IPsec High Availability

draft-ietf-ipsecme-ipsec-ha-00

Yoav Nir

March 2010

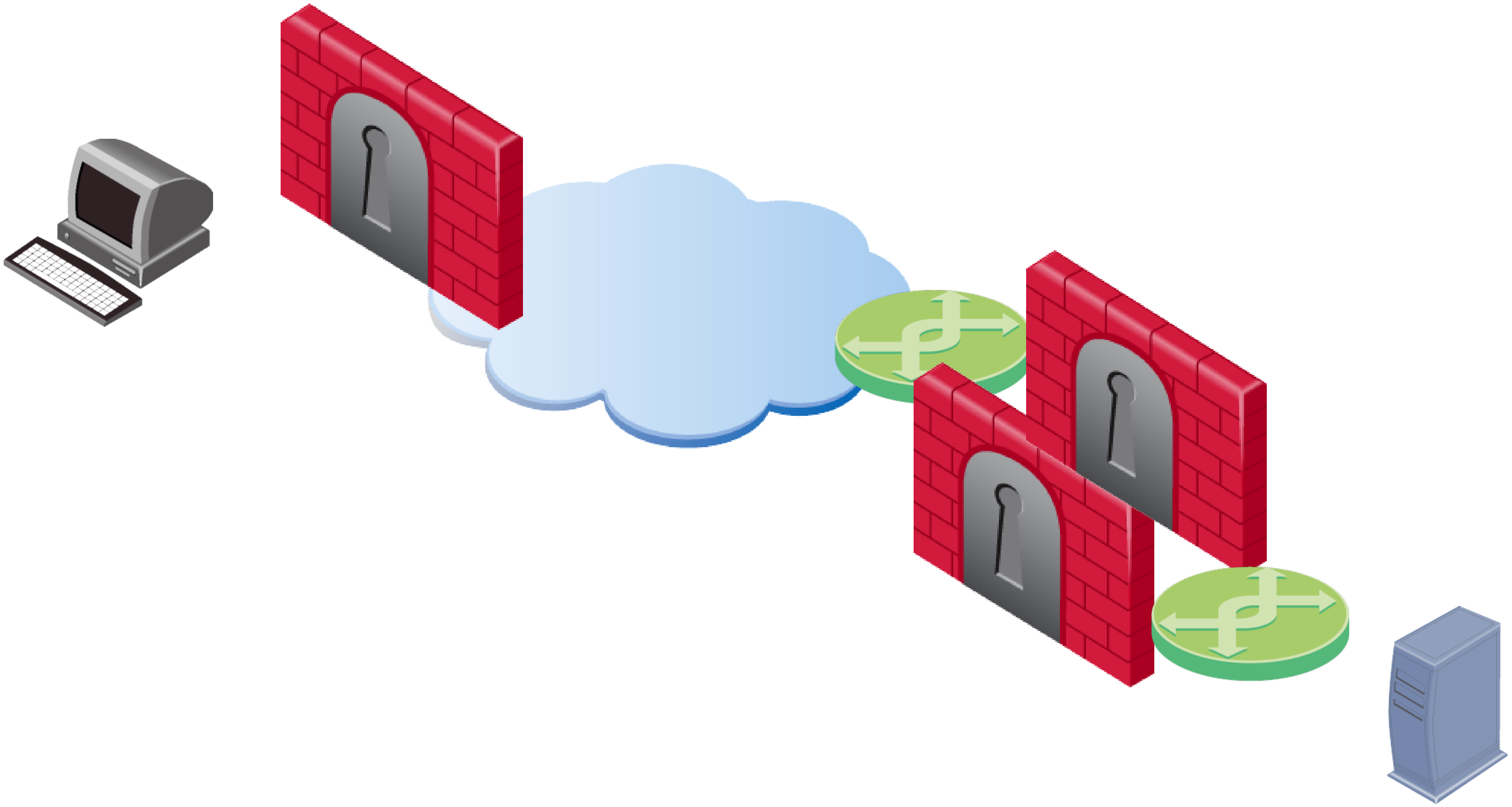
Agenda

- The Work Item
- draft-00
- Terminology
- Problems so far
- Call for contribution

The Work Item

This work item will define **a problem statement and requirements** for potential IPsec/IKEv2 mechanism (or multiple mechanisms) to simplify cluster implementations. The result will be an informational document that, once completed, may lead to chartering one or more new work items to specify the actual mechanisms. The scope is restricted to mechanism(s) that are visible to the peer, and thus usually require interoperability between vendors. **Mixed-vendor clusters, and protocols between the cluster members, are explicitly out of scope of this work item.**

The Work Item



The Work Item - Goals

- Map out as many of the challenges as possible, for multiple vendors
- Lead to solutions that:
  - Require as few as possible extensions
  - Map “weirdnesses” in the external appearance of cluster behavior, so that peers not consider it “hinky”
  - Minimize the load on the synch channel
- That's an ordered list!

draft-00

- Published February 25th. Similar to draft-nir-ipsecme-ipsecha-ps-00.
- 9 pages (so far)
- Defines terminology for discussing load sharing and high availability clusters
  - Terminology varies greatly among vendors
- Describes 4 issues
  - More to come
- Perhaps we need a “model” section with ASCII art.

Terminology

- Cluster – A set of two or more gateways implementing the same policy, and protecting the same domain. They may or may not have a single IP address.
- Member – One gateway in a cluster
- High Availability cluster – A cluster where only one member is active at any given time.
- Load Sharing cluster – A cluster where more than one member is active at any given time.
- “Hinky” - that's behavior of the cluster, consistent with the RFC, but not expected by the peer.

Terminology

- Failover – that's the event, where one member takes over the tasks (SAs) of another member
  - Because of failure of the other member?
- Synch channel – that's a medium for communications between members
  - Used to synchronize state
  - Bandwidth varies.
  - We need to minimize its use.

Problems currently in the draft

- Lots of long lived state
  - IKE SAs
  - Child SAs
  - SPD Cache entries
- All that state needs to be syched.
- If not, a failover looks like a reboot.
- Some of the state is constantly changing.

Problems currently in the draft

- IKE and IPsec counters
  - Message identifiers in IKE
  - Replay counters in IPsec
- You MUST NOT miss any IKE messages
  - If a message is lost in failover, the IKE SA is doomed
- With IPsec, packets can be skipped
  - Skip ahead on failover?
  - May look “hinky”

Problems currently in the draft

- At failover, some of the state may not have been synched
  - IKE SAs become useless
    - Maybe we need a “reset counter” extension
  - IPsec SAs may cause replay attack alerts
  - INVALID_SPI may lead some peers to tear down the IKE SA, as IKE SAs should not have mismatched state.

Problems currently in the draft

- Simultaneous use of the IKE SA by more than one member
  - Might re-use a message ID
  - Possible solution: locking mechanism over the synch channel
  - Possible solution: one member does all the work for a particular IKE SA.
    - What if the members don't share an IP address?
- Both solutions do not require modifications to peers (a good thing) but heavily use the synch channel (a bad thing)

Problems currently in the draft

- Simultaneous use of the Child SA by more than one member
  - No way to get the replay counters right
  - Possible solution: two parallel SAs
  - Explicitly allowed by RFC 4306:

```
Note that IKEv2 deliberately allows parallel SAs with
the same traffic selectors between common endpoints.
One of the purposes of this is to support traffic
quality of service (QoS) differences among the SAs
(see [RFC2474], [RFC2475], and section 4.1 of
[RFC2983]).
```

Problems currently in the draft

- So what's wrong with this solution?
  - Does not allow to associate an SA pair with a flow.
  - Long term multiple SAs, which some peers may not like.
  - Different parts of a flow may go to different members. That's up to the cluster makers to solve.

Problems currently in the draft

- Another solution is to make peers or SPD cache entries “sticky” for the cluster member.
  - Doesn't work with commodity load balancers. They don't know about SPIs, or about the selectors in the SPD cache.
  - One SA may be responsible for 95% of the load, so we've just lost the “sharing”.

Call for Contributions

- The problems currently in the draft are things I've seen in Check Point, and things I've heard about from one other vendor.
- Different vendors use different technology, and have different constraints, in particular:
  - Load balancer technology and capability
  - Synch channel speed, bandwidth and reliability.
- We need to hear from other vendors!

?

http://tools.ietf.org/html/draft-ietf-ipsecme-ipsec-ha