QS-XCAST: A QoS Aware XCAST Implementation

Elisha Abade, K. Kaji, N. Kawaguchi
Nagoya University, Japan
Introduction

• Multipoint communication
  – One source to many receivers
  – Application areas
  – Protocols
    • Multiple unicast, Multicast,

• Multicast
  – IP Multicast
  – Application Layer Multicast (ALM)
Multipoint Communication

APPLICATION AREAS:

1. Videoconferencing
2. IP Television

- Can be simplified using Multicast technology
- Multicast: Bandwidth efficiency
- Multicast deployment in global scope is challenging
- XCAST was proposed
XCAST

- XCAST: (Specifications in RFC 5058)
  - explicit multiunicast
  - List of destinations embedded in IP header
  - Routing - unicast route tables
  - Not yet fully investigated

Complementary to IP multicast model:

- IP multicast:
  - Scales with the number of receivers
- XCAST:
  - Scales with the number of groups
  - No per-session signaling and state information
XCAST

A – Sender.
B, C, D, E - Receivers

Packet from A:
Embeds ALL destinations
Has a bitmap

Router Operations:
- Table lookup for next-hops
- Grouping of destinations
- Packet replication
- Updating of the bitmaps
- Forwarding of packet copies
Motivation

• Need to deploy XCAST6 in real-world.
  – Existing routers are not XCAST-aware
  – Using Testbeds: Scale can be limited by time and resources available.
  – No Significant research on XCAST QoS
  – Existing simulators do not have XCAST routing model

• XCAST Simulation models are needed:
  – XCAST header is already complex
  – Alternative way to make XCAST QoS aware
  – Differentiated Architecture provides an option.
OMNeT++

• Generic:
  – Modeling any system where the discrete event approach is suitable.
  – Communication networks, Queuing systems etc

• QoS using DiffServ Architecture:
  – Only basic Implementation exists in OMNeT++

• Enhancements:
  – Implement XCAST6
  – Extend Basic DiffServ
  – Integrate DiffServ with XCAST6
QoS Provisioning

• Using DiffServ Architecture (RFC 2474, 2475)
• A defined set of building blocks
  – A small bit-pattern in IP packets (IPv4, IPv6)
  – 6-bit DS field (DSCP)
  – Forwarding treatment (Per-Hop-Behavior)
  – Classification and QoS revolve around DSCP
  – Hierarchical organization of nodes
    • (Core routers, Edge routers, End hosts)
  – Concept of domains (DiffServ domains)
  – Packet Marking
  – Admission Control
DiffServ Architecture

- Per-Hop-Behavior
  - Expedited Forwarding (EF) – RFC 2598,
  - Assured Forwarding- AF, RFC 2597.
    - (AFxy) – x - classes, y - drop precedence
  - Default (Best Effort – BE) – RFC 2474
The INET Framework

**INET Framework Architecture**

**EXTENSIONS**
INET MANET, OverSim, xMIPv6, ReaSE, INET/Quagga, HIPSim, VoIPTool, HTTPTools

**UTIL**
Tcpdump, ThruputMeter.

**PROTOCOL STACK**

**INET Base Modules**
IPv4, ICMPv4, ARP, IPv6, ICMPv6, MIPv6, HIP, PPP, Ethernet

**Concept:**
- Modules
- Messages

**Communication:**
- Message passing

**Modules:**
- Protocols
- Data holders
- Extra Objects

**Protocols:**
- Behavior implemented in Simple modules
- Defined in C++ code
- Both wired and wireless
Implementing XCAST in OMNeT++

**XCAST Protocol:**
- Application layer
- Transport layer
- Network layer

**Application Layer:**
- Destination hosts

**Network Layer:**
- XCAST has significant impact here
- Understanding packet structure
- Routing decisions to pass to routing protocols
Implementing XCAST in OMNeT++

- Network Layer Modules:

- IPv6 Class:
  - Invokes:
    - Routing decisions made here
    - Neighbor Discovery
    - Data delivery (to Transport)
  - Marked as Work In Progress

- IPv6 Extension Header:
  - Incomplete (OMNeT++ 4.1):
    - Only Class Declarations
    - Needed by XCAST6
Implementing XCAST in OMNeT++

- Network Layer Modules:

IPv6 Module:
- Destination List container
- Bitmap container
- Redefined `handleMessage()`
- New: `routeXcastPackets()`
- XCAST Statistics:
  - Dropped packets
  - Replications

IPv6ExtensionHeader:
- Completed:
  - Routing Extension header
- Introduced:
  - List of destinations
  - XCAST Bitmap
Implementing XCAST in OMNeT++

- Network Layer Modules:

IPv6ProtocollId
IPv6 IPv6Datagram IPv6ErrorHandling IPv6Extension Headers
RoutingTable6 IPv6ControlInfo

**IPv6ControlInfo:**
- Currently support single address:
- For XCAST6 Support:
  - List of destinations
  - Bitmap container
  - Traffic class holder

**IPv6Datagram:**
- Methods to handle:
  - Routing Extension header
  - Traffic Class
  - New IPv6ControlInfo
Implementing XCAST in OMNeT++

- **Network Layer Modules:**

  **IPv6FlatNetworkConfigurator:**
  - All host in same network
  - No support for subnets
  - Our approach:
    - NETCONF-style XML file for
    - IP addresses & Routing

  **RoutingTable6:**
  - Added: NETCONF XML processing
  - Initialization stage 3 invokes:
    - `parseXMLConfigFileForStaticRoutes()`
    - `addDefaultRoute()`
    - `addStaticRoute()`
Implementing XCAST in OMNeT++

- **Transport Layer**
  - *UDPControllInfo*
    - Destination: ALL_XCAST_NODES ("ff0e::114")
    - *UDPControllInfo and IPv6ControllInfo* exchange information across protocol layers

- **Application Layer**
  - XCAST6 Model application
    - Based on UDPBasicAPP
    - Selects a group and sends data to ALL members

- **Statistics Collection**
  - Dropped packets
  - Propagation delay
  - Number of replications etc
XCAST-DiffServ Integration

• DiffServ QoS tasks:
  – Classification,
  – Marking and
  – Shaping
• XCASTQoSClassifier
  – Inherits from IQoSClassifier Base Class
  – Implements 14 PHBs
  – Works with DropTailQoSQueue
Simulation

- IPTV network
- Hierarchically
- Core routers – Provider network,
- Edge routers – Connecting clients

- IPTV Plans (For pricing & QoS)
  - Platinum - EF
  - Gold - AF41
  - Sliver - AF31
  - Bronze - AF21
  - Delux - AF11
  - Economy - BE

- Metrics
  - Throughput
  - Average per hop delay
Performance Evaluation

- Average Throughput
Performance Evaluation

- Average per-hop delay

![Bar chart showing average per-hop delay for different DSCP-PHB classes: EF, AF41, AF31, AF21, AF11, BE. The chart compares multicast (XCAST) and unicast (UNICAST) scenarios.]
Performance Evaluation

- Multiple DiffServ Domain

![Diagram of network architecture involving source, transit domain, bandwidth broker, and receiver domain, with resource allocation request (RAR) and resource allocation answer (RAA) indicated.](image)

![Graph showing the time of RAA arrival (ms) against the number of receivers, with 'XCAST' and 'UNICAST' lines.](image)
Conclusion and Future Work

• This work:
  – Shows how to implement XCAST6 in OMNeT++
  – Shows XCAST6 QoS provisioning using DiffServ Architecture
  – Focuses on key classes of INET Framework
  – We hope it opens up XCAST QoS research.
  – Source code available in Sourceforge.

• Future Work:
  – To investigate Challenges in XCAST QoS provisioning using DiffServ Architecture.