new registries are added, the following terms are to be applied as described in the Guidelines for Writing an IANA Considerations Section in RFCs [RFC5226]: Standards Action, and Private Use.

8.1. Additions to current registries

The GDOI KEK Attribute named SIG_HASH_ALGORITHM [GDOI-REG] should be assigned several new Algorithm Type values from the RESERVED space to represent the SHA-256, SHA-384, and SHA-512 hash algorithms as defined in [FIPS.180-2.2002]. The new algorithm names should be SIG_HASH_SHA256, SIG_HASH_SHA384, and SIG_HASH_SHA512 respectively and have the values of TBD-2, TBD-3, and TBD-4 respectively.

The GDOI KEK Attributed named SIG_ALGORITHM [GDOI-REG] should be assigned a new Algorithm Type value from the RESERVED space to represent the RSA PSS encoding type. The new algorithm name should be SIG_ALG_RSA_PSS, and has the value of TBD-6.

A new GDOI SA TEK type Protocol-ID type [GDOI-REG] should be assigned from the RESERVED space. The new algorithm id should be called GDOI_PROTO_IPSEC_AH, refers to the IPsec AH encapsulation, and has a value of TBD-5.

A new Next Payload Type [ISAKMP-REG] should be assigned. The new type is called "SA Group Associated Policy (GAP)", and has a value of TBD-1.

8.2. New registries

A new namespace should be created in the GDOI Payloads registry [GDOI-REG] to describe SA GAP Payload Values. The following rules apply to define the attributes in SA SSA Payload Values:

Attribute Type	Value	Type
RESERVED	0	
ACTIVATION_TIME_DELAY	1	B
DEACTIVATION_TIME_DELAY	2	B
SENDER_ID	3	V
Standards Action	4-127	
Private Use	128-255	

A new IPsec Security Association Attribute [ISAKMP-REG] defining the preservation of IP addresses is needed. The attribute class is

called "Address Preservation", and it is a Basic type. The following rules apply to define the values of the attribute:

Name	Value
Reserved	0
None	1
Source-Only	2
Destination-Only	3
Source-And-Destination	4
Standards Action	5-61439
Private Use	61440-65535

A new IPsec Security Association Attribute [ISAKMP-REG] defining the SA direction is needed. The attribute class is called "SA Direction", and it is a Basic type. The following rules apply to define the values of the attribute:

Name	Value
Reserved	0
Sender-Only	1
Receiver-Only	2
Symmetric	3
Standards Action	4-61439
Private Use	61440-65535

9. Acknowledgements

This text updates RFC 3547, and the authors with to thank Mark Baugher and Hugh Harney for their extensive contributions.

The authors are grateful to Catherine Meadows for her careful review and suggestions for mitigating the man-in-the-middle attack she had previously identified. Yoav Nir provided many useful technical and editorial comments and suggestions for improvement.

10. References

10.1. Normative References

[I-D.ietf-msec-ipsec-group-counter-modes]
McGrew, D. and B. Weis, "Using Counter Modes with Encapsulating Security Payload (ESP) and Authentication Header (AH) to Protect Group Traffic", draft-ietf-msec-ipsec-group-counter-modes-06 (work in progress), September 2010.

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

[RFC4302] Kent, S., "IP Authentication Header", RFC 4302, December 2005.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[RFC5374] Weis, B., Gross, G., and D. Ignjatic, "Multicast Extensions to the Security Architecture for the Internet Protocol", RFC 5374, November 2008.

10.2. Informative References

[FIPS.180-2.2002]
National Institute of Standards and Technology, "Secure Hash Standard", FIPS PUB 180-2, August 2002, <http://csrc.nist.gov/publications/fips/fips180-2/fips180-2.pdf>.

[FIPS186-3]
"Digital Signature Standard (DSS)", United States of America, National Institute of Science and Technology Federal Information Processing Standard (FIPS) 186-2, June 2009.

[FIPS197] "Advanced Encryption Standard (AES)", United States of America, National Institute of Science and Technology Federal Information Processing Standard (FIPS) 197, November 2001.

[FIPS46-3]
"Data Encryption Standard (DES)", United States of America, National Institute of Science and Technology Federal Information Processing Standard (FIPS) 46-3, October 1999.

[FIPS81] "DES Modes of Operation", United States of America, National Institute of Science and Technology Federal Information Processing Standard (FIPS) 81, December 1980.

[FS00] Ferguson, N. and B. Schneier, Counterpane, "A Cryptographic Evaluation of IPsec", <http://www.counterpane.com/ipsec.html>.

[GDOI-REG] Internet Assigned Numbers Authority, "Group Domain of Interpretation (GDOI) Payload Type Values", IANA Registry, December 2004, <http://www.iana.org/assignments/gdoi-payloads>.

[HD03] Hardjono, T. and L. Dondeti, "Multicast and Group Security", Artech House Computer Security Series, ISBN 1-58053-342-6, 2003.

[I-D.weis-gdoi-mac-tek] Weis, B. and S. Rowles, "GDOI Generic Message Authentication Code Policy", draft-weis-gdoi-mac-tek-01 (work in progress), June 2010.

[ISAKMP-REG] "'Magic Numbers' for ISAKMP Protocol", <http://www.iana.org/assignments/isakmp-registry>.

[MP04] Meadows, C. and D. Pavlovic, "Deriving, Attacking, and Defending the GDOI Protocol", ESORICS 2004 pp. 53-72, September 2004.

[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/>.

[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, <http://download.nai.com/products/media/nai/misc/oft052098.ps>.

[PK01] Perlman, R. and C. Kaufman, "Analysis of the IPsec Key Exchange Standard", WET-ICE conference , 2001, <http://sec.femto.org/wetice-2001/papers/radia-paper.pdf>.

[RFC2403] Madson, C. and R. Glenn, "The Use of HMAC-MD5-96 within

ESP and AH", RFC 2403, November 1998.

[RFC2404] Madson, C. and R. Glenn, "The Use of HMAC-SHA-1-96 within ESP and AH", RFC 2404, November 1998.

[RFC2407] Piper, D., "The Internet IP Security Domain of Interpretation for ISAKMP", RFC 2407, November 1998.

[RFC2408] Maughan, D., Schneider, M., and M. Schertler, "Internet Security Association and Key Management Protocol (ISAKMP)", RFC 2408, November 1998.

[RFC2409] Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

[RFC2522] Karn, P. and W. Simpson, "Photuris: Session-Key Management Protocol", RFC 2522, March 1999.

[RFC2627] Wallner, D., Harder, E., and R. Agee, "Key Management for Multicast: Issues and Architectures", RFC 2627, June 1999.

[RFC3447] Jonsson, J. and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1", RFC 3447, February 2003.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC3686] Housley, R., "Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP)", RFC 3686, January 2004.

[RFC3740] Hardjono, T. and B. Weis, "The Multicast Group Security Architecture", RFC 3740, March 2004.

[RFC3947] Kivinen, T., Swander, B., Huttunen, A., and V. Volpe, "Negotiation of NAT-Traversal in the IKE", RFC 3947, January 2005.

[RFC4046] Baugher, M., Canetti, R., Dondeti, L., and F. Lindholm, "Multicast Security (MSEC) Group Key Management Architecture", RFC 4046, April 2005.

[RFC4106] Viega, J. and D. McGrew, "The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP)", RFC 4106, June 2005.

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

[RFC4309] Housley, R., "Using Advanced Encryption Standard (AES) CCM Mode with IPsec Encapsulating Security Payload (ESP)", RFC 4309, December 2005.

[RFC4430] Sakane, S., Kamada, K., Thomas, M., and J. Vilhuber, "Kerberized Internet Negotiation of Keys (KINK)", RFC 4430, March 2006.

[RFC4543] McGrew, D. and J. Viega, "The Use of Galois Message Authentication Code (GMAC) in IPsec ESP and AH", RFC 4543, May 2006.

[RFC4754] Fu, D. and J. Solinas, "IKE and IKEv2 Authentication Using the Elliptic Curve Digital Signature Algorithm (ECDSA)", RFC 4754, January 2007.

[RFC4868] Kelly, S. and S. Frankel, "Using HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 with IPsec", RFC 4868, May 2007.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC5903] Fu, D. and J. Solinas, "Elliptic Curve Groups modulo a Prime (ECP Groups) for IKE and IKEv2", RFC 5903, June 2010.

[RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.

[SKEME] Krawczyk, H., "SKEME: A Versatile Secure Key Exchange Mechanism for Internet", ISOC Secure Networks and Distributed Systems Symposium San Diego, 1996.

[SP.800-131] Barker, E. and A. Roginsky, "Recommendation for the Transitioning of Cryptographic Algorithms and Key Lengths", United States of America, National Institute of Science and Technology DRAFT NIST Special Publication 800-131, June 2010.

[SP.800-38A] Dworkin, M., "Recommendation for Block Cipher Modes of Operation", United States of America, National Institute

of Science and Technology NIST Special Publication 800-38A 2001 Edition, December 2001.

[STS] Diffie, W., Van Oorschot, P., and M. Wiener, "Authentication and Authenticated Key Exchanges", Designs, Codes and Cryptography, 2, 107-125 (1992), Kluwer Academic Publishers, 1992.

Appendix A. Alternate GDOI Phase 1 protocols

This section describes a manner in which other protocols could be used as GDOI Phase 1 protocols in place of the ISAKMP Phase 1 protocol. However, they are not specified as a part of this document. A separate document MUST be written in order for another protocol to be used as a GDOI Phase 1 protocol.

Other possible phase 1 protocols are also described in [RFC4046].

Any GDOI phase 1 protocol MUST satisfy the requirements specified in Section 2 of this document.

A.1. IKEv2 Exchange

Version 2 of the IKE protocol (IKEv2) [RFC5996] has been published. That protocol simplifies IKE processing, and combines the two phases of IKE. An IKEv2 Phase 1 negotiates an IPsec SA during phase 1, which was not possible in IKE. However, IKEv2 also defines a phase 2 protocol. The phase 2 protocol is protected by the Phase 1, similar in concept to how IKE Quick Mode is protected by the IKE Phase 1 protocols in [RFC2409].

It would be possible to define GDOI as a phase 2 protocol protected by an IKEv2 initial exchange. Alternatively, it would be possible to define a new protocol re-using some of the IKEv2 initial exchange (e.g., IKE_SA_INIT).

A.2. KINK Protocol

The Kerberized Internet Negotiation of Keys (KINK) [RFC4430] has defined a method of encapsulating an IKEv1 Quick Mode [RFC2409] encapsulated in Kerberos KRB_AP_REQ and KRB_AP_REP payloads. KINK provides a low-latency, computationally inexpensive, easily managed, and cryptographically sound method of setting up IPsec security associations.

The KINK message format includes a GDOI field in the KINK header. The [RFC4430] document defines the DOI for the IPsec DOI.

A new DOI for KINK could be defined which would encapsulate a GROUPKEY-PULL exchange in the Kerberos KRB_AP_REQ and KRB_AP_REP payloads. As such, GDOI would benefit from the computational efficiencies of KINK.

Appendix B. Significant Changes from RFC 3547

The following significant changes have been made from RFC 3547.

- The Proof of Possession (POP) payload was removed from the GROUPKEY-PULL exchange. It provided an alternate form of authorization, but its use was underspecified. Furthermore, Meadows and Pavlovic [MP04] discussed a man-in-the-middle attack on the POP authorization method, which would require changes to its semantics. No known implementation of RFC 3547 supported the POP payload, so it was removed. Removal of the POP payload obviated the need for the CERT payload in that exchange and it was removed as well.

- The Key Exchange Payloads (KE_I, KE_R) payloads were removed from the GROUPKEY-PULL exchange. However, the specification for computing keying material for the additional encryption function in RFC 3547 is faulty. Furthermore, it has been observed that because the GDOI registration message uses strong ciphers and provides authenticated encryption, additional encryption of the keying material in a GDOI registration message provides negligible value. Therefore, the use of KE payloads is deprecated in this memo.

- The Certificate Payload (CERT) was removed from the GROUPKEY-PUSH exchange. The use of this payload was underspecified. In all known use cases, the public key of used to verify the GROUPKEY-PUSH payload is distributed directly from the key server as part of the GROUPKEY-PULL exchange.

- Supported cryptographic algorithms were changed to meet current guidance. Implementations are required to support AES with 128-bit keys to encrypt the rekey message, and SHA-256 for cryptographic signatures. The use of DES is deprecated.

- New protocol support for AH.

- New protocol definitions were added to conform to the most recent Security Architecture for the Internet Protocol [RFC4301] and the Multicast Extensions to the Security Architecture for the Internet Protocol[RFC5374]. This includes addition of the GAP payload

- New protocol definitions were added to support Using Counter Modes with ESP and AH to Protect Group Traffic[I-D.ietf-msec-ipsec-group-counter-modes].

Authors' Addresses

Brian Weis
Cisco Systems
170 W. Tasman Drive
San Jose, California 95134-1706
USA

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

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

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

Thomas Hardjono
MIT
77 Massachusetts Ave.
Cambridge, Massachusets 02139
USA

Phone: +1-781-729-9559
Email: hardjono@mit.edu