Multicast Router Key Management Protocol (MaRK)
draft-hartman-karp-mrkmp-05

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.

Requirements Language

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 RFC 2119 [RFC2119].

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 March 10, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.
# Table of Contents

1. Introduction .......................... 4
   1.1. Terminology .......................... 4
   1.2. Relationship to IKEv2 ................. 4
   1.3. Relationship to GDOI ................. 5
2. Overview .............................. 5
   2.1. Types of Keys ......................... 5
       2.1.1. Key Encryption Key .............. 6
       2.1.2. Protocol Master Keys ............ 7
   2.2. GCKS Election ....................... 8
   2.3. Initial Exchange ..................... 9
   2.4. Group Join Exchange ................. 9
   2.5. Group Key Management ............... 10
3. GCKS Election .......................... 10
   3.1. A new GCKS is Elected .............. 12
       3.1.1. Parameters, Timers, and Events .. 12
       3.1.2. Initial ......................... 14
       3.1.3. Validate ......................... 15
       3.1.4. GCKS2 .......................... 16
       3.1.5. GCKS .......................... 17
       3.1.6. Member ......................... 18
       3.1.7. Follower ......................... 18
   3.2. Merging Partitioned Networks ....... 19
   3.3. Operations on Receiving a Packet .... 20
4. Key Download Payload ................... 21
5. Initial Exchanges Details ............. 21
6. Group Management Unicast Exchanges .... 22
   6.1. Group Join Exchange ................. 22
7. Group Key Management Operation ....... 22
   7.1. General operation .................... 22
   7.2. Out of Sequence Space ............... 22
   7.3. Changing the Active GCKS ............ 23
   7.4. Reboot Cases ....................... 23
   8.1. Joining a Group ..................... 23
   8.2. Priority Adjustment .................. 24
   8.3. Leaving a Group ..................... 24
   8.4. Out of Sequence Space ............... 24
9. Security Considerations ............... 24
10. Acknowledgements ...................... 26
11. References ............................ 26
   11.1. Normative References ............... 26
   11.2. Informative References .............. 26
Authors’ Addresses ..................... 27
1. Introduction

Many routing protocols such as OSPF [RFC2328] and IS-IS [RFC1142] 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 an entity 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

GCKS (Group Controller/Key Server): a GCKS is a particular group member which establishes security associations among other authorized group members which it serves.

group: a group specified in this document is a set of routers, called group members, which are located on a single broadcast domain/ link/ NBMA segment and use a one-to-many or multicast model of communication.

1.2. Relationship to IKEv2

IKEv2 [RFC4306] 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 [RFC2409], it provides all the needed flexibility in one-to-one authentication.
IKev2 is expanded to support authentication of routers in [I-D.mahesh-karp-rkmp]. That specification describes how IKEv2 can be used for unicast routing protocols. This specification is part of expanding that work to cover multicast routing.

1.3. Relationship to 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. After the phase 2 exchange, additional key management operations can be accomplished via GDOI messages sent within the group.

In [I-D.yeung-g-ikev2] a group management approach is defined for IKEv2. This approach is extended in [I-D.tran-karp-mrmp] to provide for management of routing messages. This specification acts as a companion to that specification, providing an election protocol and some of the interactions with routing protocols.

2. Overview

2.1. Types of Keys

MaRK manipulates several different types of symmetric keys:

PSK (Pre-Shared Key): PSKs are pair-wise unique keys used 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 MaRK.

Peer key management key: Routers share a key with the GCKS that is a result of the RP_INIT exchange.

KEK (Key Encryption Key): A 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 MaRK 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 MaRK 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. This key is distinguished from the ‘transport key’ (see next) in that this
Protocol Master Key may be used in a cryptographic operation in order to derive a specific transport key.

Transport key: A 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 design. 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 RP_INIT and RP_AUTH exchange with a GCKS. If the exchanges are successful, 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 rather than unicast to an individual peer. The authenticity, integrity and confidentiality of group key management messages need to be protected with the KEK.

As part of establishing the association, the router joining the group is given an valid period (which is identified by a start time point and an expire time point) 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, the group members delete that KEK. Unless local policy configuration dictates otherwise, the group member will perform a new initial exchange to the GCKS in order to establish a new KEK. This solution is useful for enforcing "forward security" in the cases where a router is no longer authorized to be part of the group. That is, only valid group members can obtain the new KEK while the ones which have leave the group will be rejected.

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 Master Keys

Current routing protocols directly use the protocol master key to protect the integrity of 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 MaRK 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 secure mechanism would be provisioned to enable one to confirm that the peer’s announcement of its nonce was fresh and authentic; this mechanism would almost certainly involve some form of interaction with the router wishing to guarantee freshness in order to resistant, e.g., replay attacks.

There are two key advantages of this separation between transport keys and protocol master keys. The first is that the interaction between the MaRK 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.

A simple compare between the keys described in this section is provided in the following table.
### 2.2. GCKS Election

Before a MaRK system actually starts working, the routers in the multicast group need to elect a GCKS so that they can obtain cryptographic keys to secure subsequent exchanges of routing information. MaRK 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 MaRK system and a routing protocol implementation will be introduced if the MaRK 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 MaRK 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 MaRK

<table>
<thead>
<tr>
<th>Keys</th>
<th>KMP usage vs. RP usage</th>
<th>Bootstrapping vs. Traffic</th>
<th>Group vs Pair-Wise</th>
<th>Other</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-Shared Keys</td>
<td>KMP usage</td>
<td>Bootstrapping</td>
<td>Pair-Wise</td>
<td></td>
<td>Distributed in an out-of-band way</td>
</tr>
<tr>
<td>Key Encryption Key</td>
<td>KMP usage</td>
<td>Bootstrapping</td>
<td>Group</td>
<td></td>
<td>For GCKS to distribute protocol master keys</td>
</tr>
<tr>
<td>Protocol Master Key</td>
<td>KMP usage or Both</td>
<td>Bootstrapping or Both</td>
<td>Group</td>
<td></td>
<td>Used by group members to secure routing packets or generate traffic keys</td>
</tr>
<tr>
<td>Transport Key</td>
<td>RP usage</td>
<td>Traffic</td>
<td>Group</td>
<td></td>
<td>Used by group members to secure routing packets</td>
</tr>
</tbody>
</table>
will function correctly.

A key design goal of the election protocol is to maximize the chance that some router permitted to take on the role of GCKS will be elected to that role even when attackers are injecting messages into the election process. The election process can be attacked to cause a router other than the most preferred router to be elected.

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 MaRK 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 protect integrity, confidentiality and authenticity of the subsequent 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, e.g., a preshared key, asymmetric keys or self-signing certificates. 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 MaRK messages is to get an initiator the information it needs to join a group and participate in a routing protocol. The initiator can contact a GCKS to apply to join a group that the GCKS manages. In the case a GCKS manages multiple groups concurrently, the initiator can additionally provide a group identifier to indicate which particular group it intends to join.

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 MaRK association used to create it. Once membership in a group is established, resources are not required to maintain the unicast association with the GCKS.

2.5. Group Key Management

After the establishment of a group, a KEK is shared by the GCKS and all the other group members. Using the KEK, the GCKS can securely send multicast messages to the group in order to, for example, update the set of protocol master keys, revoke the KEK, or initiate new group join exchanges.

Typically, a protocol master key may be changed for the purpose of replay protection or as a result of KEK update. The KEK needs to be updated whenever a new GCKS is elected or whenever it is administratively desirable to change the keys. For example, after an employee leaves an organization it might not be wise to keep using the KEKs (and any other keys) that the employee has accessed. A KEK update is also required 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. GKCS Election

After a successful GCKS election process, a single router is selected to act as the GCKS for a group. Similar with other popular announcer electing mechanisms (e.g., VRRP, HSRP), in MaRK, 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 six states (Initial, Validate, GCKS, GCKS2, Follower, 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 to the router with an advertisement. After receiving the advertisement from the GCKS, the router will try to register with the GCKS using the initial exchange. Typically this registration will succeed, and the state of the router is transferred to Member. After a certain period, if the router still does not receive any advertisement from a GCKS or other group members, the router then believes there is no other group member on the network and sets its state as GCKS. If during the period the router does not receive any advertisement from a GCKS but receives advertisements from other more preferred routers on the network, the router believes that the group is involved in a GCKS election process. The router then puts these routers into its candidate list. When the timer to end the Initial state expires, the router tries to authenticate the most preferred router in the candidate list and validate whether it can be a GCKS. If the validation result is positive, the router then transfers its state to Member, and the router being validated transfers its state to GCKS.

In the absence of attacks, this process functions similar to designated router election protocols in existing routing protocols. Because the election process happens before group keys are established, the initial election process is not integrity-protected. An attacker can inject fake GCKS announcements or initial announcements from fake routers that are more preferred than any router actually in the group. Such attacks can create a denial of service situation. If the election process does not converge within the expected time, or if an authentication attempt fails, then the group is probably under attack. A new state called GCKS2 is introduced. A router permitted to be the GCKS can enter the GCKS2 state after failing to validate a received announcement in the expected time. GCKS2 is used to increase the convergence speed while the system is under attack. If an initial router receives a GCKS2 announcement, the initial router can authenticate and validate the sender, and transfer its own state to Follower, similar to how it would respond to a GCKS announcement. GCKS2 routers attempt to validate each other and to use the resulting security keys to establish a router to act as GCKS. The GCKS2 state does not generate protocol master keys: until the election result in a GCKS only keying material needed for the election is produced. In the subsequent election, the router will wait for the election results from its GCKS2 router until its GCKS2 end timer expires. In this way, the authenticated entities generate a tree structure and avoid generating large amount of KEKs and protocol master keys when a adversary keeps sending fake GCKS announcements to disrupt election.

Apart from the initialization of a multicast group, 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 transfers its state to GCKS,
it will reply to the initial advertisements from other routers with
GCKS advertisements, even when the initial advertisements consist of
higher 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.

3.1. A new GCKS is Elected

This section is a detailed description of the election process.

In the following discussion, the packets are identified by all upper
case characters.

3.1.1. Parameters, Timers, and Events

Before going into detailed discussion, several parameters are
introduced:

- Initial_Anno_Interval, which is the time interval between
  INITIAL_ANNOUNCEMENTS).

- Initial_End_Interval, which is the time interval to transfer the
  state of a router from Initial to GCKS/Validate if it does not
  receive any GCKS or GCKS2 announcement on the link).

- Validate_End_Interval, which is the time interval for a router to
  transfer its state from Validate to GCKS2 if it does not find any
other more preferred router).

- **GCKS_Down_Interval**, which is the time interval for a Member router to declare a GCKS router is down).

- **GCKS2_Down_Interval**, which is the time interval for a Follower router to declare a GCKS2 router is down).

- **GCKS2_End_Interval**, which is the time interval for a router to transfer its state from GCKS2 to GCKS if it does not find any other more preferred router).

- **GCKS_Anno_Interval**, which is the time interval between GCKS_ANNOUNCEMENTS).

- **GCKS2_Anno_Interval**, which is the time interval between GCKS2_ANNOUNCEMENTS).

Correspondingly, each router in MaRK has several timers, Initial_Anno_Timer, Initial_End_Timer, Validate_End_Timer, GCKS_Down_Timer, GCKS2_Down_Timer, GCKS2_End_Timer, GCKS_Anno_Timer, GCKS2_Anno_Timer. Initial_Anno_Timer fires to trigger sending of an INITIAL_ANNOUNCEMENT based on Initial_Anouncement_Interval. Initial_End_Timer fires to trigger the transition of a router state from Initial to some other state. Validate_End_Timer fires to trigger the transition of a router state from Validate to GCKS2. GCKS_Down_Timer fires when no GCKS_ANNOUNCEMENT has been heard for GCKS_Down_Interval. GCKS2_Down_Timer fires when no GCKS2_ANNOUNCEMENT has not been heard for GCKS2_Down_Interval. GCKS2_End_Timer fires to trigger the transition of the state of a router from GCKS2 to GCKS. GCKS_Anno_Timer fires to trigger sending of a GCKS_ANNOUNCEMENT based on GCKS_Anno_Interval. GCKS2_Anno_Timer fires to trigger sending of a GCKS2_ANNOUNCEMENT based on GCKS2_Anno_Interval.

During an election process, a MaRK router may have to deal with following types of events:

- **X_Anno_Received**: an X_ANNOUNCEMENT is received.

- **Requester_Validated**: have authenticated and validated against a some router who believes we should be a GCKS or GCKS2.

- **GCKS_Validate**: a remote entity has been authenticated and validated to be a GCKS router.

- **GCKS2_Validate**: a remote entity has been authenticated and validated to be a GCKS2 router.
Referral_Validated: have authenticated and validated against a candidate who is not a GCKS router but knows one is.

Referral2_Validated: have authenticated and validated against a candidate who knows a GCKS2 router.

Authentication/Validation_Failed: the remote entity fails in the authentication or cannot be either a GCKS/GCKS2 or a referral.

X_Timer_Expired: the timer of type X expired.

KEK_Expired: we have no valid KEK.

3.1.2. Initial

The timers utilized in this state are Initial_Anno_Timer and Initial_End_Timer.

On entry:

- Send an INITIAL_ANNOUNCEMENT.

- Set the Initial_Anno_Timer with Initial_Anno_Interval.

- Set the Initial_End_Timer with Initial_End_Interval.

Events:

- Initial_Anno_Timer_Expired: send an INITIAL_ANNOUNCEMENT and reset the Initial_Anno_Timer.

- Initial_Anno_Received: if the sender of the announcement is more preferred, add the entity into the candidate list; if less preferred, send an INITIAL_ANNOUNCEMENT with a limited rate.

- GCKS_Anno_Received: add the sender of the announcement to the candidate list; set the the Validate_End_Timer with the remaining period of Initial_End_Interval; transfer to validate.

- GCKS2_Anno_Received: add the sender of the announcement to candidate list; set the Validate_End_Timer with the remaining period of Initial_End_Interval; transfer to validate.

- Requester_Validated: If the requester is looking for a GCKS router and the local policy permits, transfer the state to GCKS2 and set GCKS2_End_Interval to time remaining on Initial_End_timer.
3.1.3. Validate

The timer utilized in this state is Validate_End_Timer.

Entering this state means that there is a router which potential could be a GCKS. The purpose of this state is to confirm that it is able to establish a security association with that router and that router’s policy permits it to be a GCKS for this group. The two normal paths through the state machine are Initial leading to GCKS for the most preferred router and Initial leading to Validate leading to Member for other routers.

On entry:

- Authenticate and validate the most preferred entry in the candidate list.
- If Validate_End_timer has more time than Validate_end_Interval, set Validate_End_timer to Validate_End_interval.

Events:

- GCKS_Validated: transfer the state to Member.
- GCKS2_Validated: Transfer the state to Follower.
- Referral_Validated: perform the authentication/validation on the recommended node; move the referring from the candidate list to the black list for Blacklist_Interval.
- Referral2_Validated: perform the authentication/validation on the recommended node; move the referring node from the candidate list to the black list for Blacklist_Interval.
- Requester_Validated: If the requester is looking for a GCKS/GCKS2 router and the local policy permits, transfer the state to GCKS2.
- Validation_Failed: move the router being validated from the candidate list to black list for Blacklist_interval.
- Initial_Announcement_Received: if the sender of the announcement is more preferred, add the router into the candidate list; if less preferred, send an INITIAL_ANNOUNCEMENT with a limited rate.
- GCKS_Anno_Received: add the router sending the announcement into the candidate list and perform authentication against that entity.
- GCKS2_Anno_Received: add the router sending the announcement into the candidate list and start the authentication/validation against that entity.
- Validate_End_Timer_Expired: transfer the state to GCKS2.

3.1.4. GCKS2

The timers utilized in this state include GCKS2_Anno_Timer and GCKS2_End_Timer.

When a router transfers its state from Validate to GCKS2, it is indicated that there has been some authentication/validation problem or another node is behaving in a manner inconsistent with the election state. In this case, the purpose of the GCKS2 state is to establish sufficient security keys to integrity protect the election process. In addition, it is possible for a router to enter this state during normal operations if the router being elected GCKS gets an authentication request before Initial_End_timer expires. In this case, the router will transfer its state to GCKS if no more preferred GCKS candidate is found within a limited period.

On entry:
- Send an GCSK2_ANNOUNCEMENT.
- Set the GCKS2_Anno_Timer with GCKS2_Anno_Interval.
- Set the the GCKS2_End_Timer with GCKS2_End_Interval unless it was set on entry transferring from Initial.

Events:
- GCKS_Anno_Received: add to candidate list; start authentication/validation.
- GCKS2_Anno_Received: if more preferred, add to candidate list, start authentication/validation. If less preferred, send GCKS2_ANNOUNCEMENT if rate limiting is permitted.
- GCKS_Validated: Transfer to member state; flood KEK to the associated followers.
- GCKS2_Validated: Transfer the state to Follower; flood KEK to the associated followers.
- Referral_Validated: Perform authentication and validation on the recommended node; move the referring node from the candidate list to the black list for Blacklist_Interval.

- Referral2_Validated: if the recommended GCKS2 is more preferred, perform authentication and validation on the recommended node; move the referring from the candidate list to the black list for Blacklist_Interval.

- Requester_Validated: if the requester is looking for a GCKS2, distribute KEK.

- Validation_Failed: move the router being validated from the candidate list to black list for Blacklist_interval.

- GCKS2_End_Timer_Expired: transition the state to GCKS.

- GCKS2_Anno_Timer_Expired: send a GCKS2_ANNOUNCEMENT.

3.1.5. GCKS

The timer utilized in this state is GCKS_Anno_Timer.

On entry:

- Send a GCKS_ANNOUNCEMENT.

- Set the GCKS_Anno_Timer with GCKS_Anno_Interval.

- Generate protocol keys; if needed, generate KEK.

Events:

- GCKS_Anno_Timer_Expired: send a GCKS_ANNOUNCEMENT.

- Initial_Anno_Received: send an GCKS_ANNOUNCEMENT immediately if the rate limiting is permitted.

- GCKS2_Anno_Received: send an GCKS_ANNOUNCEMENT immediately if the rate limiting is permitted.

- GCKS_Anno_Received: if the sender is more preferred, add to candidate list and start authentication/validation; Otherwise, send an GCKS_ANNOUNCEMENT immediately if the rate limiting is permitted.

- GCKS_Validated: start network merging operations as what is illustrated in Section 3.2.
3.1.6. Member

The timer utilized in this state is GCKS_Down_Timer.

On entry:

- Set the GCKS_Down_Timer with GCKS_Down_Interval.

Events:

- GCKS_Down_Timer_Expired: Transfer the state into Initial.
- GCKS_Anno_Received: reset GCKS_Down_Timer.
- Requester_Validated: if the requester is legal, recommend the GCKS router to it.

3.1.7. Follower

The timer utilized in this state is GCKS2_Down_Timer.

On entry:

- Set the GCKS2_Down_Timer with GCKS2_Down_Interval.

Events:

- GCKS2_Down_Timer_Expired: Transfer the state into Initial.
- GCKS2_Anno_Received: reset GCKS2_Down_Timer.
- GCKS_Anno_Received: Add the announcer to the candidate list and start validation.
- Requester_Validated: if the requester is legal, recommend the GCKS2 router to it.
- GCKS_Validated: Transfer the state to member.

The following diagram illustrates the rules of transiting the states introduced this section.
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 router joins the group, 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 mark_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 and retransmit the message for several times according to its local policy. 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.

3.3. Operations on Receiving a Packet

When a router attempts to join an election process, it may have a valid KEK. For instance, when a GCKS cannot work properly, the routers on the link need to transfer their state to Initial and raise an election to find a new valid GCKS. If there is still a valid KEK shared by the router, they can use the KEK to secure the packets transmitted during the election until a new KEK is distributed by the new GCKS. A router holding the valid KEK is regarded to be more preferred than a router which doesn’t have the key. By using the KEK, it is able to prevent an attacker from disturbing the election process by broadcasting fake announcements. Therefore, after an initial router does not find any more preferred router holding the valid key, it then can transfer its state to GCKS directly.

Therefore, the operations on receiving a packet are as follows:

- Check the blacklist. If the sender of the packet is on the blacklist, discard the packet.
- If the state is GCKS, accept the packet and generate an event. GCKS announcements need to be excepted in GCKS state for merges to work.
- If there is a KEK that is not expired, check the packet integrity against any matching KEK.
- If no KEK matches or if the integrity fails to validate, discard the packet.
- If there is no KEK at all or the KEK integrity check passed, process the packet and generate an event.

It is notable this approach limits the scope of the election within
the routers managed by the failed GCKS. If there are routers newly accessing the link during the election, no router with a KEK will process their packets. However these routers can process packets from routers with the KEK. In many cases one of the routers with a KEK will be elected GCKS and the other routers can authenticate and join. In the worst case, two independent GCKSes will be elected and then merge.

4. Key Download Payload

What all is actually in the message you get at the end of phase 1 exchange (the RP_AUTH Exchange) 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 transmit start time, receive start time, and expire time.

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

5. Initial Exchanges Details

Similar with [I-D.tran-karp-mrmp], in MRKMP, when two routers needs to authenticate each other, they need to perform the initial exchanges defined in [I-D.mahesh-karp-rkmp]. For example, when a router intends to join a group, it needs to firstly perform a RP Initial (RP_INIT) Exchange with the GCKS of the group. RP_INIT is identical to the IKE_SA_INIT exchange defined in Internet Key Exchange Protocol Version 2 [RFC5996], after which the router and the GCKS can communicate privately. Note that at this point the network devices have not identified their peer. For the details of this exchange, refer to IKE_SA_INIT in Internet Key Exchange Protocol Version 2 [RFC5996].

<table>
<thead>
<tr>
<th>Router</th>
<th>GCKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR, SAi1, KEi, Ni</td>
<td>HDR, SAr1, KEr, Nr, [CERTREQ,]</td>
</tr>
</tbody>
</table>

The router and the GCKS then needs to perform an RP_AUTH exchange defined in [I-D.mahesh-karp-rkmp]. At the successful conclusion of the exchange, the router is adopted as a group member and obtains keying material (e.g., the KEK and protocol master key) to securely
communicate with other group members.

<table>
<thead>
<tr>
<th>Router</th>
<th>GCKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>HDR, SK {IDi, [CERT,] [CERTREQ,] [IDr,] AUTH, SArpi} --&gt;</td>
<td>HDR, SK {IDr, [CERT,] AUTH, SArp}</td>
</tr>
<tr>
<td></td>
<td>RP_AUTH</td>
</tr>
</tbody>
</table>

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 MaRK_WRONG_GCKS including the address of the GCKS or GCKS2 in the notification payload along with an indication of whether the router is a GCKS or GCKS2. 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 MaRK_GCKS_UNKNOWN notification.

7. Group Key Management Operation

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

A member of a group can also use the unicast exchange to request a GCKS to change a protocol master key, on the occasions, for example, where the member is going to exhaust its sequence space of the associated routing protocol. 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.

7.3. Changing the Active GCKS

When a GCKS finds a more preferred router announcing itself as a GCKS, it will forward its privilege to another one in the following conditions. The operations are introduced in Section 3.2.

When a GCKS cannot work properly, it will just stop sending the GCKS_ANNOUNCEMENT. Then after a certain time period, a new GCKS election process will be raised.

7.4. Reboot Cases

After a reboot, a router in a group will lost the state information about the group (e.g., protocol master keys, traffic keys, the sequence numbers used by GCKS). Therefore, the router needs to find and authenticate the GCKS, and apply to join the group. If the GCKS finds that the router is already a group member, the GCKS will update the transport keys (and the protocol master keys if necessary) used in the group first in order to avoid inter-session replay attacks.

8. Interface to Routing Protocol

This section describes signaling between MaRK and the routing protocol. The primary communication between these protocols is that MaRK 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 MaRK. This section discusses that signaling. All required communication from MaRK to the routing protocol can be accomplished by manipulating the key table. However an implementation MAY wish to signal MaRK 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 MaRK. This event includes the identity of the group as well as this router’s priority for being a GCKS for the group. When MaRK receives this event, it starts MaRK 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, MaRK implementation on the router needs to be notified that group is no longer joined. MaRK 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.

8.4.  Out of Sequence Space

If a routing protocol is running out its sequence space, the MaRK implementation on the router needs to be notified. The MaRK implementation then needs to contact the GCKS to request the update of the transport keys (and the protocol master keys if necessary).

9.  Security Considerations

This protocol is intended to protect against attackers who are not properly authorized mounting an integrity or availability attack on the system. All parties who are authorized to be part of a given group are equivalent; group members impersonating each other, impacting availability or integrity are all out of scope for this threat model. Protecting confidentiality of key material against parties not authorized for membership in a given group is in scope as it would directly lead to an attack on integrity or availability.
Protecting confidentiality of group policy or routing data is not required. Attackers are assumed to be able to insert and observe packets. Even if attackers can modify and suppress packets, integrity should not be impacted. Minimizing the availability impact against attackers who can modify and suppress packets is strongly desirable, although there are limits to this defense. It is important that a member of one group not be able to impact another group.

Significant complexity results from the election protocol. In order to support arbitrary authentication mechanisms including preshared keys, the election protocol itself is not signed. At least before group keys are established, the election protocol is not integrity protected. Later authentication can establish integrity, but managing availability attacks on the election protocol requires significant analysis.

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 election protocol has been designed to try and resist denial of service conditions. However, the election protocol maintains state in the form of a candidate list and black list. An attacker can consume state by generating fake election announcements. An implementation can discard state if it has insufficient resources. However, if legitimate routers are discarded from the candidate list, the protocol may take longer to converge or may not converge. If entries are removed from the black list, then more resources may be spent on attackers. So the solution has some residual denial of service possibilities. The election protocol requires significant analysis to confirm it meets its design goals.

The security of the election protocol depends on the denial of service resistance of the authentication protocol. It is important that an attacker not be able to cause an authentication to fail by injecting a packet. So, rather than failing an authentication if a bad packet is received, an implementation needs to wait and see if a good packet appears in some timeout.

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

The funding for Sam Hartman’s work on this document is provided by Huawei.

XXX add the list of people in the lunch time group unless they are willing to be listed as authors.

11. References

11.1. Normative References


11.2. Informative References


[I-D.yeung-g-ikev2] Rowles, S., Yeung, A., Tran, P., and Y. Nir, "Group Key
Management using IKEv2", draft-yeung-q-ikev2-04 (work in progress), March 2012.


Authors’ Addresses

Sam Hartman
Painless Security
Email: hartmans-ietf@mit.edu

Dacheng Zhang
Huawei Technologies co. ltd
Huawei Building No.3 Xinxi Rd., Shang-Di Information Industrial Base Hai-Dian District, Beijing China
Email: zhangdacheng@huawei.com

Gregory Lebovitz
Juniper Networks, Inc.
1194 North Mathilda Ave.
Sunnyvale, California  94089-1206
USA
Email: gregory.ietf@gmail.com