Network Working Group
Internet-Draft
Intended status: Informational
Expires: December 12, 2010

Y. Nir
Check Point
June 10, 2010

IPsec Cluster Problem Statement
draft-ietf-ipsecme-ipsec-ha-05

Abstract

This document defines terminology, problem statement and requirements for implementing IKE and IPsec on clusters. It also describes gaps in existing standards and their implementation that need to be filled, in order to allow peers to interoperate with clusters from different vendors. An agreed terminology, problem statement and requirements will allow the IPSECME WG to consider development of IPsec/IKEv2 mechanisms to simplify cluster implementations.

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 December 12, 2010.

Copyright Notice

Copyright (c) 2010 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

IKEv2, as described in [RFC4306] and [IKEv2bis], and IPsec, as described in [RFC4301] and others, allows deployment of VPNs between different sites as well as from VPN clients to protected networks.

As VPNs become increasingly important to the organizations deploying them, there is a demand to make IPsec solutions more scalable and less prone to down time, by using more than one physical gateway to either share the load or back each other up, forming a "cluster" (see Section 2). Similar demands have been made in the past for other critical pieces of an organization´s infrastructure, such as DHCP and DNS servers, web servers, databases and others.

IKE and IPsec are in particular less friendly to clustering than these other protocols, because they store more state, and that state is more volatile. Section 2 defines terminology for use in this document, and in the envisioned solution documents.

In general, deploying IKE and IPsec in a cluster requires such a large amount of information to be synchronized among the members of the cluster, that it becomes impractical. Alternatively, if less information is synchronized, failover would mean a prolonged and intensive recovery phase, which negates the scalability and availability promises of using clusters. In Section 3 we will describe this in more detail.

1.1. Conventions Used in This Document

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].

2. Terminology

"Single Gateway" is an implementation of IKE and IPsec enforcing a certain policy, as described in [RFC4301].

"Cluster" is a set of two or more gateways, implementing the same security policy, and protecting the same domain. Clusters exist to provide both high availability through redundancy, and scalability through load sharing.

"Member" is one gateway in a cluster.

"Availability" is a measure of a system´s ability to perform the service for which it was designed. It is measured as the percentage of time a service is available, from the time it is supposed to be available. Colloquially, availability is sometimes expressed in "nines" rather than percentage, with 3 "nines" meaning 99.9% availability, 4 "nines" meaning 99.99% availability, etc.

"High Availability" is a condition of a system, not a configuration type. A system is said to have high availability if its expected down time is low. High availability can be achieved in various ways, one of which is clustering. All the clusters described in this document achieve high availability. What "high" means depends on application, but usually is 4 to 6 "nines" (at most 0.5-50 minutes of down time per year in a system that is supposed to be available all the time.

"Fault Tolerance" is a condition related to high availability, where a system maintains service availability, even when a specified set of fault conditions occur. In clusters, we expect the system to maintain service availability, when one or more of the cluster members fails.

"Completely Transparent Cluster" is a cluster where the occurence of a fault is never visible to the peers.

"Partially Transparent Cluster" is a cluster where the occurence of a fault may be visible to the peers.

"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 the "active", whereas the other(s) are referred to as "stand-bys". [VRRP] is one method of building such a cluster.

"Load Sharing Cluster", or "LS Cluster" is a cluster where more than one of the members may be active at the same time. The term "load balancing" is also common, but it implies that the load is actually balanced between the members, and this is not a requirement.

"Failover" is the event where a one member takes over some load from some other member. In a hot standby cluster, this hapens when a standby member becomes active due to a failure of the former active member, or because of an administrator command. In a load sharing cluster, this usually happens because of a failure of one of the members, but certain load-balancing technologies may allow a particular load (such as all the flows associated with a particular child SA) to move from one member to another to even out the load, even without any failures.

"Tight Cluster" is a cluster where all the members share an IP address. This could be accomplished using configured interfaces with specialized protocols or hardware, such as VRRP, or through the use of multicast addresses, but in any case, peers need only be configured with one IP address in the PAD.

"Loose Cluster" is a cluster where each member has a different IP address. Peers find the correct member using some method such as DNS queries or [REDIRECT]. In some cases, a member´s IP address(es) may be allocated to another member at failover.

"Synch Channel" is a communications channel among the cluster members, used to transfer state information. The synch channel may or may not be IP based, may or may not be encrypted, and may work over short or long distances. The security and physical characteristics of this channel are out of scope for this document, but it is a requirement that its use be minimized for scalability.

3. The Problem Statement

This section starts by scoping the problem, and goes on to list each of the issues encountered while setting up a cluster of IPsec VPN gateways.

3.1. Scope

This document will make no attempt to describe the problems in setting up a generic cluster. It describes only problems related to the IKE/IPsec protocols.

The problem of synchronizing the policy between cluster members is out of scope, as this is an administrative issue that is not particular to either clusters or to IPsec.

The interesting scenario here is VPN, whether tunneled site-to-site or remote access. Host-to-host transport mode is not expected to benefit from this work.

We do not describe in full the problems of the communication channel between cluster members (the Synch Channel), nor do we intend to specify anything in this space later. Specifically, mixed-vendor clusters are out of scope.

The problem statement anticipates possible protocol-level solutions between IKE/IPsec peers, in order to improve the availability and/or performance of VPN clusters. One vendor´s IPsec endpoint should be able to work, optimally, with another vendor´s cluster.

3.2. Lots of Long Lived State

IKE and IPsec have a lot of long lived state:

- o IKE SAs last for minutes, hours, or days, and carry keys and other information. Some gateways may carry thousands to hundreds of thousands of IKE SAs.
- o IPsec SAs last for minutes or hours, and carry keys, selectors and other information. Some gateways may carry hundreds of thousands such IPsec SAs.
- o SPD (Security Policy Database) Cache entries. While the SPD is unchanging, the SPD cache changes on the fly due to narrowing. Entries last at least as long as the SAD (Security Association Database) entries, but tend to last even longer than that.

A naive implementation of a cluster would have no synchronized state, and a failover would produce an effect similar to that of a rebooted gateway. [resumption] describes how new IKE and IPsec SAs can be recreated in such a case.

3.3. IKE Counters

We can overcome the first problem described in Section 3.2, by synchronizing states - whenever an SA is created, we can synch this new state to all other members. However, those states are not only long-lived, they are also ever changing.

IKE has message counters. A peer may not process message n until after it has processed message n-1. Skipping message IDs is not allowed. So a newly-active member needs to know the last message IDs both received and transmitted.

In some cases, it is feasible to synchronize information about the IKE message counters after every IKE exchange. This way, the newly active member knows what messages it is allowed to process, and what message IDs to use on IKE requests, so that peers process them. We leave it for future discussion to determine if it is always feasible to do so.

3.4. Outbound SA Counters

ESP and AH have an optional anti-replay feature, where every protected packet carries a counter number. Repeating counter numbers is considered an attack, so the newly-active member MUST NOT use a replay counter number that has already been used. The peer will drop those packets as duplicates and/or warn of an attack.

Though it may be feasible to synchronize the IKE message counters, it is almost never feasible to synchronize the IPsec packet counters for every IPsec packet transmitted. So we have to assume that at least for IPsec, the replay counter will not be up-to-date on the newly- active member, and the newly-active member may repeat a counter.

A possible solution is to synch replay counter information, not for each packet emitted, but only at regular intervals, say, every 10,000 packets or every 0.5 seconds. After a failover, the newly-active member advances the counters for outbound IPsec SAs by 10,000. To the peer this looks like up to 10,000 packets were lost, but this should be acceptable, as neither ESP nor AH guarantee reliable delivery.

3.5. Inbound SA Counters

An even tougher issue, is the synchronization of packet counters for inbound IPsec SAs. If a packet arrives at a newly-active member, there is no way to determine whether this packet is a replay or not. The periodic synch does not solve the problem at all, because suppose we synchronize every 10,000 packets, and the last synch before the failover had the counter at 170,000. It is probable, though not certain, that packet number 180,000 has not yet been processed, but if packet 175,000 arrives at the newly- active member, it has no way of determining whether or not that packet has or has not already been processed. The synchronization does prevent the processing of really old packets, such as those with counter number 165,000. Ignoring all counters below 180,000 won´t work either, because that´s up to 10,000 dropped packets, which may be very noticeable.

The easiest solution is to learn the replay counter from the incoming traffic. This is allowed by the standards, because replay counter verification is an optional feature (see section 3.2 in [RFC4301]). The case can even be made that it is relatively secure, because non- attack traffic will reset the counters to what they should be, so an attacker faces the dual challenge of a very narrow window for attack, and the need to time the attack to a failover event. Unless the attacker can actually cause the failover, this would be very difficult. It should be noted, though, that although this solution is acceptable as far as RFC 4301 goes, it is a matter of policy whether this is acceptable.

Another possible solution to the inbound IPsec SA problem is to rekey all child SAs following a failover. This may or may not be feasible depending on the implementation and the configuration.

3.6. Missing Synch Messages

The synch channel is very likely not to be infallible. Before failover is detected, some synchronization messages may have been missed. For example, the active member may have created a new Child SA using message n. The new information (entry in the SAD and update to counters of the IKE SA) is sent on the synch channel. Still, with every possible technology, the update may be missed before the failover.

This is a bad situation, because the IKE SA is doomed. the newly- active member has two problems:

- o It does not have the new IPsec SA pair. It will drop all incoming packets protected with such an SA. This could be fixed by sending some DELETEs and INVALID_SPI notifications, if it wasn´t for the other problem...
- o The counters for the IKE SA show that only request n-1 has been sent. The next request will get the message ID n, but that will be rejected by the peer. After a sufficient number of retransmissions and rejections, the whole IKE SA with all associated IPsec SAs will get dropped.

The above scenario may be rare enough that it is acceptable that on a configuration with thousands of IKE SAs, a few will need to be recreated from scratch or using session resumption techniques. However, detecting this may take a long time (several minutes) and this negates the goal of creating a cluster in the first place.

3.7. Simultaneous use of IKE and IPsec SAs by Different Members

For LS clusters, all active members may need to use the same SAs, both IKE and IPsec. This is an even greater problem than in the case of HS clusters, because consecutive packets may need to be sent by different members to the same peer gateway.

The solution to the IKE SA issue is up to the application. It´s possible to create some locking mechanism over the synch channel, or else have one member "own" the IKE SA and manage the child SAs for all other members. For IPsec, solutions fall into two broad categories.

The first is the "sticky" category, where all communications with a single peer, or all communications involving a certain SPD cache entry go through a single peer. In this case, all packets that match any particular SA go through the same member, so no synchronization of the replay counter needs to be done. Inbound processing is a "sticky" issue, because the packets have to be processed by the correct member based on peer and SPI. Another issue is that most load balancers will not be able to match the SPIs of the encrypted side to the clear traffic, and so the wrong member may get the the other half of the flow.

The second is the "duplicate" category, where the child SA is duplicated for each pair of IPsec SAs for each active member. Different packets for the same peer go through different members, and get protected using different SAs with the same selectors and matching the same entries in the SPD cache. This has some shortcomings:

- o It requires multiple parallel SAs, which the peer has no use for. Section 2.8 or [RFC4306] specifically allows this, but some implementation might have a policy against long term maintenance of redundant SAs.
- o Different packets that belong to the same flow may be protected by different SAs, which may seem "weird" to the peer gateway, especially if it is integrated with some deep inspection middleware such as a firewall. It is not known whether this will cause problems with current gateways. It is also impossible to mandate against this, because the definition of "flow" varies from one implementation to another.
- o Reply packets may arrive with an IPsec SA that is not "matched" to the one used for the outgoing packets. Also, they might arrive at a different member. This problem is beyond the scope of this document and should be solved by the application, perhaps by forwarding misdirected packets to the correct gateway for deep inspection.

3.7.1. Outbound SAs using counter modes

For SAs involving counter mode ciphers such as [CTR] or [GCM] 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. An SA shared between more than one active member, or even failing over from one member to another need to make sure that they do not generate the same initial vector. See [COUNTER_MODES] for a discussion of this problem in another context.

3.8. Different IP addresses for IKE and IPsec

In many implementations there are separate IP addresses for the cluster, and for each member. While the packets protected by tunnel mode child SAs are encapsulated in IP headers with the cluster IP address, the IKE packets originate from a specific member, and carry that member´s IP address. For the peer, this looks weird, as the usual thing is for the IPsec packets to come from the same IP address as the IKE packets.

One obvious solution, is to use some fancy capability of the IKE host to change things so that IKE packets also come out of the cluster IP address. This can be achieved through NAT or through assigning multiple addresses to interfaces. This is not, however, possible for all implementations.

[ARORA] discusses this problem in greater depth, and proposes another solution, that does involve protocol changes.

3.9. Allocation of SPIs

The SPI associated with each child SA, and with each IKE SA, MUST be unique relative to the peer of the SA. Thus, in the context of a cluster, each cluster member MUST generate SPIs in a fashion that avoids collisions (with other cluster members) for these SPI values. The means by which cluster members achieve this requirement is a local matter, outside the scope of this document.

4. Security Considerations

Implementations running on clusters MUST be as secure as implementations running on single gateways. In other words, no extension or interpretation used to allow operation in a cluster may facilitate attacks that are not possible for single gateways.

Moreover, thought must be given to the synching requirements of any protocol extension, to make sure that it does not create an opportunity for denial of service attacks on the cluster.

As mentioned in Section 3.5, allowing an inbound child SA to fail over to another member has the effect of disabling replay counter protection for a short time. Though the threat is arguably low, it is a policy decision whether this is acceptable.

5. IANA Considerations

This document has no actions for IANA.

6. Acknowledgements

This document is the collective work, and includes contribution from many people who participate in the IPsecME working group.

The editor would particularly like to acknowledge the extensive contribution of the following people (in alphabetical order): Jitender Arora, Jean-Michel Combes, Dan Harkins, Steve Kent, Tero Kivinen, Yaron Sheffer, Melinda Shore, and Rodney Van Meter.

7. Change Log

NOTE TO RFC EDITOR: REMOVE THIS SECTION BEFORE PUBLICATION

Version 00 was identical to draft-nir-ipsecme-ipsecha-ps-00, re-spun as an WG document.

Version 01 included closing issues 177, 178 and 180, with updates to terminology, and added discussion of inbound SAs and the CTR issue.

Version 02 includes comments by Yaron Sheffer and the acknowledgement section.

Version 03 fixes some ID-nits, and adds the problem presented by Jitender Arora in [ARORA].

Version 04 fixes a spelling mistake, moves the scope discussion to a subsection of its own (Section 3.1), and adds a short discussion of the duplicate SPI problem, presented by Jean-Michel Combes.

8. Informative References

[ARORA] Arora, J. and P. Kumar, "Alternate Tunnel Addresses for IKEv2", draft-arora-ipsecme-ikev2-alt-tunnel-addresses (work in progress), April 2010.

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

[CTR] Housley, R., "Using Advanced Encryption Standard (AES) Counter Mode", RFC 3686, January 2009.

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

[IKEv2bis] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol: IKEv2", draft-ietf-ipsecme-ikev2bis (work in progress), May 2010.

[REDIRECT] Devarapalli, V. and K. Weniger, "Redirect Mechanism for

IKEv2", RFC 5685, November 2009.

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

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

[RFC4306] Kaufman, C., "Internet Key Exchange (IKEv2) Protocol", RFC 4306, December 2005.

[VRRP] Nadas, S., "Virtual Router Redundancy Protocol (VRRP)", RFC 5798, March 2010.

[resumption] Sheffer, Y. and H. Tschofenig, "IKEv2 Session Resumption", RFC 5723, January 2010.

Author´s Address

Yoav Nir
Check Point Software Technologies Ltd.
5 Hasolelim st.
Tel Aviv 67897
Israel

Email: ynir@checkpoint.com