COMMON:
Coordinated Multi-layer Multi-domain Optical Network Framework for Large-scale Science Applications (2010-2013)

PI: Dr. Vinod Vokkarane
Associate Professor, University of Massachusetts at Dartmouth
(Sabbatical: Visiting Scientist, MIT)
Contact: vvokkarane@ieee.org
Project Website: http://www.cis.umassd.edu/~vvokkarane/common/

Annual PI meeting for the ASCR Network & Middleware
Supported by DOE ASCR under grant DE-SC0004909

March 1-2, 2012
# COMMON Project Team

- **UMass Team**
  - Vinod Vokkarane (PI)
  - Arush Gadkar (Post-Doc)
  - Joan Triay (Visiting Scholar)
  - Bharath Ramaprasad (MS)
  - Mark Boddie (BS-MS)
  - Tim Entel (BS-MS)
  - Jeremy Plante (Ph.D.)
  - Thilo Schoendienst (Ph.D.)

- **ESnet Team**
  - Chin Guok
  - Andrew Lake
  - Eric Pouyoul
  - Brian Tierney
Outline

• Introduction and project objectives

• Year 1 Project Objectives:
  – Anycast Multi-domain Service
  – Multicast-Overlay Algorithms

• Year 2 and 3 Project Objectives
  – Multi/Manycast-Overlay Deployment
  – Survivable Connections
  – QoS Support (with ARCHSTONE project)
Point-to-Point Communication Services: Unicast, Anycast

- Unicast Vs Anycast

  - Unicast request: $(s,d)$
  
  - Anycast request: $(s,\{D\})$ where $s$ is the source and the $\{D\}$ is the set of candidate destinations.

  - Anycast: the source communicates with any one node from the set of candidate destinations.

Example:

- Unicast: $(S, d_4)$
- Anycast: $(S,\{d_3,d_4\})$
- Anycast: $(S,\{d_3,d_4,d_5\})$

Note: other common request parameters such as, request duration, start time, end time, bandwidth requested are not shown on the slide.
P2MP Communication Services: Multicast/Manycast

- **Multicast Vs Manycast**
  
  - **Multicast request:** \((s, \{D\})\), where \(s\) is the source node and \(D\) is the set of destination nodes \(\{d_1, d_2, ..., d_m\}\).
  
  - In Multicast, source node communicates with each destination node in \(\{D\}\).

- **Manycast request:** \((s, \{D\}, k)\), where \(s\) is the source node and the \(\{D\}\) is the set of candidate destination nodes.

- In Manycast, source node communicates with any \(k\) nodes in \(\{D\}\).

**Example:**

- **Multicast:** \((S, \{d_3, d_5\})\)
- **Manycast:** \((S, \{d_3, d_4, d_5\}, 2)\)  
  
  (Note: other common input parameters omitted)
COMMON Project Objectives

• Design and implement new services, such as anycast, multicast, manycast, survivability, and QoS across multiple domains and multiple layers.

• Year 1:
  – Design and Deploy Anycast service on OSCARS.
  – Develop Multi/Manycast Overlay models.

• Year 2:
  – Deploy Multi/Manycast Overlay models on OSCARS.
  – Design and Deploy survivability techniques on OSCARS.
  – Design QoS mechanisms to support scientific applications on multi-domain networks.

• Year 3:
  – Extend the survivability and QoS mechanisms to multi-layer multi-domain scenarios and deploy them on OSCARS.
Year 1: Deployment of Anycast Service on OSCARS

**Objectives**
- Design and implement a production-ready anycast service extension to existing OSCARS framework.
- Improve connection acceptance probability and user experience for anycast-aware services.

**Impact**
- Provide scientific community with ability to:
  (a) Allow for destination-agnostic service hosting on large-scale networks.
  (b) Increase service acceptance.

**Design & Implementation (Complete)**
- Designed anycast service as a PCE extension.
- Implementation of the PCE modules to find anycast connectivity, remove the unavailable resources, and select the best possible destination.
- Successfully completed Stress, Regression, and Integration testing of the anycast modules on OSCARS 0.6 (Q4, 2011).
- Hot deployment ready (PnP capable) anycast version of OSCARS 0.6 available at: [https://oscars.es.net/repos/oscars/branches/common-anycast/](https://oscars.es.net/repos/oscars/branches/common-anycast/)
- Year 2: Plan to work with ESG group to attach this service to a specific application.
- Looking forward to work with other application groups.
OSCARS Anycast Design

Notification Broker
• Manage Subscriptions
• Forward Notifications

AuthN
• Authentication

Web Browser User Interface

AuthZ*
• Authorization
• Costing

*Distinct Data and Control Plane Functions

Coordinator
• Workflow Coordinator

Lookup
• Lookup service

Topology Bridge
• Topology Information Management

PCE
• Constrained Path Computations

Resource Manager
• Manage Reservations
• Auditing

Path Setup
• Network Element Interface

IDC API
• Manages External WS Communications
The user interface servlets would process the anycast request as a unicast request with a big exception: the destination field will be a list of destination nodes (the anycast destination set).

- An option is to encapsulate the anycast data as an `OptionalConstraintType`, in addition to the rest of parameters mapped into a `UserRequestConstraintType`. Both, `UserRequestConstraintType` and the `OptionalConstraintType` will be part of the `ResCreateContent`.

- The `ResCreateContent` will be passed to the Coordinator to further process the anycast request.

The Coordinator, through the `CreateReservationRequest`, will get the `ResCreateContent` and map the user request constraints and optional constraints into a `PCEData` object.

- The `PCERuntime` will handle the query process to the `PCE`.

The `PCE` (using the design proposed in the following slides) will make use of the `OptionalConstraintType` (which carries the list of destinations).

The result of the `PCE` will be the path from the source node to a single destination node, so, from the path reservation and PSS modules standpoint, the rest of the flowchart will work as a unicast request.
Multi-Domain Anycast Demo

Unicast Hop Count = 4
Anycast 2|1 Hop Count = 3
Anycast 3|1 Hop Count = 2
## Anycast 3|1 Illustration

**On-demand Secure Circuits and Advance Reservation System**

A collaboration between ESnets Internet2, DANTE, and ISI East

### Reservation Creation Form

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Source</td>
<td>um:ofg:network:domain=umassd:netnode=andromeda:port=902:link=*</td>
</tr>
<tr>
<td>Destination</td>
<td>um:ofg:network:domain=es.net:netnode=anycast://bl-mr2@xe-1/3/0@<em>,sdsc-sdn2@xe-1/3/0@</em>,hous-sdn1@xe-1/3/0@*</td>
</tr>
<tr>
<td>Bandwidth (Mbps)</td>
<td>100</td>
</tr>
<tr>
<td>Description</td>
<td>Anycast 3</td>
</tr>
</tbody>
</table>
### Anycast 3|1 Illustration

**Reservation details for umassd.net-154**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NEW GRI</strong></td>
<td></td>
</tr>
<tr>
<td><strong>QUERY</strong></td>
<td></td>
</tr>
<tr>
<td><strong>GRI</strong></td>
<td>umassd.net-154</td>
</tr>
<tr>
<td><strong>Status</strong></td>
<td>FINISHED</td>
</tr>
<tr>
<td><strong>User</strong></td>
<td>umassd.net</td>
</tr>
<tr>
<td><strong>Description</strong></td>
<td>Anycast 3</td>
</tr>
<tr>
<td><strong>Start date</strong></td>
<td>1/7/2012</td>
</tr>
<tr>
<td><strong>Start time</strong></td>
<td>14:52</td>
</tr>
<tr>
<td><strong>End date</strong></td>
<td>1/7/2012</td>
</tr>
<tr>
<td><strong>End time</strong></td>
<td>14:55</td>
</tr>
<tr>
<td><strong>Created time</strong></td>
<td>2012/01/07 14:52</td>
</tr>
<tr>
<td><strong>Bandwidth (Mbps)</strong></td>
<td>100</td>
</tr>
<tr>
<td><strong>Source</strong></td>
<td>um.ogfnetworkdomain=umassd.netnode=andromeda.port=port-2link=*</td>
</tr>
<tr>
<td><strong>Destination</strong></td>
<td>um.ogfnetworkdomain=es.netnode=anycastlbi-mr2@xe-1/3/0/@,dscs-ndn2@xe-1/3/0/@,hous-sdn1@xe-1/3/0/@</td>
</tr>
<tr>
<td><strong>Local path</strong></td>
<td>3600 um.ogfnetworkdomain=es.netnode=alita-sdn1.port=xe-1/3/0/link=*</td>
</tr>
<tr>
<td></td>
<td>0.2-4094 um.ogfnetworkdomain=es.netnode=alita-sdn1.port=xe-0/10.0/link=xe-0/10.0</td>
</tr>
<tr>
<td></td>
<td>0.2-4094 um.ogfnetworkdomain=es.netnode=alita-sdn1.port=xe-7/0.0/link=xe-7/0.0</td>
</tr>
<tr>
<td></td>
<td>3600 um.ogfnetworkdomain=es.netnode=hous-sdn1.port=xe-1/3/0/link=*</td>
</tr>
<tr>
<td><strong>Interdomain path</strong></td>
<td>3600 um.ogfnetworkdomain=umassd.netnode=andromeda.port=port-2link=*</td>
</tr>
<tr>
<td></td>
<td>3600 um.ogfnetworkdomain=umassd.netnode=andromeda.port=port-3link=*</td>
</tr>
<tr>
<td></td>
<td>3600 um.ogfnetworkdomain=umassd.netnode=andromeda.port=port-4link=*</td>
</tr>
<tr>
<td><strong>Source VLAN</strong></td>
<td>3600</td>
</tr>
<tr>
<td><strong>Tagged</strong></td>
<td>true</td>
</tr>
<tr>
<td><strong>Destination VLAN</strong></td>
<td>3600</td>
</tr>
<tr>
<td><strong>Tagged</strong></td>
<td>true</td>
</tr>
</tbody>
</table>

**Contacts:** Chin Guk, Evangelos Chaniotakis, Andy Lake, Eric Pourcui

**Documentation:** ESN | Berkeley Lab | Notice to Users
Benefits of Anycast over Unicast OSCARS on live deployment

In summary, from the demo we observed the following:

1. Anycast as a communication paradigm for OSCARS eliminates or reduces blocking significantly when compared to using unicast.

2. Anycast as a communication paradigm for OSCARS significantly reduces average Hop Counts required to establish circuits when compared to unicast, thereby reducing network signaling considerably as well as utilizing fewer network resources.

3. Provisioning time (run-time complexity) for Anycast M|1 for 2 ≤ M ≤ 4 is comparable to that of Unicast as there is only a cumulative 2 second increase in provisioning time for an unit increase in cardinality of the Anycast set when compared to unicast.
Performance of Anycast Service for OSCARS

Results for single domain

- We simulated 30 unique sets of 100 AR requests (and present the average values).
- All links are bi-directional and are assumed to have 1 Gb/s bandwidth.
- For each request, the source node and destination node(s) are uniformly distributed.
- Request bandwidth demands are uniformly distributed in the range [100 Mb/s, 500 Mb/s], in increments of 100 Mb/s.
- All requests are scheduled to reserve, transmit, and release network resources within two hours such that we stress test the network by increased traffic loads in this time frame.
- The correlation factor corresponds to the probability that requests overlap during that two-hour window.

<table>
<thead>
<tr>
<th>Correlation</th>
<th>m=2</th>
<th>m=3</th>
<th>m=4</th>
<th>m=5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>48.11</td>
<td>50.94</td>
<td>49.06</td>
<td>51.89</td>
</tr>
<tr>
<td>0.3</td>
<td>29.59</td>
<td>29.34</td>
<td>32.65</td>
<td>33.93</td>
</tr>
<tr>
<td>0.5</td>
<td>24.81</td>
<td>29.92</td>
<td>33.08</td>
<td>35.04</td>
</tr>
<tr>
<td>0.7</td>
<td>14.30</td>
<td>21.57</td>
<td>22.02</td>
<td>22.25</td>
</tr>
<tr>
<td>0.9</td>
<td>14.44</td>
<td>17.91</td>
<td>20.77</td>
<td>21.21</td>
</tr>
<tr>
<td>1.0</td>
<td>11.43</td>
<td>15.76</td>
<td>16.16</td>
<td>18.20</td>
</tr>
</tbody>
</table>

Percentage blocking reduction of anycast $m/1$ over unicast.

16-node ESnet SDN core network topology used in obtaining results.
Results for multi-domain

- We simulated 5 unique sets of 50 AR requests (and present the average values).
- All links are bi-directional and are assumed to have 10 Gb/s bandwidth.
- Each request, has source node in ESnet and destination node(s) in GEANT.
- Request bandwidth demands are uniformly distributed in the range [1000 Mb/s, 5000 Mb/s], with step granularity of 1000 Mb/s.
- 2 inter-domain links between ESnet and GEANT.
- Remaining assumptions similar to single domain.

Average hop-count of successfully provisioned requests: unicast vs. anycast $m/1$.

<table>
<thead>
<tr>
<th>Correlation</th>
<th>$m=2$</th>
<th>$m=3$</th>
<th>$m=4$</th>
<th>$m=5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>66.67</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>0.3</td>
<td>35.71</td>
<td>50.00</td>
<td>57.14</td>
<td>57.14</td>
</tr>
<tr>
<td>0.5</td>
<td>15.79</td>
<td>21.05</td>
<td>31.58</td>
<td>31.58</td>
</tr>
<tr>
<td>0.7</td>
<td>4.00</td>
<td>12.00</td>
<td>12.00</td>
<td>12.00</td>
</tr>
<tr>
<td>0.9</td>
<td>3.57</td>
<td>10.71</td>
<td>10.71</td>
<td>10.71</td>
</tr>
<tr>
<td>1.0</td>
<td>6.67</td>
<td>10.00</td>
<td>10.00</td>
<td>13.33</td>
</tr>
</tbody>
</table>

Percentage blocking reduction of anycast $m/1$ over unicast.
Year 1-2: Deployment of Multi/Manycast Overlay on OSCARS

- Need for service to handle replicated data storage/retrieval.
- Data generated at a single site, distributed for study across multiple geographic locations.
- Fundamental obstacle: VPN (or Optical) Layer is point-point.
- Multicast and Manycast functionality must be implemented as a virtual overlay to the point-to-point VLAN (or optical layer).
### Year 1-2: Deployment of Multi/Manycast Overlay on OSCARS

#### Objectives
- To support point-to-multipoint connections.
- To develop an overlay model to support Multicast/Manycast communication paradigms over point-to-point unicast connections in OSCARS.

#### Impact
- Allow scientific community the ability to:
  - (a) Use a multicast service and increase the service acceptance.
  - (b) Provide different connection setup choices with different quality of service (QoS) to the scientists.

#### Progress
- Proposed three overlay models: Simple Overlay (MVWU), Drop at member node (DAMN) and Drop at any node (DAAN).
- Blocking performance results show significant improvement due to DAMN and DAAN algorithms.
- Integrated these overlay models into the OSCARS system (Year 2-3).
  - Simple Overlay (MVWU) implementation completed yesterday!
Year 1-2: Deployment of Multi/Manycast Overlay on OSCARS

• Three Multicast/Manycast overlay approaches proposed to provide point-to-multipoint (P2MP) communication over unicast-only optical/VLAN layer.
  – **MVWU – single logical-hop overlay**
    • Source of the multicast/manycast request establishes an independent lightpath (or VC) to each destination.
    • It is possible that these lightpaths overlap, thus making inefficient use of available bandwidth.
    • This can lead to unnecessarily high connection blocking.
  – **DAMN/DAAN – multiple logical-hop overlay**
    • Source of the request establishes a lightpath (or VC) to one destination.
    • The next destination can be reached by a lightpath (or VC) from the source or from the first destination.
    • Creates a Steiner tree routing scheme wherein each lightpath (or VC) may be viewed as a hop in the logical overlay layer.
• We plan to implement both overlay mechanisms on OSCARS.
Manycast Overlay Design (Best-Effort)

[1] User requests multicast connection in WBUI/API.

- Manycast group addressing scheme which uses a single MC address to represent several destinations.
- Requests forwarded to a MC aggregator responsible for breaking down a single MC request into multiple unicast requests.
- This aggregator is depicted as an independent module but may be more efficiently incorporated within API/WBUI modules.

[2] MC aggregator forwards unicast requests one at a time to Coordinator.

- Coordinator forwards requests along unaltered PCE stack to identify unicast VC for each request.
- If any VC cannot be reserved, entire MC request may be blocked.
- User may be able to specify a minimum number of destinations and request may continue as Manycast.
- The current prototype bypasses these last two steps to pursue best-effort VC establishment.
Manycast Request = (SUNN, {ATLA, KANS, WASH})

Note: Links *may* be shared by paths to different destinations. Routes use the shortest path from src to dest.
## Manycast Overlay Results

### Average VC Hop Count (Inter- + Intra-Nodal)

<table>
<thead>
<tr>
<th>Request Set</th>
<th>Unicast</th>
<th>Multicast - 2</th>
<th>Multicast - 3</th>
<th>Multicast - 4</th>
<th>Multicast - 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5</td>
<td>5.15</td>
<td>5.07</td>
<td>5.35</td>
<td>5.4</td>
</tr>
<tr>
<td>2</td>
<td>5.2</td>
<td>4</td>
<td>4.34</td>
<td>4.8</td>
<td>4.9</td>
</tr>
<tr>
<td>3</td>
<td>4.6</td>
<td>5.55</td>
<td>5.54</td>
<td>5.35</td>
<td>5.32</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>5.4</td>
<td>5.07</td>
<td>5.05</td>
<td>5.04</td>
</tr>
<tr>
<td>5</td>
<td>5.3</td>
<td>4.8</td>
<td>4.77</td>
<td>4.85</td>
<td>5.08</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>5.02</strong></td>
<td><strong>4.98</strong></td>
<td><strong>4.95</strong></td>
<td><strong>5.08</strong></td>
<td><strong>5.148</strong></td>
</tr>
</tbody>
</table>

- Each request set consists of 10 unique requests.
- Request bandwidth was sufficiently small, so no blocking observed.
Year 2-3: Design Survivability Techniques for use with OSCARS

**Objectives**

- Design and implement survivability techniques on OSCARS using path protection and destination relocation.
- Extend this feature to a multi-domain and multi-layer network.

**Impact**

- Additional resources are reserved for each request.
- Users are protected from link failures (path protection) and destination failures (relocation).

**Strategy**

- Implement basic survivability techniques to protect against link and node failures.
- **Path protection** enables survivable connections by reserving a link-disjoint backup path.
  - OSCARS will provision both primary and link-disjoint backup paths for each connection request.
  - We have extended our P2MP design to implement dedicated path protection.
- The proposed design **slightly** alters existing unicast PCE stack.
  - We have adopted the pre-coordinator approach.
  - Initial implementation completed yesterday!
A user requests a survivable connection. The request is passed to the coordinator, and the primary path is calculated. The status of the request is polled until it returns "ACTIVE", "RESERVED", or "FAILED".

If the primary path is accepted, its explicit path is encoded and added to the optional constraints of the backup request, which is then submitted.

A PCE module will prune from the topology all the links traversed by the primary path, thereby guaranteeing that the backup path will be link-disjoint.
Survivable Request = (SUNN : ATLA)
## Path Protection: Initial Results

### Average VC Hop Count (Inter- + Intra-Nodal)

<table>
<thead>
<tr>
<th>Request Type</th>
<th>Average Hops (Primary/Backup)</th>
<th>Average Provisioning Time (Seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Protected</td>
<td>7.667 / 10.333</td>
<td>47.87</td>
</tr>
<tr>
<td>Unprotected</td>
<td>7.667 / NA</td>
<td>33.28</td>
</tr>
</tbody>
</table>
Year 2-3: Design Survivability Techniques for use with OSCARS

- We plan to extend our anycast PCE design to account for single-link/node failure via anycast path protection and anycast protection using destination relocation.
- **Anycast path protection** allows the ProtectionPCE to pick the anycast destination which has the least-cost link-disjoint path pair from the source.
- **Anycast relocation** allows for the backup path to be routed to an alternate destination in the anycast set. Example: $R(S, \{D_1, D_2\}) \Rightarrow$ Primary: $S \rightarrow D_1$; Backup: $S \rightarrow D_2$.

![Fig A: Anycast Path Protection](image1)

![Fig B: Anycast Protection with Destination Relocation](image2)

We will look into the implementation of dynamic **restoration** mechanisms for all the above survivability techniques.
## Year 2-3: Supporting Quality of Service (QoS) in OSCARS

### Objectives

- To provide for user-profile based access to network resources like domains, nodes, ports, and links while provisioning connection requests.
- To support multi-layer and multi-constraint restrictions on requests based on user privileges.

### Progress

- Designing QoS user profile database extensions based on existing authentication & authorization databases.
- Research and Development on traffic policing and shaping at a request provisioning level driven by user profiles. This can be done using several metrics that balance link bandwidth and VLAN availability using policies like least-used, most-used, most recently used, least recently used, and random.

### Impact

- Classification of user requests based on service requirements.
- Preferential treatment for high-priority requests originating from data/resource-intensive scientific applications.
Year 2-3: What-if Driven Multi-Constrained/Layered OSCARS

Objectives

- Multi-domain offline/inline negotiation protocol for querying and/or reserving the best possible circuit by providing different viable reservation solutions which best suits the user, based on the user profile.

- Rank the various viable connection paths using key performance indicators (KPIs) to best suit the application requirements.

Impact

- Solution matches to user/application requirements as several viable reservation solutions are ranked.
- Re-attempts of reservation for failed reservation requests are minimized.

Progress (in collaboration with ARCHSTONE)

- Implementation in progress of the What-if engine and What-if driven GUI for OSCARS.
- Implementation of a separate offline and inline query + reservation workflow for what-if for multi-domain also in progress.
Research Papers and Journals relevant to COMMON progress

**Year 1: Anycast and Advance Reservation**


**Year 2: Multicast/Manycast Overlay & QoS in OSCARS**


