A Common API for Transparent Hybrid Multicast

Matthias Wählisch, Thomas C. Schmidt
Stig Venaas

{waehlisch, t.schmidt}@ieee.org,
stig@cisco.com
Problem Statement

- Group communication is implemented on different layers and is based on different technologies
  - This results in several forwarding paths and varying group addresses (namespaces)

Objectives:

1. Enable any application programmer to implement independent of underlying delivery mechanisms
2. Make applications efficient, but robust w.r.t. deployment aspects
Requirements

- Design of a common group communication API
- Flexible namespace support in group addressing
  - Separate routing and addressing scheme from application design
- Mapping between different namespaces
- Gateway function to forward multicast data between different technologies
- Consistent view on multicast states at a single host
Reference Scenarios

- Domains running same technology but remaining isolated
- Domains running distinct technologies but hosts are members of the same group
Overview

- Extended multicast functions implemented by a middleware
- Middleware
  - Provides extended API
  - Bridges data between technol.
- **General procedure**
  1. App. subscribes/leaves/sends to a **logical group ID**
  2. Middleware maps logical ID to **technical group ID**
  3. Technical ID is allocated or revised if already in use
Namespace Issue (or Challenge …)

Scenario: Two (or more) different addresses in different namespaces may belong to

(1) the same multicast channel (same technical ID)
(2) different multicast channels (different technical IDs)

- Can be solved based on a invertible mapping
  - Does not hold in general (cardinality of namespaces)
  - Example: Mapping IPv6 to IPv4
Assumptions

- Assumptions:
  - All group members subscribe to the same logical group ID from the same namespace
  - Domain composition and node attachment to specific technology remain unchanged during multicast session

- Problem: Traditional applications
  - Inter-domain multicast gateway bridges data
Send/Receive Calls – Required for Endhosts and Gateways

- **Mode**: Defines multicast technique
  
  - init(in Namespace n)
    - Pre-initializes the namespace for a group
  
  - join(in Address a, in Mode m)
    - Subscribes to a group
  
  - leave(in Address a, in Mode m)
  
  - send(in Address a, in Mode m)
Service Calls – Required for Gateways

- groupSet(out Address[] g, in Mode m)
  - Returns all registered multicast groups
- neighborSet(out Address[] a, in Mode m)
  - Returns the set of multicast neighbors
- designatedHost(out Bool b, in Address a)
  - Checks if the host is designated router
- updateListener(out Address g, in Mode m)
  - Upcall informs about change of listener states
- updateSender(out Address g, in Mode m)
  - Upcall informs about change of source states
Open Issues

- Mapping service (e.g., DHT)
- Encoding of routing addresses and technologies at the mapping service
- ASM service via SSM delivery
- Any scenarios not covered by the draft/API?
Conclusion

- API enables technology-agnostic programming of group-oriented applications
- API can be used to implement hybrid multicast gateway
  - Draft describes interaction with IP-layer multicast routing protocols (PIM-SM etc.)