Evolution of Multicast Engine on Linux

USAGI Project
Keio University

YOSHIFUJI, Hideaki
yoshfuji + lca2009 _AT_ linux-ipv6.org

http://www.linux-ipv6.org/materials/200901-LCA2009/

About Me

- Hideaki YOSHIFUJI
  - Core member of **USAGI Project**
  - Co-maintainer of **Networking [IPv4/IPv6]** area
  - Assistant professor, Keio University
    - Networking, especially IPv6, “ubiquitous” computing
    - Linux, open source

    etc...

What is Multicast?

- **Multicast**
  - a network addressing method for the delivery of information to a group of destinations simultaneously using the most efficient strategy to deliver the messages over each link of the network only once, creating copies only when the links to the multiple destinations split. (wikipedia)
- **IP Multicast**
  - Multicast by IP

IP Multicast Elements

- Device-level group management
- Group management API / protocols
- Packet forwarding
- Tree / routing management
- Tunneling

- Device-level group management
- Group management API / protocols
- Packet forwarding
- Tree / routing management
- Tunneling

Device-level group management

- General (net_device{})
  - dev->mc_list
- Per-Driver
  - hash / list etc.
  - IFF_PROMISC / IFF_ALLMULTI

Note:

IP layer MAY receive multicast packets not on the general list.

Device-level group management issue

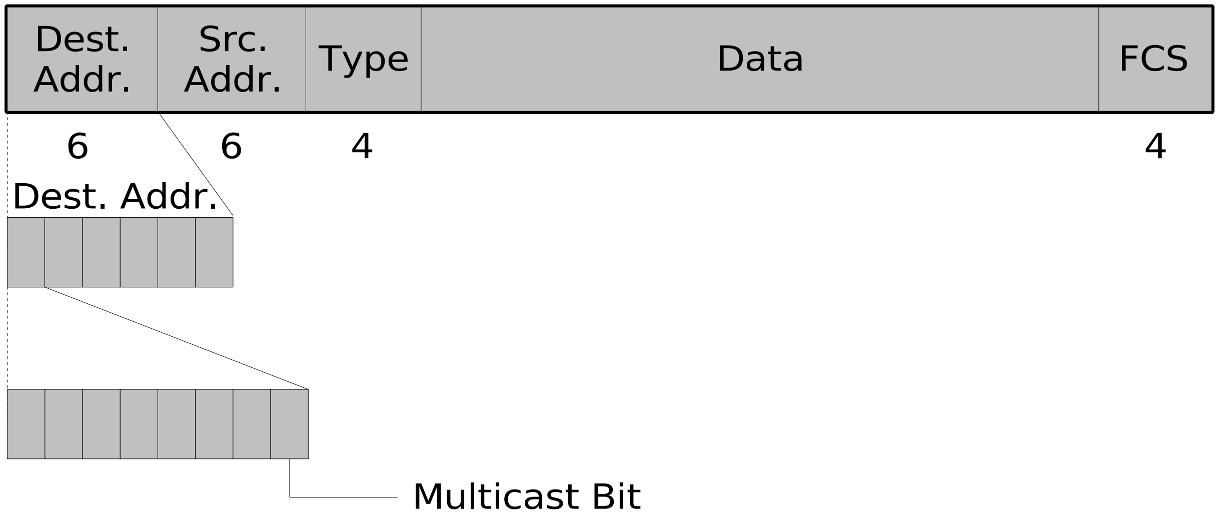
- Device-level multicast processing is critical point for IPv6.
- Some drivers did/does not set up the multicast filter appropriately.
  - Why?
    - Because ARP for IPv4 uses broadcast (ff:ff:ff:ff:ff:ff) only.

Device-level Group Management Issues (cont'ed)

- How to fix:
  - Set up “multicast” filter appropriately.
    - preferred if possible.
  - Change the mode to “ALLMULTI” (or “PROMISC”) mode if dev->mc_count is not zero.
    - Several drivers were fixed by this way.
    - Note: This will result in the device entering “ALLMULTI” (or event “PROMISC”) mode because IPv6 stack joins at least 2 multicast addreses.
      - all-node multicast address
      - solicited node multicast address
    - Note 2: Even a rich device usually needs this approach as well in the case of the group size exceeds its maximium group filter size.

PROMISC vs ALLMULTI

- PROMISC: Accepts all packets
- All Multi: Accept packets with the "Multicast" bit set
  - Poor hardwares do not have this feature.



- Device-level group management
- Group management API / protocols
- Packet forwarding
- Tree / routing management
- Tunneling

Group Management API / Protocols

- Per-socket API
  - Socket options
  - inet6_sk(sk)->ipv6_mc_list
- Per-interface API
  - idev->mc_list

Group Management API / Protocols (cont'ed)

- Inter-node protocols
  - MLDv1 / MLDv2
    - Exchange group information between nodes
    - MLDv1: Any-source Multicast (ASM) only
    - MLDv2: ASM and/or Source-Specific Multicast (SSM)
  - (LW-MLDv2: lightweight version of MLDv2)

Group Management API / Protocols Issue

- Per-interface multicast list is inefficient
  - Single linked list...
  - This list is scanned (at least) per once per multicast packet!
    - net/ipv6/ip6_input.c: ip6_mc_input()
      - check if we should accept the packet (rough filter)
    - net/ipv6/ip6_input.c: ip6_input_finish()
      - check if we should deliver the packet to the upper layer.
    - net/ipv6/ip6_output.c: ip6_output2()
      - check if we should loop'ed back the outgoing packet for our own listener(s).

Group Management API / Protocol Issue (cont'ed)

- How to fix?
  1. Introduce “host entry” for group address we are listening to. Standard routing engine will find this most specific entry, and stores it into skb->dst. Later we can determine if group address is interesting or not by cheking if the skb->dst is “host entry” or not. (ip6_mc_input2()) (O(n) -> O(log n))

Group Management API / Protocols Issue (Cont'ed)

2. We could attach a pointer for corresponding group entry (ifmcaddr6{}) from the “host entry.” We could eliminate walking over the group list again to check group source address. (ip6_mc_input(), ip6_route_output2())
3. We may be able to omit idev->mc_list by replacing walker over whole entries. (igmp6_event_query())
4. We could introduce hash for source address filters.

- Device-level group management
- Group management API / protocols
- Packet forwarding
- Tree / routing management
- Tunneling

Multicast Forwarding

- 2.6.25-
  - Merged!
  - Similar to net/ipv4/ipmr.c
    - Based on patch from Mickael Hoerdt.
  - Uses hash table for cache
    - Possibly it can be merged into routing table, but rather hard…
  - Supports PIM-SM (SSM) only.
    - ASM is not supported.

Multicast Forwarding

- 2.6.29
  - Namespace support
  - Many fixes

- Device-level group management
- Group management API / protocols
- Packet forwarding
- Tree / routing management
- Tunneling

Tree / Routing Management

A.k.a. “multicast routing”

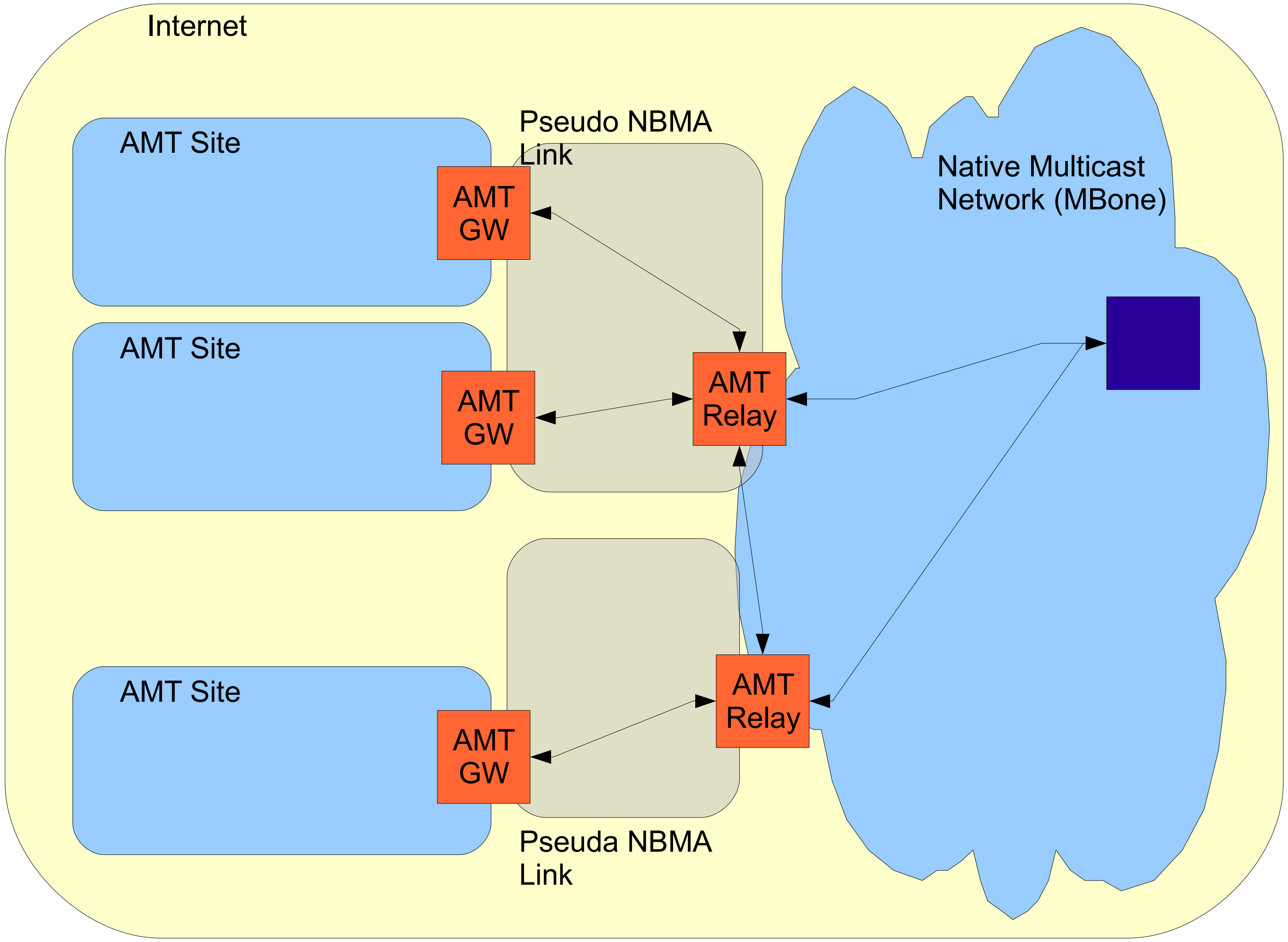
- PIM-SM (SSM)
  - mcast-tools (pim6sd)
  - XORP
- Proxy

- Device-level group management
- Group management API / protocols
- Tree / routing management
- Packet forwarding
- Tunneling

Tunneling

- AMT (Automatic Multicast Tunnel)
  - To create tunnel between multicast-enabled/capable cites across non-multicast links (such as p-t-p link, traditional internet etc.) automatically.
  - Isolated site can receive/send multicast traffic.
    - Except sending ASM from isolated site.
  - IPv4-only version is available. IPv6 version is being implemented.

AMT


Internet
AMT Site
AMT GW
Pseudo NBMA Link
Native Multicast Network (MBone)
AMT Site
AMT GW
AMT Relay
AMT Site
AMT GW
AMT Relay
Pseuda NBMA Link

Appendix: Other Progress / News in Linux IPv6

- Namespace
- More enhancements on specification conformity (for IPv6 Ready Logo Program)
  - Core, Mobile IPv6
- Protocol-independent connection tracking is now non-EXPERIMENTAL.
- USAGI Project is now voluntarily driven.
  - UMIP (USAGI-patched Mobile IPv6) was launched. New release is planned.

Conclusion

- Multicast forwarding is now available.
  - Some issues will be fixed soon.
  - Overhead of multicast packet processing will be reduced.
- Other progress
  - Namespace is available.
  - Mobile IPv6 passed Conformance Test.

Thank you