Multicast Source Filter API

draft-ietf-magma-msf-api-*.txt

Dave Thaler

dthaler@microsoft.com

Recap of API

- Spec defines socket API to enable (but not depend upon) SSM, IGMPv3, MLDv2
- Basic (delta-based) API:
  - Adds IPv4-specific socket options to make it easy for existing applications to use (primary issue is interface identification)
    - ASM: IP_{BLOCK,UNBLOCK}_SOURCE
    - SSM: IP_{ADD,DROP}_SOURCE_MEMBERSHIP
  - Defines protocol-independent options which are recommended for new apps
    - ASM: MCAST_{JOIN,LEAVE}_GROUP, MCAST_{BLOCK,UNBLOCK}_SOURCE
    - SSM: MCAST_{JOIN,LEAVE}_SOURCE_GROUP
  - No open issues on this part of the draft

Advanced (full-state) API

- Get and set filter mode and source list
- Get requires both input and output buffers
- Drafts 00-03 defined ioctl's for advanced (full-state) API
  - On many platforms getsockopt cannot take an input buffer
- Open Issue: Some recent suggestions to use getsockopt or even new functions instead of ioctl

Austin Group (POSIX) discussion

- General POSIX API principles:
  - "lack of type-safety in interfaces is a bad thing which should be avoided when possible“
  - "avoid namespace pollution
  - "standardize existing practice"
- General recommendation was that neither ioctl nor getsockopt was appropriate
  - Type-safety
  - Avoid getsockopt issues
  - A new function could easily wrap an ioctl or getsockopt if needed on some platform

Tentative Resolution for draft -04

- Move old ioctl api to Appendix as historical information
- Define new type-safe functions (described later)
- Add comment on implementation freedom:

  “A new function can be written as a wrapper over an ioctl, getsockopt, or setsockopt call, if necessary. Hence, it provides more freedom as to how the functionality is implemented in an operating system.

  For example, a new function might be implemented as an inline function in an include file, or a function exported from a user-mode library which internally uses some mechanism to exchange information with the kernel, or be implemented directly in the kernel.”

Open Issue

- From requirements list in section 2:

  “Applications should be able to detect when the new source filter APIs are unavailable (e.g., calls fail with the ENOTSUPP error) and react gracefully (e.g., revert to old non-source-filter API or display a meaningful error message to the user).”
- Is this still a requirement?
- Can this be met on all platforms?
- Should we limit the degree of implementation freedom?

Proposed IPv4-specific functions

int setipv4sourcefilter(s, struct in_addr interface, struct in_addr group, uint32_t fmode, uint32_t numsrc, struct in_addr *slist)

int getipv4sourcefilter(s, struct in_addr interface, struct in_addr group, uint32_t *fmode, uint32_t *numsrc, struct in_addr *slist)

- Semantics unchanged from previous ioctls

Proposed generic functions

int setsourcefilter(s, uint32_t interface, struct sockaddr *group, uint32_t fmode, uint_t numsrc, struct sockaddr_storage *slist)

int getsourcefilter(s, uint32_t interface, struct sockaddr *group, uint32_t fmode, uint_t *numsrc, struct sockaddr_storage *slist)

- sockaddr_storage vs sockaddr rationale:
  - slist is an array
  - Ioctl api used it for group too but now that’s a pointer

Questions

- Implementors: is this API acceptable?
- Any other issues?
- When target WG last call?

An operational problem with IGMP snooping switches

(mailing list thread)

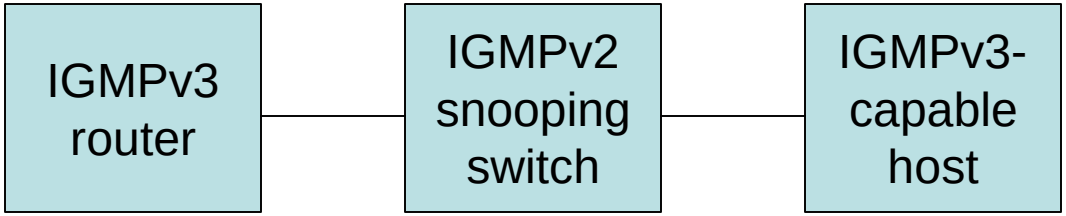
Dave Thaler

dthaler@microsoft.com

IGMPv3 Deployment Status

- The good news:
  - IGMPv3 hosts are being used by real customers
  - IGMPv3 routers are being used by real customers
- The bad news:
  - Since start of ’03, three unrelated organizations using different switches have reported a similar problem…

Summary of problem



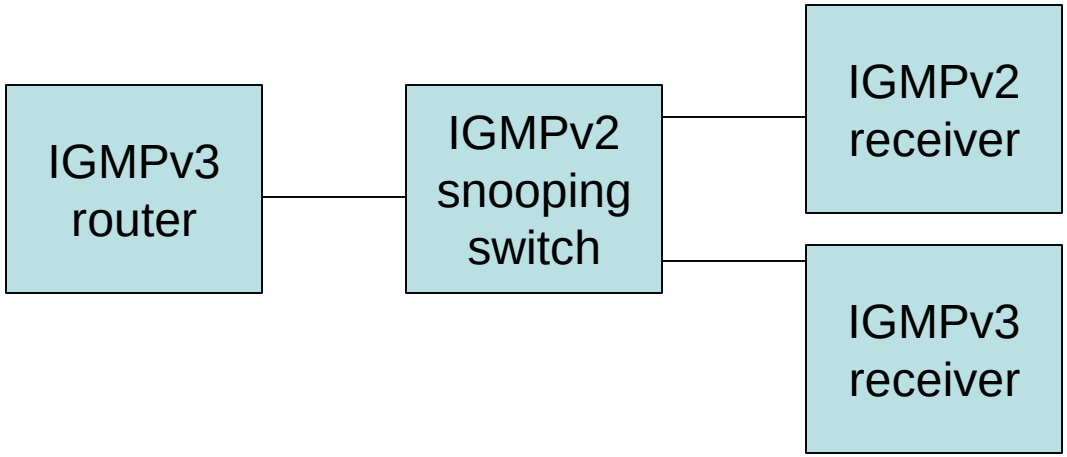
- Router sends IGMPv3 query, which tells hosts to use IGMPv3
- Hosts send IGMPv3 reports
- Switch mishandles unrecognized messages somehow
  - Doesn’t forward reports
  - Forwards report but doesn’t forward data
  - Forwards report and data but not queries, data then times out (no periodic refreshes)
- Net result: Multicast breaks when you upgrade a router or host (whichever comes last) from IGMPv2 to IGMPv3

Operational Workarounds

- Best solution is to ensure router is configured correctly
  - Only configure IGMPv3 on an interface without IGMPv2-only snooping switches
  - Harder if routers and switches owned by separate organizations
  - Impossible(?) if router requires IGMPv3 to support one customer and thereby breaks another customer that has an IGMPv2 switch
- Customers have asked that hosts be configurable to force IGMPv2 even though RFC behavior would use IGMPv3
  - Note that SSM won't work with these broken switches anyway
  - This requires manual configuration on every host

Snooping draft impact

- Currently only solves part of the problem
  - Flood unrecognized IGMP (covers IGMPv3 queries and reports)
  - If no reports for a group, flood data to router ports (or all ports if configured to do so)
  - Doesn’t cover case where a report was received



What should IETF do?

- At least document the problem
  - MAGMA or MBoneD?
- Any other solutions?