Scalable Application Layer Multicast

Suman Banerjee
Bobby Bhattacharjee
Christopher Kommareddy

http://www.cs.umd.edu/projects/nice
Group Communication

Network-layer Multicast
Replication at routers

Sequence of Direct Unicasts
Replication only at source
Application-layer Multicast

Examples:
- Narada, Yoid, Gossamer, HMTP, Scribe, Bayeux, CAN-multicast, DT, …
- NICE

Replication at end-hosts
Application-layer Multicast

Replication at end-hosts

Metrics

- Tree Quality
- State / Control Overheads
- Robustness
Talk Outline

- Introduction
- NICE Application-layer Multicast Protocol
- Results
- Conclusions
NICE Application-layer Multicast

- Scales to large group sizes
  - Low average and worst case control overheads
  - Does not compromise tree quality or robustness

- Even low-bandwidth applications are efficient
  - Web tickers

Uses a hierarchy
NICE Topologies

- **Control topology**
  - Detects host failures and re-structure the overlay

- **Data delivery topology**
  - Basic path: Implicitly defined by the hierarchy
  - Can be independent of the control path
NICE Hierarchy

A Set of Members
NICE Hierarchy

Clusters
- Non-overlapping
- Proximity-based
- Size: $k$ to $3k-1$
NICE Hierarchy

Clusters
- Non-overlapping
- Proximity-based
- Size: $k$ to $3k-1$

Layer 0

Graph-theoretic center is the cluster leader
NICE Hierarchy

Leaders form the higher layer and repeats
NICE Hierarchy

Layer 2

log N layers
Control Topology

- Soft state about all cluster peers
- HeartBeats
Control Topology

- Soft state about all cluster peers
- HeartBeats

State and Control message overheads:
- Average: Constant
- Worst case: $O(k \log N)$
Basic Data Path
Basic Data Path
Basic Data Path
Basic Data Path
NICE Invariants

- Cluster sizes between $k$ and $3k-1$

- Cluster leader is the central member
  - Leaders for next higher layer

NICE protocol maintains these invariants
NICE Protocol Operations

- Cluster Retire
- Cluster Merge
- Cluster Split
- Member Depart
- Member Join
Join Procedure

Assume a Rendezvous Point
Join Procedure
Join Procedure

Diagram:
- Node C0
  - Connects to B0 and B1
- Node B1
  - Connects to B0
- Node B0
- Node B2
- Node RP
- L2: {C0}
- A3
Join Procedure
Join Procedure

L1: \{B0, B1, B2\}
Join Procedure

Join L0

A3

B2

B0

C0

B1

RP
Join Procedure

Diagram showing nodes labeled B0, B1, C0, RP, A3, L0: {...} connected by arrows.
Join Procedure

Overhead: $O(\log N)$ RTTs and $O(k \log N)$ messages
- Optimizations possible
Cluster Split

Cluster size: 4 to 11
Cluster Split

B0 – C0 – B1

B2
Cluster Split
Cluster Split

• Split into two new clusters

• Each new cluster has at least $3k/2$ members
Cluster Split
Cluster Split
Cluster Split
Results

- Simulations
  - 10,000 node Transit-Stub graphs
  - Group sizes up to 2048
  - Comparisons with Narada [CMU]

- Wide-area Experiments
  - Members at 8 sites
  - Group sizes up to 96

- Dynamic joins and (ungraceful) leaves

- Constant rate data source
Evaluation Metrics

- **Tree Quality: Stress**
  - Number of copies of the same data packet on a link/router
  - Example: Stress on link [A-1] = 2

- **Tree Quality: Stretch**
  - Ratio of the overlay latency to the direct unicast latency
  - Example: Stretch for receiver D = 5/3

- **State at end-hosts**
  - Control overheads

- **Robustness**
  - Host failures
Example Scenario

128 members join

16 members leave within 10 seconds

Time (in seconds)
Tree Quality: Stress

First 200 seconds ...

Resource usage at links

128 end-hosts join

128 Join

Narada-5

NICE

Average link stress

Time (in secs)
Tree Quality: Stretch

End-to-end latency to receivers

First 200 seconds ...

128 end-hosts join

Average receiver path length (hops)

Time (in secs)
Failure Recovery

After 1000 secs

Periodic leaves in sets of 16

Fraction of hosts that correctly received data

Time (in secs)

16 x 5 Leave

---

NICE

Narada-5
Control Overheads

Control traffic bandwidth at the access links

- Narada-5 (Avg)
- NICE (Avg)

Join

16 X 5
Leave

Time (in secs)
## Control Overheads

<table>
<thead>
<tr>
<th>Group Size</th>
<th>Narada-30</th>
<th>NICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>32</td>
<td>9.23</td>
<td>1.03</td>
</tr>
<tr>
<td>128</td>
<td>65.62</td>
<td>1.19</td>
</tr>
<tr>
<td>512</td>
<td>199.96</td>
<td>1.93</td>
</tr>
<tr>
<td>1024</td>
<td>-</td>
<td>2.81</td>
</tr>
<tr>
<td>1560</td>
<td>-</td>
<td>3.28</td>
</tr>
<tr>
<td>2048</td>
<td>-</td>
<td>5.18</td>
</tr>
</tbody>
</table>

Bandwidth overheads averaged over all network routers
Wide-area Testbed

A: cs.ucsb.edu
B: asu.edu
C: cs.umd.edu
D: glue.umd.edu
E: wam.umd.edu
F: umbc.edu
G: poly.edu
H: ecs.umass.edu
Failure Recovery

Includes the effects of network losses

Distribution of losses for packets in random membership change phase

Fraction of hosts that correctly receive data

64 members
Average member lifetime = 30 secs

Time (in secs)
Failure Recovery

Includes the effects of network losses

Cumulative distribution of losses at members in random membership change phase
Related Work

- Mesh-first
  - Narada, Gossamer

- Tree-first
  - Yoid, HMTP

- Implicit
  - Scribe, Bayeux, CAN-multicast, Delaunay-Triangulation

“A Comparative Study of Application Layer Multicast Protocols”, S. Banerjee and B. Bhattacharjee
- Available at: http://www.cs.umd.edu/~suman/publications.html
Current Work

- Detailed analysis of tree quality
  - Stress and stretch

- Implementing applications
  - Video delivery
Conclusions

- NICE scales to large member groups
  - Low control overhead
  - Does not sacrifice tree quality or robustness

- Scalability using hierarchy

http://www.cs.umd.edu/projects/nice