draft-ietf-pim-sm-bsr-04.txt

PIM WG, IETF-60, San Diego, August 3 2004

Alexander Gall gall@switch.ch

Nidhi Bhaskar nbhaskar@cisco.com
Stig Venaas venaas@uninett.no

Changes from `-03` to `-04`

1. Minor issues
2. Using holdtime at all PIM routers to manage RP-set
3. Scoping

Minor issues

- Removed PIM-SM-specific text, spec now applies to “RP-based PIM protocols”
- Clarified that Bi-dir flag must not be ignored if understood but Bi-directional PIM disabled
- Removed final selection of RP from set of candidates, must be part of respective PIM spec.

Holdtime

Up to `-03`: use holdtime at BSR only

- State at routers must be removed explicitely through BSMs
- Routers may lose part of RP-set during election of a new BSR
  - First BSM of new BSR is empty (can be ignored)
  - Next BSM might be incomplete but replaces old RP-set
- Withdrawal of group-range is tricky, BSR must synthesize BSMs with RP count zero

BSR times out all mappings associated with an C-RP simultaneously, not individual group mappings.

Holdtime (cont'd)

Proposal: holdtime already included in BSM, use it at all PIM routers

Define group-to-RP mapping (GRPM) as basic building block

- Group range (address/mask)
- RP address
- RP priority
- Holdtime
- Hash mask length

GRPMs are timed out independently

Holdtime (cont'd)

Processing at BSR router

- Create GRPMs from received C-RP-Advs
- Add new GRPMs to C-RP-set, reset timer of existing
- Remove GRPM if timer expires or holdtime in C-RP-Adv = 0
- Select subset from C-RP-set as RP-set
- Construct BSM from RP-set

Processing at non-BSR router

- Create GRPMs from received BSMs
- Add new GRPMs to local RP-set, reset timer of existing
- Remove GRPM if timer expires or holdtime in BSM = 0

Basically a mechanism for caching individual GRPMs, BSR acts as "relay".

Holdtime (cont'd)

Possible loss of mappings during BSR failover

- BS Timer expires (missed last two BSMs)
- BSR election and collection of complete C-RP-set takes up to 213s with default timer values. Can be reduced to ≈150s (let C-RP send an Adv immediately)
- Default Holdtime is 150s

Proposal to give more slack

- All routers store a copy of the last BSM
- Use copy to refresh RP-set when BS Timer expires
- Gives new BSR ≈150s to receive complete C-RP-set
- Should be enough to not lose mappings in most common case
- Lightweight, doesn't need additional timers

Scoping

Scoping support in current draft

- Based on RFC2365, scope defined by address/mask
- Separate BSR election per scope zone
- Separate BSMF (BSM fragment) per scope
  - marked with “Z” bit
  - filtered at ZBRs (zone boundary router)
- Notion of “global scope” (better name would be “non-scoped”)
  - BSMF with Z-bit cleared
  - not filtered at ZBRs
  - Must be supported for compatibility with non-scoped BSR (RFC2362)
- No text about differences between IPv4/v6

Fundamental differences between IPv4/IPv6 scoping architectures

- IPv4
  - “Prefix-based” (scopes modelled after IPv6)
    - Link-local: 224.0.0.0/24
    - Site-local: 239.255.0.0/16
    - Org-local: 239.192.0.0/14
    - Global: 224.0.1.0-238.255.255.255 (224/4 w/o above)
  - Nested, variable scopes in org-local range w/ longer prefixes plus artificial ordering of non-overlapping prefixes link < site < org
- IPv6
  - “Manifest scoping” with 4-bit scope-ID → fixed # of scopes
  - Scopes nested by scope-ID, address ranges do not overlap

Very little practical experience with scoping.

Representation of IPv6 scopes

- “Encoded-Group address” uses address/mask
- IPv6 has 4-bit scope ID *preceeded* by flag bits

```
|   8    |  4 |  4 |                   112 bits                   |
+------ -+----+----+----------------------------------------------+
|11111111|flgs|scop|                   group ID                   |
+--------+----+----+----------------------------------------------+
```

- An entire IPv6 scope range can’t be represented as a single prefix

Representation of IPv6 scopes

Options

1. Allow a scope to be composed of a set of disjoint ranges, include all in a single BSMF or use separate BSFM for each

   + Same code can handle v4/v6

   - Unwieldy (up to 16 ranges per scope)

   - Overlapping sets of ranges are problematic. Could declare them illegal, but need to specify how to handle conflicts.

2. Only use the scope-ID to identify a scope and match at zone boundaries

   + Simple, robust

   - Can't share code with IPv4

   - Locks to a particular scoping architecture

Non-scoped/global scope BSM

- Draft distinguishes global scope (Z-bit = 0) and admin scope (Z-bit = 1) BSM
- Terminology not adequate for IPv6 (global = scope-ID E)
- Non-scoped BSM seems useless for IPv6 (all addresses have intrinsic scope), pure IPv4 legacy
- Scoped ranges embedded in non-scoped BSM (e.g. through misconfiguration) leak into all scope zones of the domain

Basic questions

1. Is the scoping architecture well defined by RFCs 2365 (Admin scoping) and 3513 (IPv6 addr. arch.)?
2. Do we need to be able to adapt to different scoping architectures in the fututre?

If 1 = no, must clarify scoping architecture first. Figure out what options we have (suspend BSR spec and get scoping straight first, remove scoping from current draft and press on, . . .)

If 1 = yes and 2 = no, go ahead.

If 1 = yes and 2 = yes, figure out what kind of flexibility is required (how do you parametrize a class of scoping architectures? Can address/mask + "set of ranges" accomodate all future architectures? Can BSR be made extensible in another manner?)