Getting higher in the SpoVNet stack…

- Services in SpoVNet
  - Reside above Underlay Abstraction
  - Offer interface to applications
  - Several services as part of the architecture
    - MCP-O (Data Dissemination)
    - ES (Event Service)
    - Security Service
  - Focus here: MCP-O
Goal: Efficient Data Dissemination

- **Group Communication** in SpoVNet instances
  - Solution: Application Layer Multicast (ALM)
- Several existing approaches
  - Narada, NICE, Yoid, Nemo, ...
  - Each with different target applications
- Drawbacks
  - Considering *homogeneous* (UDP-)underlays
  - Measure link property themselves
    - Mostly latency

**SpoVNet Vision**
- Global service provision
  - Heterogeneity of nodes and network!
- Different application demands

---

**NICE**

- Achieve scalability through **hierarchical clustering**
  - Limit protocol overhead
- Consider node distances (n-2-n latencies)
- Logical structure determines date dissemination
MCP-O Protocol Design

Basic idea (enhance NICE approach to…)
- Evaluate application demands for structural design decisions
- Consider Cross-Layer-Information (provided by CLIO comp.)
- Build efficient dissemination topologies for every use case
- Use underlay functionality where possible

Service Demands Adaption

Use **Service Metric** to affect building of structure

\[
d : K \times K \rightarrow \mathbb{R}_{\geq 0}
\]

\[
d : (x, y) \mapsto \sum_i a_i f_i(x, y)
\]

- Weights \(a_i\) determined through application requirements
- \(f_i(x,y)\) e.g. latency, same WiFi domain, same IP Multicast domain, …
Service Demands Adaption

- But: Hierarchical approach not always the best choice
  - Good scalability, but cutting back e.g. latencies 😊
- Example: Latency-sensitive app with < 16 Members
- Adapt cluster size $K$
  - $K = \frac{\text{lowest current upstream}}{\text{data stream}}$
  - Unicast brings best achievable latency, protocol decides reasonable overhead in bounds

Examples for Protocol Behavior

- Single Source Videostreaming application
  - Locate High-Bandwidth nodes near source
  - Make them cluster leader, upstream determines cluster size
    - Flat tree to bound latency and jitter
- Realtime Game
  - Latency critical factor
  - Use unicast where feasible
WiFi Integration

- First Steps: Wireless Integration
- Save transmission overhead through broadcast
  - NICE-WLI
- Introduce Gateway-Nodes for mediation
  - WiFi domain not part of structure, but connected through Gateway
  - Prevent Gateways from becoming cluster leaders (save overhead)

Approach saves data transmission overhead
- Figure: $P(\text{WLAN})$ denotes probability of WiFi node

Traffic per node

<table>
<thead>
<tr>
<th>Node count</th>
<th>0</th>
<th>10</th>
<th>20</th>
<th>30</th>
<th>40</th>
<th>50</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bytes</td>
<td>0</td>
<td>10000</td>
<td>20000</td>
<td>30000</td>
<td>40000</td>
<td>50000</td>
</tr>
</tbody>
</table>

Network heterogeneity in global ALM provision.
Summary & Outlook

Efficient data transmission for SpoVNet applications through
- Considering node and network heterogeneity
- Use appropriate dissemination strategies
  - IP Multicast, broadcast, QoS reservations,…

WiFi support already integrated

Further Work
- Multiple group support per instance
- Usage of underlay features