

Network Working Group	R. Singh, Ed.
Internet-Draft	G. Kalyani
Intended status: Standards Track	Cisco
Expires: September 30, 2011	Y. Nir
	Check Point
	Y. Sheffer
	Independent
	D. Zhang
	Huawei
	March 29, 2011

Protocol Support for High Availability of IKEv2/IPsec
draft-ietf-ipsecme-ipsecha-protocol-05

Abstract

The IPsec protocol suite is widely used for business-critical network traffic. In order to make IPsec deployments highly available, more scalable and failure-resistant, they are often implemented as IPsec High Availability (HA) clusters. However there are many issues in IPsec HA clustering, and in particular in IKEv2 clustering. An earlier document, "IPsec Cluster Problem Statement", enumerates the issues encountered in the IKEv2/IPsec HA cluster environment. This document resolves these issues with the least possible change to the protocol. This document defines an extension to the IKEv2 protocol to solve the main issues of "IPsec Cluster Problem Statement" in the commonly deployed hot-standby cluster, and provides implementation advice for other issues. The main issues solved are the synchronization of IKEv2 Message ID counters, and of IPsec Replay Counters.

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 September 30, 2011.

Copyright Notice

Copyright (c) 2011 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

[Table of Contents](#)

- *1. [Introduction](#)
- *2. [Terminology](#)
- *3. [Issues Resolved from IPsec Cluster Problem Statement](#)
 - *3.1. [Large Amount of State](#)
 - *3.2. [Multiple Members Using the Same SA](#)
 - *3.3. [Avoiding Collisions in SPI Number Allocation](#)
 - *3.4. [Interaction with Counter Modes](#)
- *4. [The IKEv2/IPsec SA Counter Synchronization Problem](#)
- *5. [SA Counter Synchronization Solution](#)
 - *5.1. [Processing Rules for IKE Message ID Synchronization](#)
 - *5.2. [Processing Rules for IPsec Replay Counter Synchronization](#)
- *6. [IKEv2/IPsec Synchronization Notification Payloads](#)
 - *6.1. [The IKEV2 MESSAGE ID SYNC SUPPORTED Notification](#)
 - *6.2. [The IPSEC_REPLAY_COUNTER SYNC SUPPORTED Notification](#)
 - *6.3. [The IKEV2 MESSAGE ID SYNC Notification](#)
 - *6.4. [The IPSEC_REPLAY_COUNTER SYNC Notification](#)
- *7. [Implementation Details](#)
- *8. [IKE SA and IPsec SA Message Sequencing](#)
 - *8.1. [Handling of Pending IKE Messages](#)
 - *8.2. [Handling of Pending IPsec Messages](#)

- *8.3. [IKE SA Inconsistencies](#)
- *9. [Step by Step Details](#)
- *10. [Interaction with other drafts](#)
- *11. [Security Considerations](#)
- *12. [IANA Considerations](#)
- *13. [Acknowledgements](#)
- *14. [Change Log](#)
 - *14.1. [Draft -04](#)
 - *14.2. [Draft -03](#)
 - *14.3. [Draft -02](#)
 - *14.4. [Draft -01](#)
 - *14.5. [Draft -00](#)
- *15. [References](#)
 - *15.1. [Normative References](#)
 - *15.2. [Informative References](#)
- *Appendix A. [IKEv2 Message ID Sync Examples](#)
 - *Appendix A.1. [Normal Failover - Example 1](#)
 - *Appendix A.2. [Normal Failover - Example 2](#)
 - *Appendix A.3. [Simultaneous Failover](#)
- *[Authors' Addresses](#)

1. Introduction

The IPsec protocol suite, including IKEv2, is a major building block of virtual private networks (VPNs). In order to make such VPNs highly available, more scalable and failure-resistant, these VPNs are implemented as IKEv2/IPsec Highly Available (HA) clusters. However there are many issues with the IKEv2/IPsec HA cluster. [Section 4](#) below enumerates the issues around the IKEv2/IPsec HA cluster solution. In the case of a hot-standby cluster implementation of IKEv2/IPsec based VPNs, the IKEv2/IPsec session is first established between the peer and the active member of the cluster. Later, the active member

continuously syncs/updates the IKE/IPsec SA state to the standby member of the cluster. This primary SA state sync-up takes place upon each SA bring-up and/or rekey. Performing the SA state synchronization/update for every single IKE and IPsec message is very costly, so normally it is done periodically. As a result, when the failover event happens, this is first detected by the standby member and, possibly after a considerable amount of time, it becomes the active member. During this failover process the peer is unaware of the failover event, and keeps sending IKE requests and IPsec packets to the cluster, as in fact it is allowed to do because of the IKEv2 windowing feature. After the newly-active member starts, it detects the mismatch in IKE Message ID values and IPsec replay counters and needs to resolve this situation. Please see [Section 4](#) for more details of the problem.

This document proposes an extension to the IKEv2 protocol to solve the main issues of IKE Message ID synchronization and IPsec SA replay counter synchronization and gives implementation advice for others. Following is a summary of the solutions provided in this document:

- *IKEv2 Message ID synchronization: this is done by syncing up the expected send and receive Message ID values with the peer, and updating the values at the newly active cluster member.

- *IPsec Replay Counter synchronization: this is done by incrementing the cluster's outgoing SA replay counter values by a "large" number; in addition, the newly-active member requests the peer to increment the replay counter values it is using for the peer's outgoing traffic.

Although this document describes the IKEv2 Message ID and IPsec replay counter synchronization in the context of an IPsec HA cluster, the solution provided is generic and can be used in other scenarios where IKEv2 Message ID or IPsec SA replay counter synchronization may be required.

Implementations differ on the need to synchronize the IKEv2 Message ID and/or IPsec replay counters. Both of these problems are handled separately, using a separate notification for each capability. This provides the flexibility of implementing either or both of these solutions.

2. Terminology

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 [\[RFC2119\]](#).

"SA Counter Synchronization Request/Response" are the request viz. response of the informational exchange defined in this document to synchronize the IKEv2/IPsec SA counter information between one member of the cluster and the peer.

Some of the terms listed below are reused from [\[RFC6027\]](#) with further clarification in the context of the current document.

*"Hot Standby Cluster", or "HS Cluster" is a cluster where only one of the members is active at any one time. This member is also referred to as the "active" member, whereas the other(s) are referred to as "standby" members. VRRP [\[RFC5798\]](#) is one method of building such a cluster. The goal of the Hot Standby Cluster is to create the illusion of a single virtual gateway to the peer(s).

*"Active Member" is the primary member in the Hot-Standby cluster. It is responsible for forwarding packets on behalf of the virtual gateway.

*"Standby Member" is the primary backup member. This member takes control, i.e. becomes the active member, after the failover event.

*"Peer" is an IKEv2/IPsec endpoint that maintains an IPsec connection with the Hot-Standby cluster. The Peer identifies the cluster by the cluster's (single) IP address. If a failover event occurs, the standby member of the cluster becomes active, and the peer normally doesn't notice that failover has taken place. Although we treat the peer as a single entity, it may also be a cluster.

*"Multiple failover" is the situation where, in a cluster with three or more members, multiple failover events happen in rapid succession, e.g. from M1 to M2, and then to M3. It is our goal that the implementation should be able to handle this situation, i.e. to handle the new failover event even if it is still processing the old failover.

*"Simultaneous failover" is the situation where two clusters have an IPsec connection between them, and failover happens at both ends at the same time. It is our goal that implementations should be able to handle simultaneous failover.

The generic term "IKEv2/IPsec SA Counters" is used throughout this document. This term refers to both IKEv2 Message ID counters and IPsec replay counters. According to the IPsec standards, the IKEv2 Message ID counter is mandatory, and used to ensure reliable delivery as well as to protect against message replay in IKEv2; the IPsec SA replay counters are optional, and are used to provide the IPsec anti-replay feature.

3. Issues Resolved from IPsec Cluster Problem Statement

The IPsec Cluster Problem Statement [\[RFC6027\]](#) enumerates the problems raised by IPsec clusters. The following table lists the problem statement's sections that are resolved by this document.

- *3.2. Lots of Long Lived State
- *3.3. IKE Counters
- *3.4. Outbound SA Counters
- *3.5. Inbound SA Counters
- *3.6. Missing Synchronization Messages
- *3.7. Simultaneous use of IKE and IPsec SAs by Different Members
 - 3.7.1. Outbound SAs using counter modes
- *3.8. Different IP addresses for IKE and IPsec
- *3.9. Allocation of SPIs

The main problem areas are solved using the protocol extension defined below, starting with [Section 5](#); additionally, this section provides implementation advice for other issues in the following subsections.

3.1. Large Amount of State

Section 3.2 of the Problem Statement mentions that a lot of state needs to be synchronized for a cluster to be transparent. The actual volume of that data is very much implementation-dependent, and even for the same implementation, the amounts of data may vary wildly. An IPsec gateway used for inter-domain VPN with a dozen other gateways, and having SAs that are rekeyed every 8 hours, will need a lot less synchronization traffic than a similar gateway used for remote access, and supporting 10,000 clients. This is because counter synchronization is proportional to the number of SAs and requires little data, and the setting up of an SA requires a lot of data. Additionally, remote access IKE and IPsec SA setup tend to happen at a particular time of day, so the example gateway with the 10,000 clients may see 30-50 IKE SA setups per second at 9:00 AM. This would require very heavy synchronization traffic over that short period of time.

If a large volume of traffic is necessary, it may be advisable to use a dedicated high-speed network interface for synch traffic. When packet loss can be made extremely low, it may be advisable to use a stateless transport such as UDP, to minimize network overhead.

If these methods are insufficient, it may be prudent that for some SAs the entire state is not synchronized. Instead, only an indication of

the SA's existence is synchronized. This, in combination with a sticky solution (as described in section 3.7 of the problem statement) ensures that the traffic from a particular peer does not reach a different member before an actual failover happens. When that happens, the method described in [\[I-D.ietf-ipsecme-failure-detection\]](#) can be used to quickly force the peer to set up a new SA.

[3.2. Multiple Members Using the Same SA](#)

In a load-sharing cluster of the "duplicate" variety (see section 3.7 of the problem statement) multiple members may need to send traffic with the same selectors. To actually use the same SA the cluster would have to synchronize the Replay Counter after every packet, and that would impose unreasonable requirements on the synch connection. A far better solution would be to not synchronize the outbound SA, and create multiple outbound SAs, one for each member. The problem with this option is that the peer might view these multiple parallel SAs as redundant, and tear down all but one of them. Section 2.8 of [\[RFC5996\]](#) specifically allows multiple parallel SAs, but the reason given for this is to have multiple SAs with different QoS attributes. So while this is not a new requirement of IKEv2 implementations, we re-iterate here that IPsec peers MUST accept the long-term existence of multiple parallel SAs, even when QoS mechanisms are not in use.

[3.3. Avoiding Collisions in SPI Number Allocation](#)

Section 3.9 of the problem statement describes the problem of two cluster members allocating the same SPI number for two different SAs. This would violate section 4.4.2.1 of [\[RFC4301\]](#). There are several schemes to allow implementations to avoid such collisions, such as partitioning the SPI space, a request-response over the synch channel, and locking mechanisms. We believe that these are sufficiently robust and available so that we don't need to make an exception to RFC 4301, and we can leave this problem for the implementations to solve. Cluster members MUST NOT generate multiple inbound SAs with the same SPI.

[3.4. Interaction with Counter Modes](#)

For SAs involving counter mode ciphers such as CTR [\[RFC3686\]](#) or GCM [\[RFC4106\]](#) there is yet another complication. The initial vector for such modes MUST NOT be repeated, and senders use methods such as counters or LFSRs to ensure this property. For an SA shared between multiple active members (load sharing cases), implementations MUST ensure that no initial vector is ever repeated. Similar concerns apply to an SA failing over from one member to another. See [\[RFC6054\]](#) for a discussion of this problem in another context. Just as in the SPI collision problem, there are ways to avoid a collision of initial vectors, and this is left up to implementations.

In the context of load sharing, parallel SAs are a simple solution to this problem as well.

4. The IKEv2/IPsec SA Counter Synchronization Problem

The IKEv2 protocol [\[RFC5996\]](#) states that "An IKE endpoint MUST NOT exceed the peer's stated window size for transmitted IKE requests". All IKEv2 messages are required to follow a request-response paradigm. The initiator of an IKEv2 request MUST retransmit the request, until it has received a response from the peer. IKEv2 introduces a windowing mechanism that allows multiple requests to be outstanding at a given point of time, but mandates that the sender's window should not move until the oldest message it has sent is acknowledged. Loss of even a single message leads to repeated retransmissions followed by an IKEv2 SA teardown if the retransmissions remain unacknowledged.

An IPsec Hot Standby Cluster is required to ensure that in the case of failover, the standby member becomes active immediately. The standby member is expected to have the exact value of the Message ID counter as the active member had before failover. Even assuming the best effort to update the Message ID values from active to standby member, the values at the standby member can still be stale due to the following reasons:

- *The standby member is unaware of the last message that was received and acknowledged by the previously active member, as the failover event could have happened before the standby member could be updated.
- *The standby member does not have information about on-going unacknowledged requests sent by the previously active member. As a result after the failover event, the newly active member cannot retransmit those requests.

When a standby member takes over as the active member, it can only initialize the Message ID values from the previously updated values. This would make it reject requests from the peer when these values are stale. Conversely, the standby member may end up reusing a stale Message ID value which would cause the peer to drop the request. Eventually there is a high probability of the IKEv2 and corresponding IPsec SAs getting torn down simply because of a transitory Message ID mismatch and retransmission of requests, negating the benefits of the high availability cluster despite the periodic update between the cluster members.

A similar issue is also observed with IPsec anti-replay counters if anti-replay protection is enabled, which is commonly the case. Regardless of how well the ESP and AH SA counters are synchronized from the active to the standby member, there is a chance that the standby member would end up with stale counter values. The standby member would then use those stale counter values when sending IPsec packets. The peer would reject/drop such packets since when the anti-replay

protection feature is enabled, duplicate use of counters is not allowed. Note that IPsec allows the sender to skip some counter values and continue sending with higher counter values. We conclude that a mechanism is required to ensure that the standby member has correct Message ID and IPsec counter values when it becomes active, so that sessions are not torn down as a result of mismatched counters.

5. SA Counter Synchronization Solution

This document proposes two separate approaches to resolving the issues of mismatched IKE Message ID values and IPsec counter values.

*In the case of IKE Message ID values, the newly active cluster member and the peer negotiate a pair of new values so that future IKE messages will not be dropped.

*For IPsec counter values, the newly-active member and the peer both increment their respective counter values, "skipping forward" by a large number, to ensure that no IPsec counters are ever reused.

Although conceptually separate, the two synchronization processes would typically take place simultaneously.

First, the peer and the active member of the cluster negotiate their ability to support IKEv2 Message ID synchronization and/or IPsec Replay Counter synchronization. This is done by exchanging one or both of the `IKEV2_MESSAGE_ID_SYNC_SUPPORTED` and `IPSEC_REPLAY_COUNTER_SYNC_SUPPORTED` notifications during the `IKE_AUTH` exchange. When negotiating these capabilities, the responder **MUST NOT** assert support of a capability unless such support was asserted by the initiator. Only a capability whose support was asserted by both parties can be used during the lifetime of the SA.

This per-IKE SA information is shared with the other cluster members.

Peer	Active Member
<hr/>	
HDR, SK {IDi, [CERT], [CERTREQ], [IDr], AUTH, [N(IKEV2_MESSAGE_ID_SYNC_SUPPORTED),] [N(IPSEC_REPLAY_COUNTER_SYNC_SUPPORTED),] SAi2, TSi, TSr} ----->	
 <----- HDR, SK {IDr, [CERT+], [CERTREQ+], AUTH, [N(IKEV2_MESSAGE_ID_SYNC_SUPPORTED),] [N(IPSEC_REPLAY_COUNTER_SYNC_SUPPORTED),] SAR2, TSi, TSr}	

After a failover event, the standby member MAY use the IKE Message ID and/or IPsec Replay Counter synchronization capability when it becomes the active member, and provided support for the capabilities used has been negotiated. Following that, the peer MUST respond to any synchronization message it receives from the newly-active cluster member, subject to the rules noted below.

After the failover event, when the standby member becomes active, it has to synchronize its SA counters with the peer. There are now three possible cases:

1. The cluster member wishes to only perform IKE Message ID value synchronization. In this case it initiates an Informational exchange, with Message ID zero and the sole notification IKEV2_MESSAGE_ID_SYNC.
2. If the newly-active member wishes to perform only IPsec replay counter synchronization, it generates a regular IKEv2 Informational exchange using the current Message ID values, and containing the IPSEC_REPLAY_COUNTER_SYNC notification.
3. If synchronization of both counters is needed, the cluster member generates a zero-Message ID message as in case #1, and includes both notifications in this message.

This figure contains the IKE message exchange used for SA counter synchronization. The following subsections describe the details of the sender and receiver processing of each message.

Standby [Newly Active] Member	Peer
- - - - -	
HDR, SK {N(IKEV2_MESSAGE_ID_SYNC), [N(IPSEC_REPLAY_COUNTER_SYNC)]} ----->	<----- HDR, SK {N(IKEV2_MESSAGE_ID_SYNC)}

Alternatively, if only IPsec Replay Counter synchronization is desired, a normal Informational exchange is used, where the Message ID is non-zero:

Standby [Newly Active] Member	Peer

HDR, SK{N(IPSEC_REPLAY_COUNTER_SYNC)} ----->	
<----- HDR	

5.1. Processing Rules for IKE Message ID Synchronization

The newly-active member sends a request containing two counter value, one for the member (itself) and another for the peer, as well as a random nonce. We denote the values M1 and P1. The peer responds with a message containing two counter values, M2 and P2. The goal of the rules below is to prevent an attacker from replaying a synchronization message, thereby invalidating IKE messages that are currently in process.

- *M1 is the next sender's Message ID to be used by the member. M1 MUST be chosen so that it is larger than any value known to have been used. It is RECOMMENDED to increment the known value at least by the size of the IKE sender window.
- *P1 SHOULD be 1 more than the last Message ID value received from the peer, but may be any higher value.
- *The member SHOULD communicate the sent values to the other cluster members, so that if a second failover event takes place, the synchronization message is not replayed. Such a replay would result in the eventual deletion of the IKE SA (see below).
- *The peer MUST reject any received synchronization message if M1 is lower than or equal to the highest value it has seen from the cluster. This includes any previous received synchronization messages.
- *M2 MUST be at least the higher of the received M1, and one more than the highest sender value received from the cluster. This includes any previous received synchronization messages.
- *P2 MUST be the higher of the received P1 value, and one more than the highest sender value used by the peer.
- *The request contains a Nonce field. This field MUST be returned in the response, unchanged. A response MUST be silently dropped if the received Nonce does not match the one that was sent.

*Both the request and the response MUST NOT contain any additional payloads, other than an optional IPSEC_REPLAY_COUNTER_SYNC notification in the request.

*The request and the response MUST both be sent with a Message ID value of zero.

5.2. Processing Rules for IPsec Replay Counter Synchronization

Upon failover, the newly-active member MUST increment its own Replay Counter (the counter used for outgoing traffic), so as to prevent the case of its traffic being dropped by the peer as replay. We note that IPsec allows the replay counter to skip forward by any amount. The estimate is based on the outgoing IPsec bandwidth and the frequency of synchronization between cluster members. In those implementations where it is difficult to estimate this value, the counter can be incremented by a very large number, e.g. 2^{30} . In the latter case, a rekey SHOULD follow shortly afterwards, to ensure that the counter never wraps around.

Next, the cluster member estimates the number of incoming messages it might have missed, using similar logic. The member sends out a IPSEC_REPLAY_COUNTER_SYNC notification, either stand-alone or together with a IKEV2_MESSAGE_ID_SYNC notification.

If the IPSEC_REPLAY_COUNTER_SYNC is included in the same message as IKEV2_MESSAGE_ID_SYNC, the peer MUST process the Message ID notification first (which might cause the entire message to be dropped as a replay). Then, it MUST increment the replay counters for all Child SAs associated with the current IKE SA by the amount requested by the cluster member.

6. IKEv2/IPsec Synchronization Notification Payloads

This section lists the new notification payload types defined by this extension.

6.1. The IKEV2_MESSAGE_ID_SYNC_SUPPORTED Notification

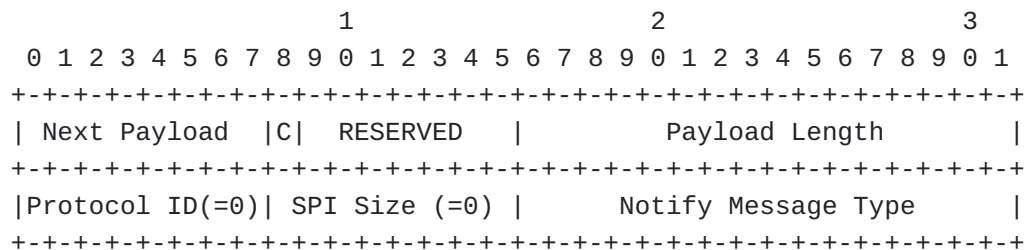
This notification payload is included in the IKE_AUTH request/response to indicate support of the IKEv2 Message ID synchronization mechanism described in this document.

1										2										3											
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Next Payload (C)										RESERVED										Payload Length											
Protocol ID(=0)										SPI Size (=0)										Notify Message Type											

The 'Next Payload', 'Payload Length', 'Protocol ID', 'SPI Size', and 'Notify Message Type' fields are the same as described in Section 3 of [\[RFC5996\]](#). The 'SPI Size' field MUST be set to 0 to indicate that the SPI is not present in this message. The 'Protocol ID' MUST be set to 0, since the notification is not specific to a particular security association. The 'Payload Length' field is set to the length in octets of the entire payload, including the generic payload header. The 'Notify Message Type' field is set to indicate IKEV2_MESSAGE_ID_SYNC_SUPPORTED, value TBD by IANA. There is no data associated with this notification.

6.2. The IPSEC_REPLAY_COUNTER_SYNC_SUPPORTED Notification

This notification payload is included in the IKE_AUTH request/response to indicate support for the IPsec SA Replay Counter synchronization mechanism described in this document.



The 'Next Payload', 'Payload Length', 'Protocol ID', 'SPI Size', and 'Notify Message Type' fields are the same as described in Section 3 of [\[RFC5996\]](#). The 'SPI Size' field MUST be set to 0 to indicate that the SPI is not present in this message. The 'Protocol ID' MUST be set to 0, since the notification is not specific to a particular security association. The 'Payload Length' field is set to the length in octets of the entire payload, including the generic payload header. The 'Notify Message Type' field is set to indicate IPSEC_REPLAY_COUNTER_SYNC_SUPPORTED, value TBD by IANA. There is no data associated with this notification.

6.3. The IKEV2_MESSAGE_ID_SYNC Notification

This notification payload type (value TBD by IANA) is defined to synchronize the IKEv2 Message ID values between the newly-active (formerly standby) cluster member and the peer.

1																2																3																
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1																	
Next Payload																C	RESERVED																Payload Length															
Protocol ID(=0)																SPI Size (=0)																Notify Message Type																
Nonce Data																																																
EXPECTED_SEND_REQ_MESSAGE_ID																																																
EXPECTED_RECV_REQ_MESSAGE_ID																																																

It contains the following data.

*Nonce Data (4 octets): the random nonce data. The data should be identical in the synchronization request and response.

*EXPECTED_SEND_REQ_MESSAGE_ID (4 octets): this field is used by the sender of this notification payload to indicate the Message ID it will use in the next request that it will send to the other protocol peer.

*EXPECTED_RECV_REQ_MESSAGE_ID (4 octets): this field is used by the sender of this notification payload to indicate the Message ID it is expecting in the next request to be received from the other protocol peer.

6.4. The IPSEC_REPLAY_COUNTER_SYNC Notification

This notification payload type (value TBD by IANA) is defined to synchronize the IPsec SA Replay Counters between the newly-active (formerly standby) cluster member and the peer. Since there may be numerous IPsec SAs established under a single IKE SA, we do not directly synchronize the value of each one. Instead, a delta value is sent and all Replay Counters for Child SAs of this IKE SA are incremented by the same value. Note that this solution requires that all these Child SAs either use or do not use Extended Sequence Numbers [\[RFC4301\]](#). This notification is only sent by the cluster.

1																2																3															
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1																
Next Payload C																RESERVED																Payload Length															
Protocol ID(=0)																SPI Size (=0)																Notify Message Type															
Incoming IPsec SA delta value																																															

The notification payload contains the following data.

*Incoming IPsec SA delta value (4 or 8 octets): The sender requests that the peer should increment all the Child SA Replay Counters for the sender's incoming (the peer's outgoing) traffic by this value. The size of this field depends on the ESN bit associated with the Child SAs: if the ESN bit is 1, the field's size is 8 octets, otherwise it is 4 octets. We note that this constrains the Child SAs of each IKE SA to either all have the ESN bit on or off.

7. Implementation Details

This protocol does not change any of the existing IKEv2 rules regarding Message ID values.

The standby member can initiate the synchronization of IKEv2 Message ID's under different circumstances.

*When it receives a problematic IKEv2/IPsec packet, i.e. a packet outside its expected receive window.

*When it has to send the first IKEv2/IPsec packet after a failover event.

*When it has just received control from the active member and wishes to update the values proactively, so that it need not start this exchange later, when sending or receiving the request.

The standby member can initiate the synchronization of IPsec SA Replay Counters:

*If there has been traffic using the IPsec SA in the recent past and the standby member suspects that its Replay Counter may be stale.

Since there can be a large number of sessions at the standby member, and sending synchronization exchanges for all of them may result in overload, the standby member can choose to initiate the exchange in a

"lazy" fashion: only when it has to send or receive the request. In general, the standby member is free to initiate this exchange at its discretion.

8. IKE SA and IPsec SA Message Sequencing

The straightforward definitions of message sequence numbers, retransmissions and replay protection in IPsec and IKEv2 are strained by the failover scenarios described in this document. This section describes some policy choices that need to be made by implementations in this setting.

8.1. Handling of Pending IKE Messages

After sending its "receive" counter, the cluster member MUST reject any incoming IKE messages that are outside its declared window. A similar rule applies to the peer. Local policies vary, and strict implementations will reject any incoming IKE message arriving before Message ID synchronization is complete.

8.2. Handling of Pending IPsec Messages

For IPsec, there is often a trade-off between security and reliability of the protected protocols. Here again there is some leeway for local policy. Some implementations might accept incoming traffic that is outside the replay window for some time after the failover event. Strict implementations will only accept traffic that's inside the "safe" window.

8.3. IKE SA Inconsistencies

IKEv2 is normally a reliable protocol. As long as an IKE SA is valid, both peers share a single, consistent view of the IKE SA and all associated Child SAs. Failover situations as described in this document may involve forced deletion of IKE messages, resulting in inconsistencies, such as Child SAs that exist on only one of the peers. Such SAs would cause an INVALID_SPI to be returned when used by that peer.

The Working Group discussed at some point a proposed set of rules for dealing with such situations. However we believe that these situations should be rare in practice; as a result the "default" behavior of tearing down the entire IKE SA is to be preferred over the complexity of dealing with a multitude of edge cases.

9. Step by Step Details

This section goes through the sequence of steps of a typical failover event, looking at a case where the IKEv2 Message ID values are synchronized.

- *The active cluster member and the peer device establish the session. They both announce the capability to synchronize counter information by sending the `IKEV2_MESSAGE_ID_SYNC_SUPPORTED` notification in the `IKE_AUTH` Exchange.

- *Some time later, the active member dies, and a standby member takes over. The standby member sends its own idea of the IKE Message IDs (both incoming and outgoing) to the peer in an Informational message exchange with Message ID zero.

- *The peer first authenticates the message. The peer compares the received values with the values available locally and picks the higher value. It then updates its Message IDs with the higher values and also propose the same values in its response.

- *The peer should not wait for any pending responses while responding with the new Message ID values. For example, if the window size is 5 and the peer's window is 3-7, and if the peer has sent requests 3, 4, 5, 6, 7 and received responses only for 4, 5, 6, 7 but not for 3, then it should include the value 8 in its `EXPECTED_SEND_REQ_MESSAGE_ID` payload and should not wait for a response to message 3 anymore.

- *Similarly, the peer should also not wait for pending (incoming) requests. For example if the window size is 5 and the peer's window is 3-7 and if the peer has received requests 4, 5, 6, 7 but not 3, then it should send the value 8 in the `EXPECTED_RECV_REQ_MESSAGE_ID` payload, and should not expect to receive message 3 anymore.

10. Interaction with other drafts

The usage scenario of the IKEv2/IPsec SA counter synchronization proposal is that an IKEv2 SA has been established between the active member of a hot-standby cluster and a peer, then a failover event occurred with the standby member becoming active. The proposal further assumes that the IKEv2 SA state was continuously synchronized between the active and standby members of the cluster before the failover event.

- *Session resumption [\[RFC5723\]](#) assumes that a peer (client or initiator) detects the need to re-establish the session. In IKEv2/IPsec SA counter synchronization, it is the newly-active member (a gateway or responder) that detects the need to

synchronize the SA counter after the failover event. Also in a hot-standby cluster, the peer establishes the IKEv2/IPsec session with a single IP address that represents the whole cluster, so the peer normally does not detect the event of failover in the cluster unless the standby member takes too long to become active and the IKEv2 SA times out by use of the IKEv2 liveness check mechanism. To conclude, session resumption and SA counter synchronization after failover are mutually exclusive.

*The IKEv2 Redirect mechanism for load-balancing [\[RFC5685\]](#) can be used either during the initial stages of SA setup (the IKE_SA_INIT and IKE_AUTH exchanges) or after session establishment. SA counter synchronization is only useful after the IKE SA has been established and a failover event has occurred. So, unlike Redirect, it is irrelevant during the first two exchanges. Redirect after the session has been established is mostly useful for timed or planned shutdown/maintenance. A real failover event cannot be detected by the active member ahead of time, and so using Redirect after session establishment is not possible in the case of failover. So, Redirect and SA counter synchronization after failover are mutually exclusive.

*IKEv2 Failure Detection [\[I-D.ietf-ipsecme-failure-detection\]](#) solves a similar problem where the peer can rapidly detect that a cluster member has crashed based on a token. It is unrelated to the current scenario because the goal in failover is for the peer not to notice that a failure has occurred.

11. Security Considerations

Since Message ID synchronization messages need to be sent with Message ID zero, they are potentially vulnerable to replay attacks. Because of the semantics of this protocol, these can only be denial-of-service (DoS) attacks, and we are aware of two variants.

*Replay of Message ID synchronization request: This is countered by the requirement that the Send counter sent by the cluster member should always be monotonically increasing, a rule that the peer enforces by silently dropping messages that contradict it.

*Replay of the Message ID synchronization response: This is countered by sending the nonce data along with the synchronization payload. The same nonce data has to be returned in the response. Thus the standby member will accept a reply only for the current request. After it receives a valid response, it MUST NOT process the same response again and MUST discard any additional responses.

[12. IANA Considerations](#)

This document introduces four new IKEv2 Notification Message types as described in Section 6. The new Notify Message Types must be assigned values between 16396 and 40959.

Name	Value
IKEV2_MESSAGE_ID_SYNC_SUPPORTED	TBD by IANA
IPSEC_REPLAY_COUNTER_SYNC_SUPPORTED	TBD by IANA
IKEV2_MESSAGE_ID_SYNC	TBD by IANA
IPSEC_REPLAY_COUNTER_SYNC	TBD by IANA

[13. Acknowledgements](#)

We would like to thank Pratima Sethi and Frederic Detienne for their review comments and valuable suggestions for the initial version of the document.

We would also like to thank the following people (in alphabetical order) for their review comments and valuable suggestions: Dan Harkins, Paul Hoffman, Steve Kent, Tero Kivinen, David McGrew, and Pekka Riikonen.

[14. Change Log](#)

This section lists all the changes in this document.

NOTE TO RFC EDITOR: Please remove this section before publication.

[14.1. Draft -04](#)

Extended Sec. 3 for better coverage of other IPsec cluster-related issues, and how they are resolved within the existing standards.

[14.2. Draft -03](#)

Clarified the rules for Message ID sync, so that replay attacks can be avoided without a failover counter.

Added wording regarding inconsistent IKE state (basically choosing to ignore the problem) and further rules dealing with pending traffic. The IPsec replay counter delta value now refers to incoming traffic. The associated notification is only sent from the cluster to the peer, and not back.

[14.3. Draft -02](#)

Addressed comments by Yaron Sheffer posted on the WG mailing list. Numerous editorial changes.

14.4. Draft -01

Added "Multiple and Simultaneous failover" scenarios as pointed out by Pekka Riikonen.

Now document provides a mechanism to sync either IKEv2 message or IPsec replay counter or both to cater different types of implementations. HA cluster's "failover count" is used to encounter replay of sync requests by attacker.

The sync of IPsec SA replay counter optimized to to have just one global bumped-up outgoing IPsec SA counter of ALL Child SAs under an IKEv2 SA.

The examples added for IKEv2 Message ID sync to provide more clarity. Some edits as per comments on mailing list to enhance clarity.

14.5. Draft -00

Version 00 is identical to draft-kagarigi-ipsecme-ikev2-windowsync-04, started as WG document.

Added IPSECME WG HA design team members as authors.

Added comment in Introduction to discuss the window sync process on WG mailing list to solve some concerns.

15. References

15.1. Normative References

[1]	Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels" , BCP 14, RFC 2119, March 1997.
[2]	Kent, S. and K. Seo, " Security Architecture for the Internet Protocol ", RFC 4301, December 2005.
[3]	Kaufman, C., Hoffman, P., Nir, Y. and P. Eronen, " Internet Key Exchange Protocol Version 2 (IKEv2) ", RFC 5996, September 2010.

15.2. Informative References

[1]	Housley, R., " Using Advanced Encryption Standard (AES) Counter Mode With IPsec Encapsulating Security Payload (ESP) ", RFC 3686, January 2004.
[2]	Viega, J. and D. McGrew, " The Use of Galois/Counter Mode (GCM) in IPsec Encapsulating Security Payload (ESP) ", RFC 4106, June 2005.
[3]	Devarapalli, V. and K. Weniger, " Redirect Mechanism for the Internet Key Exchange Protocol Version 2 (IKEv2) ", RFC 5685, November 2009.
[4]	Sheffer, Y. and H. Tschofenig, " Internet Key Exchange Protocol Version 2 (IKEv2) Session Resumption ", RFC 5723, January 2010.
[5]	

	Nadas, S., " Virtual Router Redundancy Protocol (VRRP) Version 3 for IPv4 and IPv6 ", RFC 5798, March 2010.
[6]	Nir, Y., " IPsec Cluster Problem Statement ", RFC 6027, October 2010.
[7]	McGrew, D. and B. Weis, " Using Counter Modes with Encapsulating Security Payload (ESP) and Authentication Header (AH) to Protect Group Traffic ", RFC 6054, November 2010.
[8]	Nir, Y, Wierbowski, D, Detienne, F and P Sethi, " A Quick Crash Detection Method for IKE ", Internet-Draft draft-ietf-ipsecme-failure-detection-08, April 2011.

[Appendix A. IKEv2 Message ID Sync Examples](#)

This (non-normative) section presents some examples that illustrate how the IKEv2 Message ID values are synchronized. We use a tuple notation, denoting the two counters EXPECTED_SEND_REQ_MESSAGE_ID and EXPECTED_RECV_REQ_MESSAGE_ID on a member as (EXPECTED_SEND_REQ_MESSAGE_ID, EXPECTED_RECV_REQ_MESSAGE_ID).

[Appendix A.1. Normal Failover - Example 1](#)

```

Standby (Newly Active) Member                                     Peer
- - - - -
Sync Request (2, 3) ----->

                                Peer has the values (4, 5) so it sends
                                <----- (4, 5) as the Sync Response

```

[Appendix A.2. Normal Failover - Example 2](#)

```

Standby (Newly Active) Member                                     Peer
- - - - -
Sync Request (2, 5) ----->

                                Peer has the values (2, 4) so it sends
                                <----- (5, 4) as the Sync Response

```

Appendix A.3. Simultaneous Failover

In the case of simultaneous failover, both sides send the synchronization request, but whichever side has the higher value will be eventually synchronized.

Standby (Newly Active) Member	Peer

Sync Request (4,4) ----->

<----- Sync Request (5,5)

Sync Response (5,5) ----->

<----- Sync Response (5,5)

Authors' Addresses

Raj Singh (Editor) Singh, Ed. Cisco Systems, Inc. Divyashree Chambers, B Wing, O'Shaugnessy Road Bangalore, Karnataka 560025 India Phone: +91 80 4301 3320 EMail: rsj@cisco.com

Kalyani Garigipati Kalyani Cisco Systems, Inc. Divyashree Chambers, B Wing, O'Shaugnessy Road Bangalore, Karnataka 560025 India Phone: +91 80 4426 4831 EMail: kagarigi@cisco.com

Yoav Nir Nir Check Point Software Technologies Ltd. 5 Hasolelim St. Tel Aviv, 67897 Israel EMail: ynir@checkpoint.com

Yaron Sheffer Sheffer Independent EMail: aronf.ietf@gmail.com

Dacheng Zhang Zhang Huawei Technologies Ltd. EMail: zhangdacheng@huawei.com