Resilient Multicast using Overlays

Aravind Srinivasan
Suman Banerjee, Ryan Braud, Seungjoon Lee, Bobby Bhattacharjee

University of Maryland
Data delivery to a group of hosts

- Network-layer Multicast
- Application-layer multicast
Overlay Node Failures
Overlay Node Failures
Overlay Node Failures
Overlay Node Failures

- Network losses are transient
- Overlay node failures are persistent until detected and repaired
  - ~ 10s of seconds
Resilience

High delivery ratio
Have latency requirements
Streaming audio or video applications

Solution: Redundant, low-overhead data paths
Probabilistic Resilient Multicast

- Randomized Forwarding
  - Handles overlay node failures
  - Proactive

- Triggered NAKs
  - Handles network losses
  - Reactive
Randomized Forwarding
Randomized Forwarding

Each node chooses a few other cross tree edges at random and forwards data with low probability.
Randomized Forwarding

Each node chooses a few other cross tree edges at random and forwards data along them with low probability.
Randomized Forwarding

Each node chooses a few other (r) cross tree edges at random and forwards data with low probability ($\beta$)

Uses a low overhead random node discovery mechanism.
Overheads

- Data overhead due to duplicates
- Detected and suppressed

Typical overheads \((r \beta) = 3\%\)

Random forward probability = 0.01
Number of random neighbors = 3
It performs very well!

- Random choices help
- Larger the affected region, greater the number of cross edges incident on it
- Increases resilience against node failures
Analysis

With high probability,

– All the non-leaf nodes that did not fail successfully get data.

– A large fraction of leaf nodes that did not fail successfully get data. (e.g. 97%)
Data Delivery Ratio

Overhead 3%

link loss=0.5%, group change=1/second, deadline=8 sec, 512 hosts
Maximum Data Outage

Overhead 3%

link loss=0.5%, group change=1/sec, deadline=8 sec, 512 hosts

CDF

Maximum Gap (in second)

PRM 3,0.01

BE
Latency of Data Delivery

Overhead 3%

link loss=0.5%, group change=0.1/sec, deadline=8 sec, 512 hosts

CDF

Avg. Latency observed at nodes (in millisecond)
# Data Overheads

<table>
<thead>
<tr>
<th>Scheme, Change rate</th>
<th>Latency (s)</th>
<th>Data Delivery Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>80%</td>
</tr>
<tr>
<td>FEC 0.1/s</td>
<td>0.5</td>
<td>87-100</td>
</tr>
<tr>
<td></td>
<td>2.0</td>
<td>62-75</td>
</tr>
<tr>
<td></td>
<td>8.0</td>
<td>50-62</td>
</tr>
<tr>
<td></td>
<td>64.0</td>
<td>37-50</td>
</tr>
<tr>
<td>PRM 1.0/s</td>
<td>0.2</td>
<td>9-12</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>0-1</td>
</tr>
<tr>
<td></td>
<td>2.0</td>
<td>0-1</td>
</tr>
<tr>
<td></td>
<td>8.0</td>
<td>0-1</td>
</tr>
</tbody>
</table>
## Scalability

<table>
<thead>
<tr>
<th>Size</th>
<th>Control Overheads</th>
<th>Delivery Ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BE</td>
<td>PRM</td>
</tr>
<tr>
<td>128</td>
<td>2.9</td>
<td>4.0</td>
</tr>
<tr>
<td>256</td>
<td>3.3</td>
<td>4.4</td>
</tr>
<tr>
<td>512</td>
<td>3.4</td>
<td>4.7</td>
</tr>
<tr>
<td>1024</td>
<td>4.1</td>
<td>5.5</td>
</tr>
<tr>
<td>2048</td>
<td>5.8</td>
<td>7.4</td>
</tr>
<tr>
<td>4096</td>
<td>10.1</td>
<td>13.5</td>
</tr>
</tbody>
</table>
Experimentation on Internet

- Scalable Resilient Media Streaming System
  - MPEG4IP player
  - MPEG-4 movie clip streamed cyclically from Darwin Media Streaming Server
  - PRM-enhanced NICE application-layer multicast

- Tested on Emulab and RON testbed
  - RON testbed: 32 hosts in USA, Canada, Europe, Asia

- Dynamic joins and leaves of clients
Implementation

PRM-based media streaming system implemented and tested on the Internet

From apple.com
Darwin Media Streaming Server

Designated Source

- Based on RTP
- Interoperates with any:
  - Streaming server
  - Playback client

From sourceforge.net
MPEG4IP player

Integrated Client

Proxy-based Client
Data Delivery Ratio

PRM incurs 3% overhead
Group change: 4.8/min
Aggregate Delivery

PRM incurs 3% overhead
Group change: 4.8/min
Summary

- NICE-resilient multicast
  - Low overhead
  - High data delivery
  - Efficiently scales to large groups

- Implementation and Experimentation on the Internet
  - Implementation of a media streaming service
  - Experiments on Emulab and RON testbed