

This folder belongs to:

# Hands-On Dynamic Circuit Network Workshop

January 31- February 1, 2009

Texas A&M University

College Station, TX



**OSCARS**



**DOE**



**Office of  
Science**



**DRAGON**





# Agenda

- **Day 1**
  - 9am Overview of GMPLS and DRAGON
  - 10am Exercise #1: Designing a GMPLS Control Plane for Ethernet Data Planes
  - 12noon Lunch
  - 1pm Continue working on Exercise #1
  - 2pm Overview of Web Services and OSCARS
  - 3pm Exercise #2: Intra-domain Provisioning with OSCARS
  - 5pm Adjourn
- **Day 2**
  - 9am Overview of Inter-domain implementation in OSCARS
  - 10am Exercise #3: Inter-domain Provisioning with OSCARS
  - 12noon Lunch
  - 1pm Continue working on Exercise #3
  - 3pm Use of DCN and peering dynamic networks
  - 4pm Adjourn

If you are interested in future workshop offerings, please check <http://events.internet2.edu>

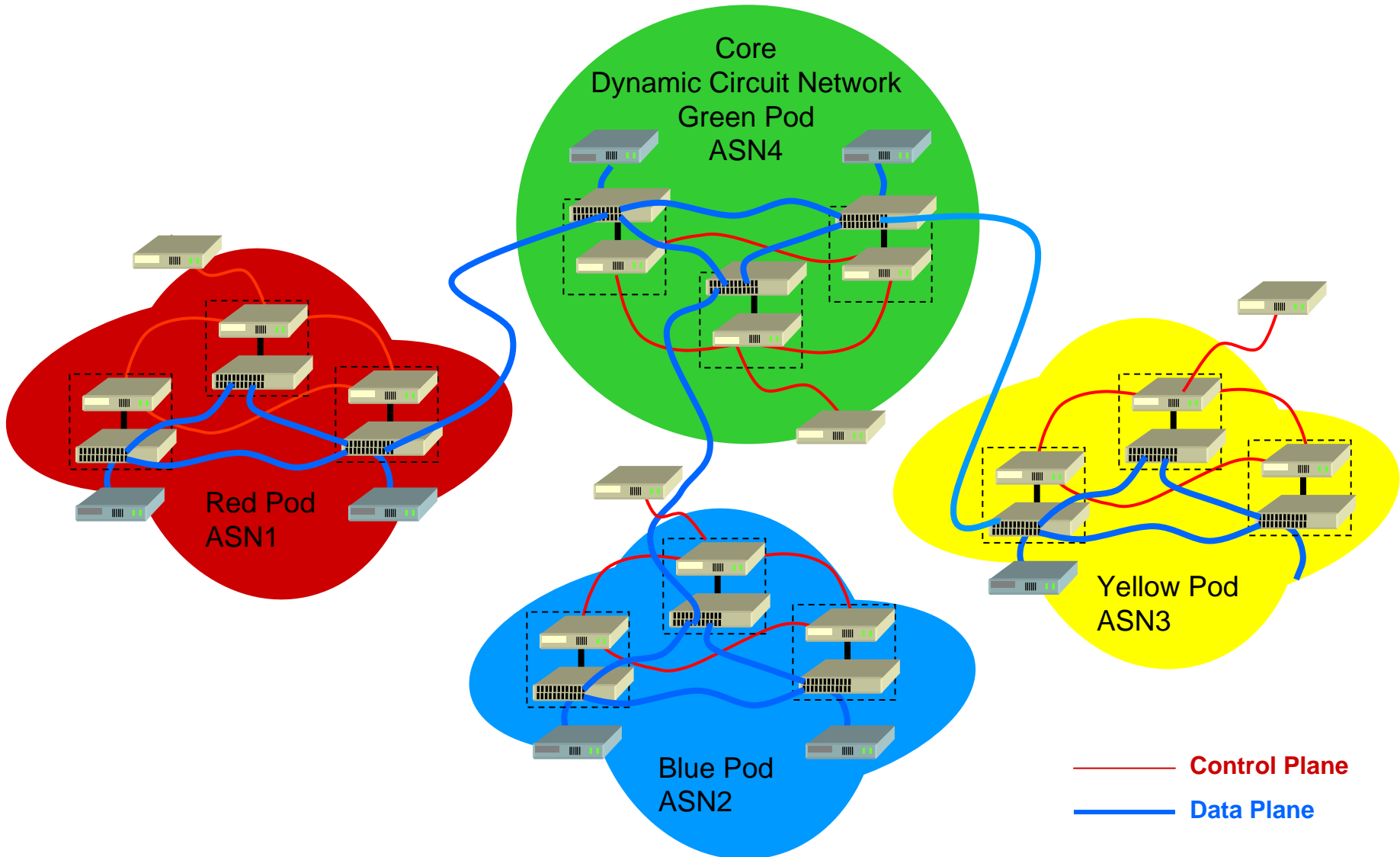


# Login information

- **Wireless Network:**
  - SSID: DCNworkshop
  - WPA Personal Key: Workshop!
- **Login to all VLSRs, NARBs, and End Systems:**
  - ssh port 22
  - username: user[1-16], password: Workshop!
  - username: root, password: rootme
- **Login to all Dell PowerConnect switches**
  - telnet port 23
  - username: admin, password: admin
- **OSCARS configuration; login to the NARB/IDC machine:**
  - ssh port 22
  - username: tomcat55, password: dragon
- **OSCARS axis2 login:**
  - <https://idc.<color>.pod.lan:8443/axis2/axis2-admin/>
  - username: admin, password: axis2
- **OSCARS web user interface:**
  - <https://idc.<color>.pod.lan:8443/OSCARS/>
  - username: oscars-admin, password: oscars



# DCN Workshop Architecture







## Exercise #1: Designing a GMPLS Control Plane for Ethernet Data Planes

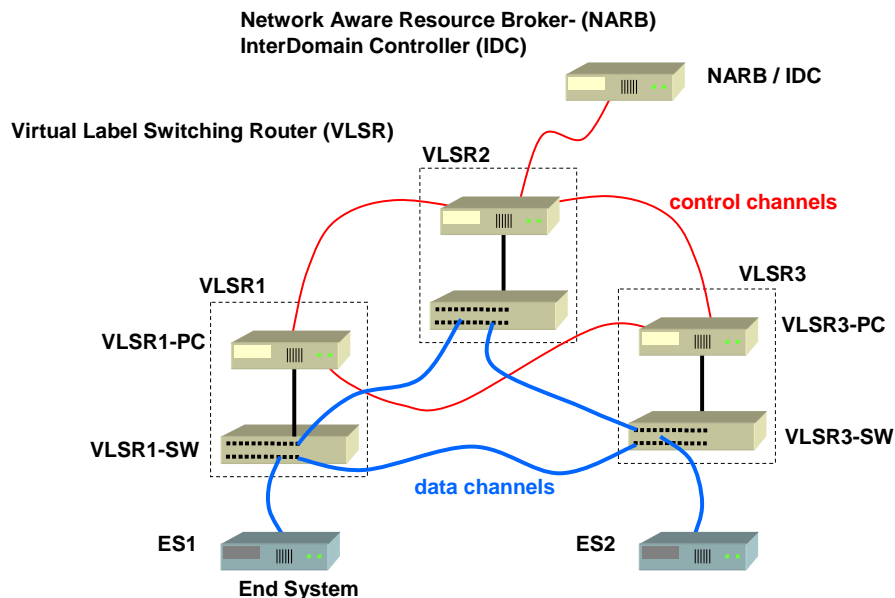
Objective: In this exercise, we will design, build, configure, and test a GMPLS control plane for a dynamic network capable of providing services across an Ethernet data plane.

There will be four engineering teams. Each team will be responsible for one network domain, or “pod”. There are four pod domains: Red, Blue, Yellow, and Green with ASNs of 1, 2, 3, and 4 respectively. The pod architecture is organized to reflect a wide area network with multiple regional networks attached. The Green pod is in the position of the central wide area network in this configuration. The control plane could easily be configured in alternate topologies, such as full or partial mesh configurations. This specific topology is utilized here to facilitate classroom instruction, and also reflects current deployments.

The networks constructed in each exercise will form the basis for the subsequent exercises. Each network pod consists of three Virtual Label Switching Routers (VLSRs), two End Systems (ES), and one machine hosting both the Network Aware Resource Broker (NARB) and Inter-Domain Controller (IDC) software.

### Pod Architecture Overview

We will be building a network consisting of three Ethernet switches acting as network elements (see diagram below). Two PCs will act as End Systems. The team will develop a network diagram incorporating route diversity, where the end systems attach to the network, and all the interface and/or port assignments required to support the network.



Each device in the network will require a “management” IP address. For End Systems, this would be a normal IP address for that host. For the network elements, management addresses are used primarily to access the device for configuration. All management interfaces should have publicly routable IPv4 addresses. In this exercise, we will use the management address where a loopback address might be traditionally used for router IDs. In general, we use these management addresses wherever a publicly reachable address for the device would normally be required.

In addition to these management interfaces, each device should have a separate “data plane” interface over which the dynamic circuits will be delivered. The End Systems provided in the workshop lab all have dual gigE ports, eth0 and eth1. In this exercise eth0 will be used for management and control, and eth1 for data plane.

(Note: Each PC has dual 1gigE integrated ethernet ports: eth0 and eth1. We use one physical interface for management and control and a separate interface for data plane. We have arbitrarily designated eth0 for management and control, and eth1 as the data plane interface. The standards do allow for control, management, and data to be provisioned over the same physical interface, but the DRAGON implementation does not yet support this feature.)

In a real network, network elements have a separate “management” address and “loopback” address assigned to each device. The management addresses are used by the network operations to access the devices for configuration and monitoring and are often deliberately not visible to general users. The loopback addresses are used to associate a publicly reachable address with each device that will always be “up”. The GMPLS control channels are provisioned over GRE tunnels between network elements. In these exercises the GRE tunnels use the management addresses as the tunnel endpoints. We have not assigned separate loopback addresses in this workshop in order to simplify the overall addressing. Instead, we use same address for the loopback and management address.

For these exercises we use the 192.168.0.0/16 subnet for management addressing. Appendix A (Management Plane Configuration) shows the details of the management plane architecture and configuration. The management plane is already configured and in place on the workshop pods, so there are no class tasks required for this.

There are three main areas which must be addressed when building a dynamic network:

Data Plane – Exercise 1, Step 1

Control Plane Infrastructure (control communication channels) – Exercise 1, Step 2

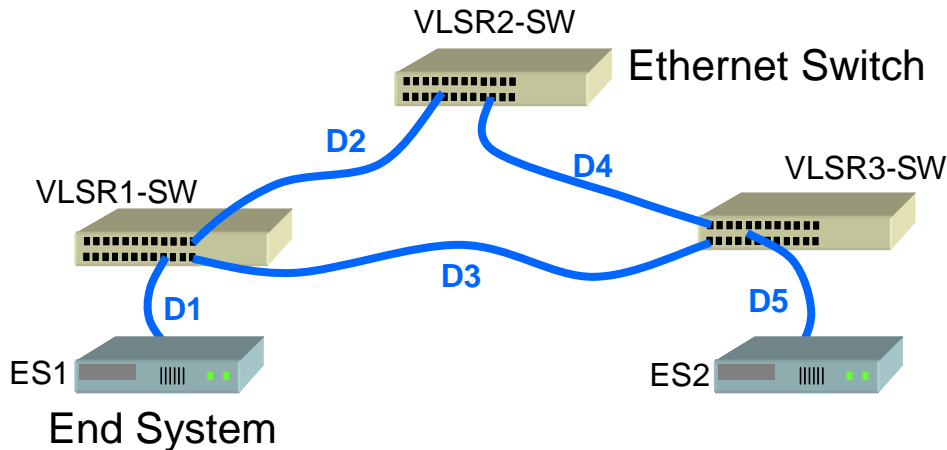
Control Plane Software (DRAGON) configuration – Exercise 1, Steps 3-6

Exercise 1 includes six steps as indicated above which will guide us through the process of building and configuring each of these. Upon completion of Exercise 1, we will have configured and tested the dynamic provision of Ethernet VLAN based network services using the DRAGON GMPLS based control plane.

Appendix B (Detailed Pod Architecture and Configuration) contains the details of the eventual architecture and configuration for the workshop pods. Reference will be made to the information in this Appendix as part of many of the steps throughout this workshop.

## Step 1: Data Plane Layout and Build

In this step we will create the physical data plane connections between the network elements. The generic data plane configuration for a pod is as shown below.



### Data Plane via Cat5 Patch Cable

Each group will need to visit the pod rack and connect the specific data plane for their individual pods. It is recommended that you reference the information in Appendix B, and fill out the table below to facilitate cable connections on the pod racks. Each team may also want to fill out the worksheet in Appendix A for Exercise 1 steps 1-3. For the workshop configuration create a list of data plane links. Number them  $D_1$ , to  $D_n$ . Identify the device and port/interface for each end of the data link.

ID	Network Element	Interface/Port	Network Element	Interface/Port
D2 (example)	VLSR1-SW	port 4	VLSR2-SW	port 3
D1				
D2				
D3				
D4				
D5				

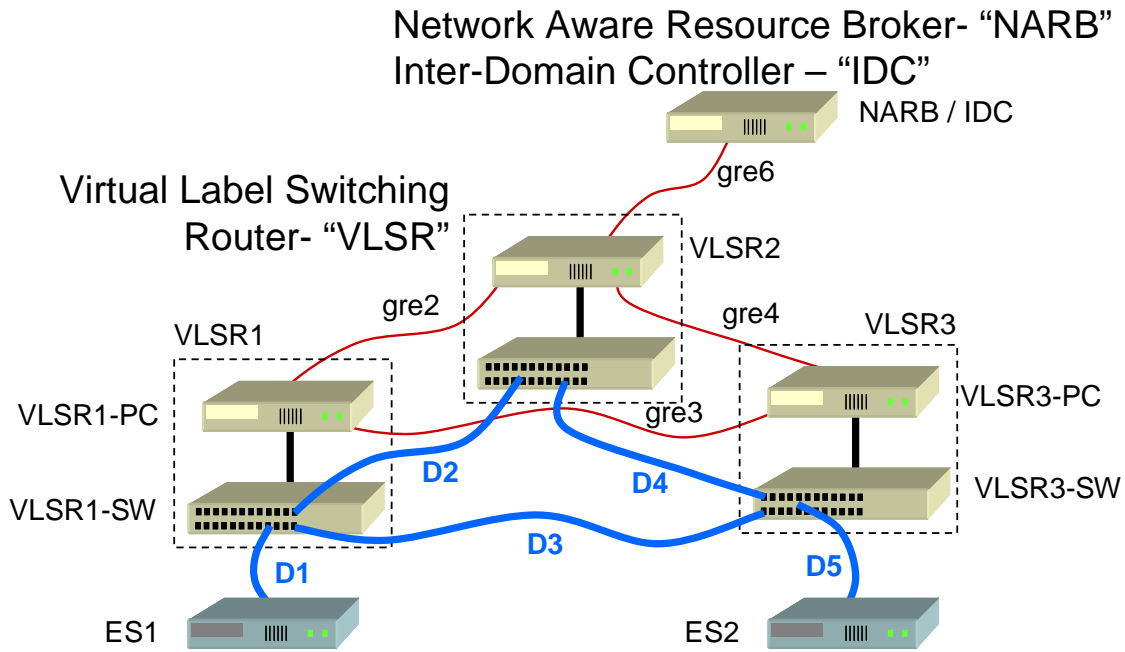
→Attendees should now go to the lab and install the appropriate Ethernet cabling for the desired data plane connectivity←

**Step 2: Design and Build the Control Plane Infrastructure (control communication channels) to support the GMPLS operations.**

Since each of the Ethernet switches will act as a dynamically provisioned switching element, each switch will need a control PC associated with it to run the OSPF-TE and RSVP-TE protocols. These switch-PC pairs are referred to as VLSRs. The control PC is responsible for running the routing and signaling protocols and reconfiguring the associated Ethernet switch as necessary.

Every networked device participating in the dynamically switched network will require a “control link” to be established with neighboring devices. In GMPLS, these control plane devices may not be physically adjacent to each other. In order to allow for intervening network infrastructure, we establish GRE tunnels between the neighboring control plane speakers, and assign IP addresses to these tunnel interfaces for the control links.

With respect to the VLSRs, all control plane links are established with the VLSR-PC (not the VLSR-SW). The team should layout the control plane links that will cover the data plane designed in step 1. The generic control plane channel configuration for a pod is as shown below. The red lines indicate the control channels along with the control plane, layered on top of the data plane from the previous exercise.



Each control plane link will require a GRE tunnel be established between logically adjacent network elements (VLSRs and/or NARB/IDC). Since ESs are not running routing and signaling protocols, we do not need to setup GRE tunnels between VLSRs and ESs. We use the management addresses as the endpoints for building all GRE tunnels.

The 10.0.0.0/8 subnet is utilized for control plane addressing. In a real network, the intra-domain control plane links do not strictly need to be public. All inter-domain communications are accomplished via Web Service based communication between IDCs. Therefore, the only address which strictly needs to be routed and reachable is the IDC Web Service entry point.

Each control link will require a /30 subnet be assigned from the appropriate CIDR blocks:

Red.....10.1.0.0/16  
Blue.....10.2.0.0/16  
Yellow...10.3.0.0/16  
Green....10.4.0.0/16

Tip: In order to keep things manageable, number all the control links uniquely within the domain, and congruently with the data plane links. Then use this numbering scheme to identify which GRE interface to use for each control link. The intra-domain GRE names will include the color of the pod as follows:

Red: red-gre#  
Blue: blue-gre#  
Yellow: ylw-gre#  
Green: grn-gre#

The GRE number can also be used as part of the control plane subnet as follows:

10.<asn>.<gre#>.h

Red: red-gre2: 10.1.2.0/30  
Blue: blue-gre5: 10.2.5.0/30  
Green: grn-gre6: 10.4.6.0/30

(Note: The GRE interface names defined at each end of the control link do **not** need to be the same. However, doing so will reduce confusion and help in debugging configuration problems.)

In this step attendees should make a list of control channels to be configured in their respective pod. This information will be utilized as part of the configurations in step 4. Each control link should include the associated data link, the GRE tunnel # to be used, the IP addresses of the tunnel endpoints, and the control plane IP addresses to be assigned to the GRE interface. You may wish to build this list for each device in the network (rather than by link) in order to facilitate the set up process later. Appendix B provides detailed information which can be used to build this list.

GRE Name	Local Network Element	Local Ctrl Addr	Local Tunnel Addr	Remote Network Element	Remote Ctrl Addr	Remote Tunnel Addr
red-gre2 (example)	vlsr1-pc	10.1.2.1	192.168.1.4	vlsr2-pc	10.1.2.2	192.168.1.6

→Attendees should now create a list of control plane links for the control channels in their pod←

### Step 3: Assign Traffic Engineering (TE) Addresses

The TE-addresses are “identifiers” for the data links in the dynamic network. While they are formatted like regular IP addresses, they are not IP addresses in the context of being able to use them as addresses for sending or receiving of data. TE-addresses are strictly utilized by the control plane to identify, compute paths, and signal network provisioning across the dynamic network. TE-Addresses provide a means to associate specific data plane interfaces with their control links. Since the TE addresses are not used for IP routing across the interfaces, in order to send data over a dynamically allocated circuit, the user will need to allocate and assign a data plane IP address to the interface associated with the newly created circuit. The TE address should not be used for this purpose.

We use the 11.0.0.0/8 net for this purpose in order to provide a numbering scheme congruent with the control plane addressing.

Traffic Engineering (TE) addresses should be assigned for each data plane interface. For this exercise, and to keep the addressing scheme somewhat intuitive, we use the 11.0.0.0/8 net in a similar fashion to the 10.0.0.0/8 net assignments used for control links:

Red data plane link D2: 11.1.2.0/30  
Yellow data plane link D4: 11.3.4.0/30

Since the inter-domain solution is based on Web Services, it makes no difference if TE addresses are based on private address space compared to globally unique addresses. The GMPLS standards do allow for configurations where inter-domain data is shared at a GMPLS level. In this configuration, use of private address space would be a bad practice. However, for the solution and configuration presented in this class, private address space for TE links numbering is fine.

In this step attendees should make a list of data plane links and associated TE addresses for the data plane links to be configured in their respective pod. This information will be utilized as part of the configurations in step 4. Each data plane link should include the data link id, the local/remote network element, local/remote TE addresses, and the associated control channel id. Appendix B provides detailed information which can be used to build this list. Please note that end systems attach at special locations to the dynamic network known as “edge” ports. These connections do not have TE addresses or associate control channels. Details on how to handle these data links and end systems will be included in the subsequent control plane software configuration steps. For now, all that is needed is to note the associated network elements.

Data Plane Link ID	Local Network Device	Local TE Addr	Remote TE Addr	Remote Network Device	Associated Control Channel ID
D2-example	vlsr1-sw	11.1.2.1	11.1.2.2	vlsr2-sw	red-gre2
D1	<b>edge port – no TE link</b>				
D2					
D3					
D4					
D5	<b>edge port – no TE link</b>				

→Attendees should now create a list of TE link addresses for the data plane links in their pod←

#### Step 4: Configure the VLSRs

Now we use the information gathered in the previous steps to configure the DRAGON control plane software. This will allow us to dynamically provision vlan based Ethernet “circuits” across the pods. For this exercise we will limit ourselves to “intra-domain” or “intra-pod” provisioning operations. Appendix B provides detailed information which will be helpful to understanding the configuration items to follow.

All pod PCs have been pre-loaded with Debian Linux, the software dependency packages required for the DRAGON software and/or systems support, and a recent release of the DRAGON GMPLS software suite. Detailed instructions for installation of the DRAGON software is provided as part of the DCN Software Suite release. Instructions for obtaining and installing this software are provided as part of the slides presented and in the course material.

You can login to the machines in your pod via SSH using the standard port, 22. The username is **root**, password **rootme**. Or, you can optionally login with username **user1...user16** with the password **Workshop!** and run `sudo` for root access.

We will configure the three VLSRs to reflect this network diagram. To configure a VLSR, there are several steps:

#### A. Configure the GRE Tunnels (Control Plane Communication Channels)

Here we need to configure the GRE tunnels on each of the VLSRs in the pod. As described earlier, these are the control plane communication channels which the VLSRs used for routing and signaling communications. On each VLSR-PC, we need to create the appropriate GRE tunnels and then assign control plane addresses to those GRE interfaces. We have modified the Debian startup process to invoke the `/etc/rc.local` script.

For this step, all GRE tunnels in the pod are already up and running. However, it will be a good idea to get familiar with tunnel set up by following the steps as shown below:

Check `/etc/rc.local`. You should find the following `modprobe` commands at the top of the file, as well as a call to the `setup_gre_tunnels.sh` script in `/usr/local/dragon/etc`:

```
# needed for GRE tunnel support, e.g. 'ip tunnel'
/sbin/modprobe ip_gre
# needed for tagged VLAN support, e.g. 'vconfig'
/sbin/modprobe 8021q
# run the script that set up the gre_tunnels
/usr/local/dragon/etc/setup_gre_tunnels.sh
```

Note: In the first exercise, circuit setup will be in port-to-port/untagged mode, so the support for tagged VLAN (the '8021q' Linux module) is not necessary, but we will add the support here for exercises that come later. Use the following command to view the current GRE tunnel configuration on each of your hosts:

```
red_vlsr1_pc:~# ip addr show
...
5: red-gre2@NONE: <POINTOPOINT,NOARP,UP> mtu 1476 qdisc noqueue
   link/gre 192.168.1.4 peer 192.168.1.6
   inet 10.1.2.1/30 scope global red-gre2
6: red-gre3@NONE: <POINTOPOINT,NOARP,UP> mtu 1476 qdisc noqueue
   link/gre 192.168.1.4 peer 192.168.1.8
   inet 10.1.3.1/30 scope global red-gre3
```



The script `/usr/local/dragon/etc/setup_gre_tunnels.sh` on each machine is used to configure the GRE tunnels, and is included in the startup process as shown above.

```
# GRE between red-vlsr1 and red-vlsr2
ip tunnel del red-gre2
ip tunnel add red-gre2 mode gre remote 192.168.1.6 local 192.168.1.4 ttl 255
ip link set red-gre2 up
ip addr add 10.1.2.1/30 dev red-gre2
ip route add 10.1.2.2/32 dev red-gre2
```

## B. Edit VLSR Software Configuration Files

There are four configuration files on each VLSR that must be edited in this step:

- o `dragon.conf`: this is the configuration file for the dragon daemon which provides the user interface and provisioning entry point into the DRAGON control plane.
- o `ospfd.conf`: this is the configuration file for the OSPF daemon, which is the GMPLS OSPF-TE routing engine.
- o `RSVPD.conf`: this is the configuration file for the RSVP daemon, which is the GMPLS RSVP-TE signaling engine.
- o `zebra.conf`: this is the configuration file for the zebra daemon, which provides the interface the system kernel for routing related configurations and changes.

In each case a sample file has been provided which will be copied to the default configuration file name and then edited by the attendees. Please note that these configuration files will need to be edited on **all three VLSR** machines of your pod.

In the directory `/usr/local/dragon/etc`, copy the sample conf files to working files. Below is an example on `red_vlsr1_pc`:

```
red_vlsr1_pc:~# cd /usr/local/dragon/etc
red_vlsr1_pc:/usr/local/dragon/etc# cp RSVPD.conf.sample RSVPD.conf
red_vlsr1_pc:/usr/local/dragon/etc# cp zebra.conf.sample zebra.conf
red_vlsr1_pc:/usr/local/dragon/etc# cp ospfd.conf.sample ospfd.conf
red_vlsr1_pc:/usr/local/dragon/etc# cp dragon.conf.sample dragon.conf
```

Edit the `dragon.conf` file. For now, this file only contains the hostname assignment for this VLSR and the login password. For simplicity sake, we set the password to “dragon”.

```
hostname red-vlsr1
password dragon
```

Edit the `zebra.conf` file. For now, this file only contains the hostname assignment for this VLSR and the login password. For simplicity sake, we set the password to “dragon”.

```
hostname red-vlsr1
password dragon
enable password dragon
```

Next, edit the `rsvpd.conf` file. This configuration file tells the signaling daemon which control channels (GRE tunnels) to use for signaling. In this file, all we need to do is to define each GRE interface RSVP will be using for signaling. For example, green vlsr1-pc will have two GRE interfaces: grn-gre2 (to VLSR2) and grn-gre3 (to VLSR3). Edit `rsvpd.conf` to look like the following:

```
interface grn-gre2 tc none mpls
interface grn-gre3 tc none mpls
api 4000
```

Finally, edit the `ospfd.conf` file. This is the configuration file for the GMPLS OSPF-TE routing engine. We use this file to tell the OSPF-TE routing engine the following information:

- identify which control channels to use for establishment of OSPF-TE adjacencies (i.e., location of peering VLSRs)
- describe the data plane (TE link) characteristics and capabilities
- define the association between control channels and data plane links
- define how to contact the “data plane switch fabric” which in this case is an Ethernet switch associated with each VLSR PC

It is here where we define the data plane topology over which provisioning actions can occur. In addition, this configuration file will tell OSPF which control links are associated with which data links. The resulting information exchanged by the individual OSPF-TE speakers allows for the construction of a Traffic Engineering Topology Database (TEDB) which contains a topological view of the dynamic data plane infrastructure. This provides the basis for subsequent path computation and signaling operations which will be described in subsequent sections.

An example `ospfd.conf` file is shown below. There are comments in the form of “#explanation comments here#” which are not included in the sample files and are not needed in the files each pod team will be creating. They are included here to provide some additional information regarding the configuration statements. The following text is excerpted from one VLSR. Each pod team should review this configuration example and make appropriate edits for the `ospfd.conf` files on each of their VLSRs:

```
! OSPFd sample configuration file
#here we define the hostname, passwords, and location for log files#
hostname yellow-vlsr1-ospf
password dragon
enable password dragon
log stdout
log file /var/log/ospfd.log
!
!
#here we define the control channels over which OSPF-TE adjacencies will be established.#
interface ylw-gre2
description GRE tunnel between yellow-vlsr1 and yellow-vlsr2
ip ospf network point-to-point
!
interface ylw-gre3
description GRE tunnel between yellow-vlsr1 and yellow-vlsr3
ip ospf network point-to-point
!
```

```

#here we define the router-id and the networks which OSPF-TE will include in its
advertisements. Please notice that the networks below are the GRE tunnel networks.#
router ospf
  ospf router-id 192.168.3.4
  network 10.3.2.0/30 area 0.0.0.0
  network 10.3.3.0/30 area 0.0.0.0
#here we define the router-address for the OSPF-TE process, this is the generally the same
as for the router-id above.
  ospf-te router-address 192.168.3.4
#the ospf-te statements define the data plane topology associated with this VLSR. In the
example below, the "11.3.2.1" is the TE address identified in an earlier exercise. This
is used by routing, path computation, and signaling to identify this particular link. The
"snmp switch-ip 192.168.3.3" indicates that the associate Ethernet switch is located at
that address and can be controlled via SNMP commands. The "switch-port 4" indicates that
the specific data plane link (TE link 11.3.2.1) terminates on port 4 of the associated
Ethernet switch#
#the "swcap l2sc encoding ethernet" commands indicates this is an Ethernet TE link in the
data plane. GMPLS allows for other types of links like packet, sonet, lambda, fiber, etc.
These types of data plane links are not covered in this workshop.#
#the "max-bw" indicates the maximum bandwidth of the data plane link being described.
This typically matches the physical capabilities of the link. The values below are in
bytes, so number below indicates a 1 Gbps link.#
#the "max-rsv-bw" indicates how much of the total bandwidth is available for dynamic
services. For instance if someone had a 10 Gbps link and only wanted dynamic services to
take up to 5 Gbps, then, that number could be reflected here.#
#"max-lsp-bw" indicates the maximum granularity for a single lsp provision. This is
typically set to the same value as max-rsv-bw, so a single lsp could take all the
bandwidth available for that data plane link.#
#the max-lsp-bw #" commands are intended to allow application of policies based on
different administrative groups. These are not used in this manner at this time.#
#the "vlan # to #" statement indicates which vlans are available for dynamic services on
this data plane link."
#the "metric" statement is a standard value to allow one link to have more (or less)
preference over others#
  ospf-te interface ylw-gre2
    level gmpls
    data-interface ip 11.3.2.1 protocol snmp switch-ip 192.168.3.3 switch-port 4
    swcap l2sc encoding ethernet
    max-bw 125000000
    max-rsv-bw 125000000
    max-lsp-bw 0 125000000
    max-lsp-bw 1 125000000
    max-lsp-bw 2 125000000
    max-lsp-bw 3 125000000
    max-lsp-bw 4 125000000
    max-lsp-bw 5 125000000
    max-lsp-bw 6 125000000
    max-lsp-bw 7 125000000
    vlan 100 to 200
    metric 10
  exit
#the above exit command indicates that the description of the data plane link (TE Link
11.3.2.1) associated with the control plane channel (ylw-gre2) has been fully described.
The following statements contain similar information for the other data plane links in the
topology for this VLSR. The example below is for ylw-gre3 without the explanation
comments from above.#
  ospf-te interface ylw-gre3
    level gmpls
    data-interface ip 11.3.3.1 protocol snmp switch-ip 192.168.3.3 switch-port 5
    swcap l2sc encoding ethernet
    max-bw 125000000
    max-rsv-bw 125000000
    max-lsp-bw 0 125000000

```

```
max-lsp-bw 1 125000000
max-lsp-bw 2 125000000
max-lsp-bw 3 125000000
max-lsp-bw 4 125000000
max-lsp-bw 5 125000000
max-lsp-bw 6 125000000
max-lsp-bw 7 125000000
vlan 100 to 200
metric 10
exit
```

### C. Check the configuration of the covered Ethernet switches

After SSH'ing to a VLSR PC, execute “telnet <switch\_ip>” and login with the username **admin** and password **admin**.

The switch IP addresses are 192.168.x.3, 192.168.x.5, and 192.168.x.7 for the switches covered by VLSR1, VLSR2 and VLSR3, respectively.

**Note:** We have already pre-loaded the switches with the correct configuration. The CLI commands below were used to initialize the Dell PowerConnect 5324 switches for their role in these workshop networks – other switches will differ slightly in their initial configuration.

The switch is set up to match the features described in the GMPLS \*.conf files. Every switch is slightly different in how they handle VLANs, spanning tree, MTUs, and in whether or not they support SNMP. For the workshop, we have provided Dell PowerConnect 5324 Ethernet switches. We need to setup the proper management IP address, set the SNMP read/write community string, and initialize ports that will be under VLSR control. We want to disable spanning tree protocol and filter BPDU packets on all the VLSR ports in order to prevent STP flooding. For these Dell switches in the workshop configuration, we want ports 1 and 2 to be in VLAN #1 – this VLAN will serve as a management interface. Ports g3 through g24 will be available for data plane switching.

```
red_vlsr1_sw# configure
red_vlsr1_sw(config)# no spanning-tree
red_vlsr1_sw(config)# spanning-tree bpdu filtering
red_vlsr1_sw(config)# port jumbo-frame
red_vlsr1_sw(config)# interface range ethernet g(3-24)
red_vlsr1_sw(config-if)# switchport mode general
red_vlsr1_sw(config-if)# spanning-tree disable
red_vlsr1_sw(config-if)# switchport general pvid 2
red_vlsr1_sw(config-if)# exit
red_vlsr1_sw(config)# interface range ethernet g(1-2)
red_vlsr1_sw(config-if)# switchport mode general
red_vlsr1_sw(config-if)# switchport general pvid 1
red_vlsr1_sw(config-if)# exit
red_vlsr1_sw(config)# logging console debugging
red_vlsr1_sw(config)# enable password level 15 admin
red_vlsr1_sw(config)# username admin password admin level 15
red_vlsr1_sw(config)# snmp-server community dragon rw
red_vlsr1_sw(config)# exit
red_vlsr1_sw#
```

The teams should login to the VLSR switches to peruse the configuration and to display the port-to-VLAN mappings.

To show the port-to-VLAN assignments, execute the following command on the switch:

```
red_vlsr1_sw# show vlan
```

Vlan	Name	Ports	Type	Authorization
1	1	g(1-2),ch(1-8)	other	Required

To show the entire switch configuration, run this command:

```
red_vlsr1_sw# show running-config
```

At first glance, you may realize that the running-config is different from the default configuration that is shown above. Unfortunately, this is just the way Dell PowerConnect switches arrange the output of the `show running-config` command.

### Step 5: Configure the NARB

The NARB provides intra-domain service routing functions. NARB is the DRAGON path computation element (PCE) used to create an Explicit Route Object (ERO) for the RSVP PATH message. This ERO specifies the path that the Label Switched Path (LSP) is to take with a domain. Topology information for the local domain may be provided in the `narb.conf` file, but it is not required for this workshop. This file must be edited to describe the linkage to the local OSPF module.

In this exercise, we will implement the NARB in order to provide advanced path computation capabilities on an intra-domain basis. While the intra-domain case will not take full advantage of the NARB’s capabilities, its presence will satisfy requirements to provide EROs for certain types of provisioning requests...most notably the “local ID” mode that we will implement in the next step.

Procedure:

Setting up the NARB actually consists of configuring two functional entities: 1) an intra-domain OSPF listener and 2) the NARB itself. Traditionally, the NARB and all of its component daemons are run on their own host. While not strictly necessary, the path computation algorithms and link state processing in a large network could consume significant computational resources – perhaps more than a typical network element might have available.

In this exercise, we will run the NARB processes on a separate host. We must therefore allocate and set up appropriate GRE tunnels. One GRE tunnel must be established from the NARB server to one of the intra-domain VLSRs. Any VLSR will work and we have picked VLSR2 to connect to the NARB in each pod – this link will be configured in `ospfd` to flood LSAs to the NARB from inside the domain.

First, login to the NARB PC at 192.168.x.10 via SSH. Follow the steps outlined below to complete the configuration of the NARB component.

- 1) In the directory `/usr/local/dragon/etc`, copy these sample conf files to working files. Below is an example on red-narb:

```
red-narb:~# cd /usr/local/dragon/etc
red-narb:/usr/local/dragon/etc# cp narb.conf.sample narb.conf
red-narb:/usr/local/dragon/etc# cp rce.conf.sample rce.conf
red-narb:/usr/local/dragon/etc# cp schema_combo.rsd.sample schema_combo.rsd
red-narb:/usr/local/dragon/etc# cp ospfd-intra.conf.sample ospfd-intra.conf
red-narb:/usr/local/dragon/etc# cp zebra.conf.sample zebra.conf
```

We will not be using the NARB in inter-domain mode, so just create an empty `ospfd-inter.conf` file:

```
red-narb:/usr/local/dragon/etc# touch ospfd-inter.conf
```

- 2) Edit the `/usr/local/dragon/etc/ospfd-intra.conf` file

In this file, we need to describe a passive adjacency between the NARB's intra-domain ospfd instance and any one of the VLSR ospfd instances within the local domain. The following statements make up the red domain `ospf-intra.conf` file:

```
hostname red-narb
password dragon
enable password dragon
log stdout
log file /var/log/ospfd.log
interface red-gre6
  description GRE tunnel red-gre6
  ip ospf network point-to-point
router ospf
  ospf router-id 192.168.1.10
  network 10.1.6.0/30 area 0.0.0.0
  ospf-te router-address 192.168.1.10
```

**Note: There is no TE LINK between the NARB and VLSR.**

- 3) Edit the `narb.conf` file.

We must define the local domain ID – we use the management CIDR block, but any unsigned integer could be used (e.g. 1 for red, 3 for yellow, etc). We also need to specify the linkage to the intra-domain OSPF instances.

Below is an example for the red pod's `narb.conf` file:

```
domain-id { ip 192.168.1.0 }
cli { host red-narb password dragon }
intra-domain-ospfd { address localhost port 2617 originate-interface 10.1.6.2 area 0.0.0.0 }
```

- 4) Before starting the NARB, login to the VLSR with the adjacency to the NARB, i.e. VLSR2 in our case. Edit the `ospfd.conf` file so that the ospfd on vlsr2 will use `yourColor-gre6` to form an

adjacency to the intra-domain ospfd process running on the NARB. Again, remember, there is no TE-link between the NARB and the VLSR2. Do **not** add any `ospf-te` interface or `level gmp1s` configuration commands on VLSR2. You only need to add these sections to your `ospfd.conf` file on VLSR2:

```
interface yourColor-gre6
  description GRE tunnel yourColor-gre6
  ip ospf network point-to-point
router ospf
  network 10.X.6.0/30 area 0.0.0.0
```

5) Also, login to all VLSRs in your pod. Now that the NARB is configured as part of the local control plane, we need to tell the dragon daemons how to contact the NARB.

Add the following lines to `/usr/local/dragon/etc/dragon.conf` on each of your VLSRs:

```
configure narb intra-domain ip-address 192.168.X.10 port 2609
set narb-extra-options query-with-confirmation
set narb-extra-options query-with-holding
```

6) Start the DRAGON software on the VLSR PCs

The final step is to start all of the DRAGON protocol agents. Use the following shell commands on the VLSR PCs to start the VLSR daemons, and then use `ps | grep` to see the DRAGON processes:

```
red_vlsr1_pc:~# /usr/local/dragon/bin/dragon.sh start-vlsr
red_vlsr1_pc:~# ps auxwww | grep dragon
```

The `dragon.sh` command starts the `zebra`, `ospf`, `RSVP`, and `dragon` daemons and places them in the background. Note: `dragon.sh` is used to start, restart, stop, status all of the protocol daemons. Invoke `dragon.sh` without any command line arguments to get a list of options.

After starting the daemons, we want to verify that the protocols are talking properly. For example, when `ospfd` starts up, it will try to issue HELLO exchange on the configured GRE links. Ideally, adjacencies are formed and link state announcements (LSAs) are flooded across to neighbors to build the link state database, i.e. the intra-domain topology of the network. You can query the status of these OSPF adjacencies by logging in to the `ospfd` CLI port (2604): (By default, the password for logging into the DRAGON CLI is “dragon” unless you have overridden this in the configuration files.)

```
red_vlsr1_pc:~# telnet localhost 2604
password> *****
red_vlsr1-ospf> sh ip ospf interface
red_vlsr1-ospf> sh ip ospf-te database detail
red_vlsr1-ospf> sh ip ospf neighbor
red_vlsr1-ospf> exit
```

By default, the protocol daemons will write to log files in `/var/log`. You can inspect these files to get a more detailed understanding of the state of the protocols or interfaces.

Go back to the shell, and run

```
red_vlsr1-pc:~# less /var/log/ospfd.log
red_vlsr1-pc:~# less /var/log/RSVPD.log
```

7) Make sure the VLSRs in the local domain are all up and active. Then start the NARB with:

```
red-narb:~# /usr/local/dragon/bin/dragon.sh start-narb
...
#####
DRAGON NARB Started...
#####
NARB_OUTPUT @[2008/01/24 18:44:56] : Running without connection to OSPFd.....
NARB_OUTPUT @[2008/01/24 18:44:56] : No abstract topology generated.....
dragon-sw: started narb daemons.
red-narb:~#
```

You will see a message saying that the NARB is running without connection to OSPFd. The OSPFd that is being mentioned is the inter-domain OSPFd. Since we are not using the NARB for inter-domain provisioning, we can ignore that message.

The CLI ports associated with NARB elements are as follows:

- 2604 ospfd-intra-domain CLI
- 2626 NARB CLI
- 2688 RCE CLI

You can telnet to these processes and inspect the OSPF adjacencies and topology information.

For example, to check intra-domain topology via the RCE CLI on your NARB PC:

```
red-narb:~# telnet localhost 2688
rce:cli>show topology intradomain
.....Router ID Opaque LSA.....
Adv_router (192.168.1.4), Router_id (192.168.1.4)
Adv_router (192.168.1.6), Router_id (192.168.1.6)
Adv_router (192.168.1.8), Router_id (192.168.1.8)
.....TE Link Opaque LSA.....
Adv_router (192.168.1.4), Link_id (192.168.1.6), IfAddrs[11.1.2.1-11.1.2.2]
Adv_router (192.168.1.4), Link_id (192.168.1.8), IfAddrs[11.1.3.1-11.1.3.2]
Adv_router (192.168.1.6), Link_id (192.168.1.4), IfAddrs[11.1.2.2-11.1.2.1]
Adv_router (192.168.1.6), Link_id (192.168.1.8), IfAddrs[11.1.4.1-11.1.4.2]
Adv_router (192.168.1.8), Link_id (192.168.1.4), IfAddrs[11.1.3.2-11.1.3.1]
Adv_router (192.168.1.8), Link_id (192.168.1.6), IfAddrs[11.1.4.2-11.1.4.1]
.....The End.....
rce:cli>
```

## Step 6: Configure the Local-ID

The “Local ID” is a non-standard construct developed by DRAGON to facilitate circuit creation from one VLSR edge port to another. This feature can be used to allow a VLSR to proxy for an otherwise RSVP unaware device – e.g. a video camera, or computational cluster, etc. The DRAGON software



requires root access to run the RSVP and OSPF protocols because of the need to access raw sockets, so it is convenient to be able to create a circuit without directly involving the end systems in the control plane. In the R&E networks, it is expected that Ethernet LSPs will often be used to link clusters in one location to clusters in another location – mapping a group of ports on one switch with a group of ports on some other switch. Since RSVP does not support VLANs as a termination point, the DRAGON software provides for the creation of a “local ID” on a local switch which can then be used as a sub-object for terminating a PATH request at the local switch. Various types of "local ID" configurations can be created. The options include a "local id" is which is associated with a single port (tagged or untagged), a group of untagged ports, or a group of tagged ports. These can be created via the CLI or specified in the dragon.conf file. This local ID can then be specified in the LSP configuration at the source and destination.

In order to use local ID, there must be an active NARB available and configured in the dragon.conf files at the source and destination VLSRs. As a general rule, pointers to the NARB should be configured on all VLSRs throughout the network.

While this is a non-standard feature incorporated into the DRAGON software, it provides a capability to link clustered systems on one switch to another set of clustered systems on another switch. The “local ID” is a terminating entity within a VLSR, allowing the signaling protocol to terminate the LSP to an internally defined VLAN port group.

Procedure:

1) A useful tool: `/usr/local/dragon/sbin/narb_test`

`narb_test` is a tool to query the NARB for an ERO. We use this tool often to test if a host is reachable to another host.

For example, from `red-vlsr1`, we can query `red-narb` (192.168.1.10) to see if `red-vlsr1` can reach `red-vlsr3` as shown below, which VLAN tags are available, and the first available end-to-end (E2E) VLAN tag:

```
red_vlsr1_pc:~# narb_test -H 192.168.1.10 -S 192.168.1.4 -D 192.168.1.8 -V -a
NARB@[2008/01/23 02:16:47] : Request successful! ERO returned...
NARB@[2008/01/23 02:16:47] : HOP-TYPE [strict]: 11.1.3.1 [UnnumIfId: 262244(4,100)]
NARB@[2008/01/23 02:16:47] : HOP-TYPE [strict]: 11.1.3.2 [UnnumIfId: 262244(4,100)]
NARB@[2008/01/23 02:16:47] : E2E VLAN TAG [ 100 ]
NARB@[2008/01/23 02:16:47] : ALL E2E VLAN TAGS: 100 101 102 103 104 105 106 107
108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147
148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167
168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187
188 189 190 191 192 193 194 195 196 197 198 199 200
```

2) The local ID constructs must be defined before they can be used to establish an LSP. You can configure the local-id via the CLI, or you can define them in the \*.conf files.

Edit the `dragon.conf` files on **VLSR1** and **VLSR3** to define several local-ids to enable port, group, and tagged-group LSPs to/from VLSR1 and VLSR3. The following statements will do so:

```
set local-id port 3
set local-id group 1000 add 3
set local-id tagged-group 150 add 3
```

You have to restart dragon daemon for these statement to take into effect (on each VLSR, execute `dragon.sh restart-vlsr`). Or, you can enter these statements directly into the DRAGON CLI (e.g. telnet red-vlsr[1 or 3] 2611) without restarting the dragon daemon.

Once you have configured a number of Local-IDs for general use, or several specific Local-IDs, you are ready to login to the DRAGON CLI on VLSR1 (or VLSR3) and edit an LSP. When using the tagged-group Local-ID, the vtag must match on both the source and destination. This is fundamentally an issue associated with flat VLANs. Unlike MPLS labels, the VLAN tag is not actually swapped at each node and must be maintained end-to-end. There are some emerging scenarios where this constraint will be reduced or eliminated, but the standards and generally available layer2 products do not support this capability yet.

### 3) Circuit creation – setting up LSPs

Finally...It is time to actually configure and provision the LSP. For this first exercise, we will be using the DRAGON CLI to create the LSP. This is a two step process: 1) define and edit the LSP and its parameters, 2) “commit” the LSP to place it into service. Afterward, we’ll explore some of the ways to inspect the status of an LSP or to debug issues in the control plane protocols.

Login to the vlsr1 DRAGON CLI on port 2611 and configure the LSP:

```
red_vlsr1_pc:~# telnet localhost 2611
Password:  *****

red-vlsr1-dragon> edit lsp test1
red-vlsr1-dragon(edit-lsp-test1)# set source ip-address 192.168.1.4 port 3
destination ip-address 192.168.1.8 port 3
red-vlsr1-dragon(edit-lsp-test1)# set bandwidth eth100M swcap l2sc encoding
ethernet gpid ethernet
red-vlsr1-dragon(edit-lsp-test1)# set vtag any
red-vlsr1-dragon(edit-lsp-test1)# exit

red-vlsr1-dragon> show lsp
                        **LSP status summary**

Name          Status      Dir   Source (IP/LSP ID)  Destination (IP/Tunnel ID)
-----
test1         Edit        =>   192.168.1.4        192.168.1.8
                        3                      3
```

Now, commit the LSP to put it into service:

```
red-vlsr1-dragon> commit lsp test1
```

The LSP will be set between red-vlsr1-sw port 3 to red-vlsr3-sw port 3.

You can check the status of the LSP with:

```
red-vlsr1-dragon> show lsp
                        **LSP status summary**
Name           Status      Dir   Source (IP/LSP ID)  Destination (IP/Tunnel ID)
-----
test1          In service => 192.168.1.4        192.168.1.8
                                   3                    3
```

You can see a bit more detail about the LSP by running the command `show lsp test1` in the DRAGON CLI.

It is important to understand that while the LSP has been set up and is operational, the interface on the End System(s) still needs some additional work. The easiest way to take advantage of the point to point layer2 path that has been established is to configure it as an IP interface. **The LSP is in port-to-port/untagged mode, so the data plane interface is eth1 on ES1 and ES2.** The teams can choose any available IP subnet for the link, run `ifconfig` on the data interfaces, and then ping across it.

Login to es2:

```
red_es2_pc:~# ifconfig eth1 10.0.0.2 netmask 255.255.255.252
```

Login to es1:

```
red_es1_pc:~# ifconfig eth1 10.0.0.1 netmask 255.255.255.252
red_es1_pc:~# ping 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.070 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=0.050 ms
...
```

Please leave the ping command running while you proceed to the next step.

As we stated before, the TE address associated with the interface is **not** useable for IP packet forwarding. It is not actually configured on any of the data plane interfaces. It is only configured in the control plane protocols as a means of identifying the paired interfaces/ports of the data plane topology.

Finally, delete the LSP by running the following command in the DRAGON CLI:

```
red_vlsr1_pc:~# telnet localhost 2611
Password: *****
red-vlsr1-dragon> delete lsp test1
```

If you kept the ping running from the previous step, you should be able to see the pings stop when you execute the `delete lsp` command. We would also recommend that you login to your Ethernet

switches and run the `show vlan` command to see the VLAN port memberships changing during the LSP setup and teardown process.

After you are done with your link, it's always a good practice to delete the IP address from your end systems:

```
red_es1_pc:~# ifconfig eth1 0.0.0.0
```

```
red_es2_pc:~# ifconfig eth1 0.0.0.0
```

If your group finishes exercise #1 early, try running the following `wireshark` commands. Execute these commands on VLSR1 or VLSR3 to dump the GMPLS RSVP/OSPF traffic on interface `gre3`:

```
red_vlsr1_pc:~# tshark -nV -i red-gre3 ip proto 46  
red_vlsr1_pc:~# tshark -nV -i red-gre3 ip proto 89
```

While `tshark` is running, login to the DRAGON CLI and provision the same circuit as above. You may want to pipe the output to the `less` command, or write the output to a file for further analysis.

The output of the first `tshark` command (`ip proto 46`) will dump all of the GMPLS-RSVP-TE packets that are sent or received on `red-vlsr1`'s `red-gre3` interface. The second `tshark` command (`ip proto 89`) will dump all of the GMPLS-OSPF-TE packets on the same interface.

## Exercise #2: Intra-domain Provisioning with OSCARS

Objective 1: Configure OSCARS over the GMPLS control and data plane

Objective 2: Reserve and provision Intra-domain circuits using OSCARS

Upon completion of Exercise 1, we have constructed several independent networks each capable of dynamically provisioning LSPs across their respective domains. In this exercise, we will introduce a higher level set of services which make use of the capabilities constructed in Exercise 1. The DRAGON control plane provides the ability to dynamically provision across multi-layer, multi-technology network infrastructures. The OSCARS software adds to this a Web Service based AAI (Authentication, Authorization Infrastructure) and Scheduling capabilities. In addition, we will utilize the OSCARS Web Service mechanisms for Inter-Domain messaging associated with multi-domain circuit provisioning. The Inter-Domain component is covered in Exercise 3.

OSCARS is a Java application which uses X.509 certificates for user authentication, SSL Encryption for messaging, and signed SOAP messages for user data containers. Additional details regarding the OSCARS software and architecture is contained documentation referenced here [ref].

As in Exercise #1 with the DRAGON software, we have pre-installed OSCARS and its dependencies in each pod. For more information on how OSCARS was deployed in the pod environment, please refer to sections 1 through 3 of Appendix D, DCN Software Suite: OSCARS Inter-Domain Controller (IDC) Installation Guide. The task for Exercise 2 is to configure OSCARS such that it will integrate with the underlying DRAGON control plane software.

### Procedure:

**SSH to the NARB (192.168.x.10) in your domain using the following login information:**

**Username: tomcat55**

**Password: dragon**

It is very important that you follow the remaining steps while logged in as user `tomcat55` because special environment variables have been defined in `/home/tomcat55/.bash_profile` which are necessary for correct operation of Apache Tomcat.

### Step 1: Configure OSCARS

The `oscars.properties` file is the main area in which the OSCARS IDC retrieves installation-specific settings. These include settings for accessing the MySQL database, AAA, the perfSONAR Lookup Service, interacting with DRAGON, and more. The `oscars.properties` file is located on your NARB/IDC PC at:

```
/usr/local/tomcat/shared/classes/server/oscars.properties
```

Your installation comes with a default `oscars.properties` file. Edit the default file and update the sections marked in bold below for your pod. You will need to put in the correct addresses for your VLSRs in the `pss.dragon` section.

```
### Example main properties file.
# Values are filled for an example dragon/terce site.

### Hibernate section

# login name and password used by the OSCARS server to log into the MySQL
# database - change for your site
hibernate.connection.username=oscars
hibernate.connection.password=oscars

#Local URLs. If not set will assume port 8443 and /axis2/services/OSCARS[Notify]
idc.url=https://idc.red.pod.lan:8443/axis2/services/OSCARS
notify.ws.broker.url=https://idc.red.pod.lan:8443/axis2/services/OSCARSNotify

### OSCARS sections ###

### AAA configuration - change for your site
aaa.salt=os

# random names for cookies - may be changed per site
aaa.userName=oscars
aaa.sessionName=oscarssess
# whether cookie has to be sent over SSL; change this to 1 once you have SSL
# set up
aaa.secureCookie=1

#### WBUI setting ###
# set default layer to 2 or 3. Set to layer 2 if
# configured to use dragon. Set to 3 for Juniper routers.
wbui.defaultLayer=2

### pathfinder configuration
# can be either 0 (don't find path, only used for aaa testing) or 1
pathfinder.findPath=1

# currently can be traceroute or terze
pathfinder.pathMethod=perfsonar,terce

#TEDB configuration - can be terze or oscars
tedb.tedbMethod=terce

#TERCE configuration
terce.url=http://127.0.0.1:8080/axis2/services/TERCE

##Lookup Service and topology Service configuration
lookup.useGlobal=0
lookup.home.1=http://idc.green.pod.lan:9995/perfSONAR_PS/services/hLS
lookup.topology.1=http://idc.green.pod.lan:8012/perfSONAR_PS/services/topology
perfsonar.topology_url=http://idc.green.pod.lan:8012/perfSONAR_PS/services/topology

#notification messages
# These properties determine the types of notifications generated by the IDC.
# the modules property can be set to email for email notifications
# It may also be set to ws for web service notifications.
#
notify.observer.1=net.es.oscars.notify.EmailObserver
notify.observer.2=net.es.oscars.notify.FileWriterObserver
#notify.observer.3=net.es.oscars.notify.WSObserver
```

```

# External services
# Properties for tasks such as registering with the topology service
# or subscribing to other domain's notifications
#
external.service.1=subscribe
external.service.2=topology
external.service.3=lsRegister
external.service.4=lsDomainUpdate

# VLAN policy filter. The default is node but if you would like to be more/less
# restrictive about VLAN use you may set it here
#policy.vlanFilter.scope=port

#Notification broker config options
notifybroker.pep.1=net.es.oscars.notify.ws.policy.IDCEventPEP

#logging configuration - location of the per reservation logs
# should be ${CATALINA_HOME}/logs
# reservation logs will be put in a subdirectory: reservations
logging.rsvlogdir=/usr/local/tomcat/logs

## PSS configuration
# currently can be dragon or esnet
pss.method=dragon

pss.dragon.password=dragon
pss.dragon.ssh.portForward=1
pss.dragon.ssh.user=oscars
pss.dragon.ssh.key=/home/tomcat55/.ssh/id_rsa
pss.dragon.remotePort=2611
pss.dragon.hasNarb=1
pss.dragon.setERO=1
pss.dragon.vlsr1=127.0.0.1
pss.dragon.vlsr1.ssh=192.168.1.4
pss.dragon.vlsr2=127.0.0.1
pss.dragon.vlsr2.ssh=192.168.1.6
pss.dragon.vlsr3=127.0.0.1
pss.dragon.vlsr3.ssh=192.168.1.8

## mail that reservation notification is send to - change for your site
mail.webmaster=admin@red.pod.lan

#Other properties
mail.userAdd.subject=OSCARS user added

```

## Step 2: Verify the OSCARS installation

### 1) Start Apache Tomcat:

```

tomcat55@green-narb:~$ /usr/local/tomcat/bin/startup.sh
Using CATALINA_BASE:   /usr/local/tomcat
Using CATALINA_HOME:   /usr/local/tomcat
Using CATALINA_TMPDIR: /usr/local/tomcat/temp
Using JRE_HOME:        /usr/local/java5

```

Wait for the server finish starting by watching the output in

`/usr/local/tomcat/logs/catalina.out:`

```
tomcat55@green-narb:~$ tail -f /usr/local/tomcat/logs/catalina.out
...
Jul 16, 2008 2:54:51 PM org.apache.jk.server.JkMain start
INFO: Jk running ID=0 time=0/263 config=null
Jul 16, 2008 2:54:51 PM org.apache.catalina.storeconfig.StoreLoader load
INFO: Find registry server-registry.xml at classpath resource
Jul 16, 2008 2:54:51 PM org.apache.catalina.startup.Catalina start
INFO: Server startup in 14107 ms
```

In your web browser, visit [https://idc.\[yourColor\].pod.lan:8443](https://idc.[yourColor].pod.lan:8443)

If installation was successful, a web page will load with the Tomcat logo and a message that reads “If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!”

## 2) Verifying Axis2 Installation

In your web browser, visit [https://idc.\[yourColor\].pod.lan:8443/axis2/axis2-admin/](https://idc.[yourColor].pod.lan:8443/axis2/axis2-admin/) and login with:

Username: admin

Password: axis2

On the left bar, click on “Available Services” and verify that you see the OSCARS and TERCE services listed on the right. For more information about the Axis2 installation, please refer to sections 8.1 and 8.2 of Appendix D.

## 3) Verifying OSCARS Web-Based User Interface (WBUI) Installation

In your web browser, visit [https://idc.\[yourColor\].pod.lan:8443/OSCARS](https://idc.[yourColor].pod.lan:8443/OSCARS)

You will be presented with the OSCARS login form. However, you will not be able to login just yet. We need to create a new user account before attempting to login.

**Please note:** Internet Explorer is **NOT** currently supported by OSCARS – you will not be able to get past the login page if you are using Internet Explorer!



### Step 3: Creating the First User Account in OSCARS

As shown below, execute the `idc-useradd` script located in the `/home/tomcat55/dcn-software-suite-0.4/idc/tools/utils/` directory to create your first OSCARS user so that you can log into the web interface.

Add a new user for each member of your group using the example provided below.

```
tomcat55@green-narb:~$ cd /home/tomcat55/dcn-software-suite-0.4/idc/tools/utils
tomcat55@green-narb:~/dcn-software-suite-0.4/idc/tools/utils$ ./idc-useradd
* indicates a required field
Login*: oscars-admin
Password*: oscars
Confirm Password*: oscars
First Name*: OSCARS
Last Name*: Administrator
Cert Subject: [leave blank]
Cert Issuer: [leave blank]

1. Red Pod
2. Green Pod
3. Blue Pod
4. Yellow Pod
Select the user's organization (by number): 2

1. OSCARS-user
2. OSCARS-engineer
3. OSCARS-administrator
4. OSCARS-service
5. OSCARS-operator
6. OSCARS-site-administrator
7. OSCARS-publisher
8. OSCARS-may-specify-path
Select the user's role(s) (numbers separated by spaces): 2 3
Personal Description: [leave blank]
Email(Primary)*: myemail@mydomain.net
Email(Secondary): [leave blank]
Phone(Primary)*: 1234567890
Phone(Secondary): [leave blank]
New user 'oscars-admin' added.
tomcat55@green-narb:~/dcn-software-suite-0.4/idc/tools/utils$
```

### Step 4: Configure TERCE

TERCE is a web service component that acts as the topology exchange and route computation element for OSCARS. In the future, it will act as the intermediary between OSCARS and the NARB, but only static topology description and route files are supported in this release. To ensure the TERCE properties file can locate the static topology and route files in your domain, it must be edited:

Edit `/usr/local/tomcat/shared/classes/terce.conf/terce-ws.properties`

Change the properties file to look like the following (where *location-of-your-topology-file* is replaced with the custom path to your topology file and likewise for *location-of-your-static-routes-file*).

```
#static tedb properties
tedb.type=static
tedb.static.db.interdomain=location-of-your-topology-file
tedb.static.db.intradomain=location-of-your-topology-file

#static rce properties
rce.type=static

rce.static.file=location-of-your-static-routes-file
```

Modify the **terce-ws.properties** file to have the inter-domain and intra-domain db files point to `/usr/local/tomcat/shared/classes/terce.conf/tedb-inter.xml` and `/usr/local/tomcat/shared/classes/terce.conf/tedb-intra.xml`, respectively. The static RCE file should point to `/usr/local/tomcat/shared/classes/terce.conf/static-routes.xml`.

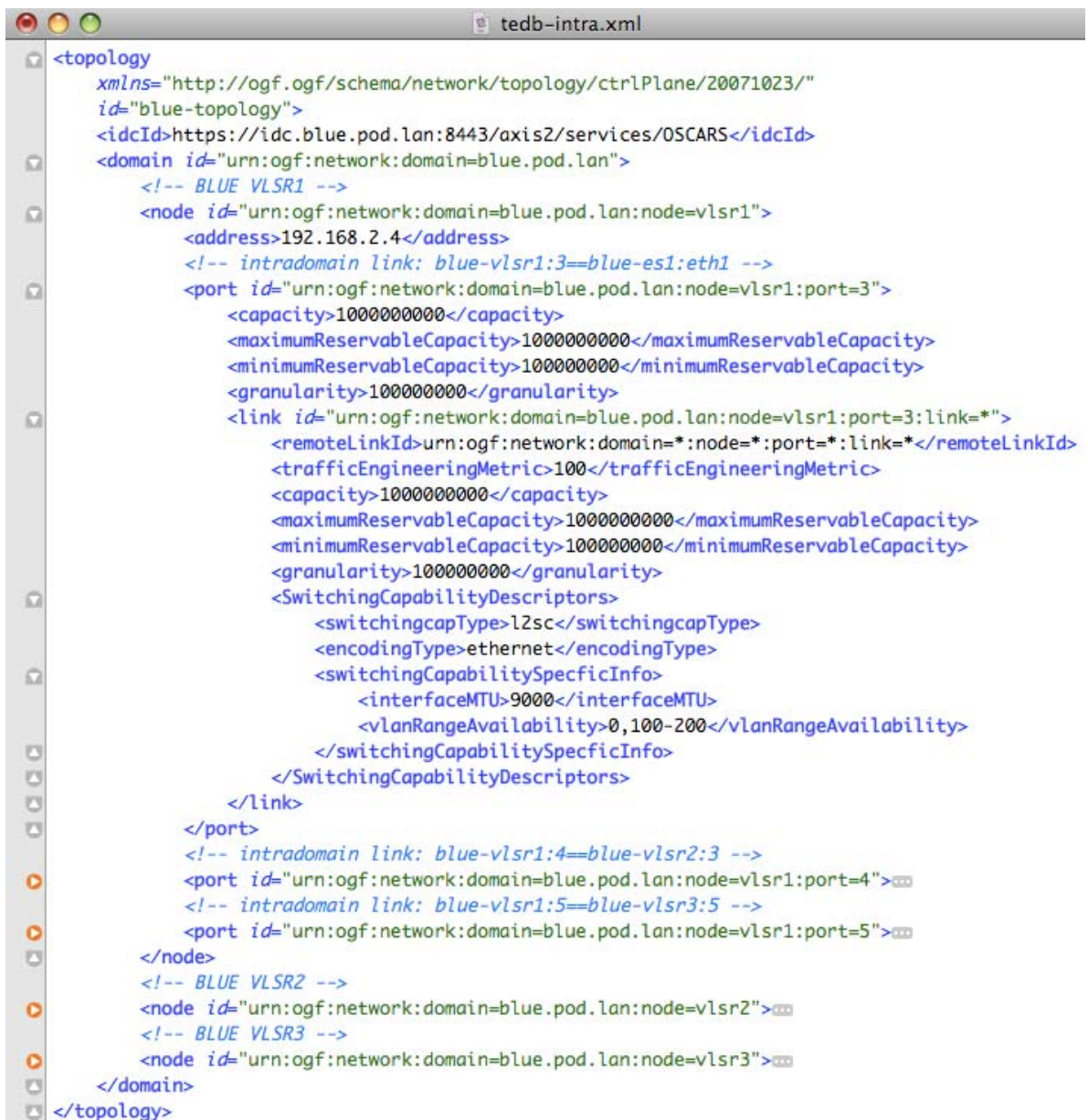
For more information about the format of the `tedb-intra.xml`, `tedb-inter.xml`, and `static-routes.xml` files, please see section 5 of Appendix D. We will be working with these files in the next two steps and examples of each file are provided.

## Step 5: Describing the data plane topology in XML format

As of the current release, OSCARS must be provided with a static XML description of the data plane topology. The next software release will gather this information dynamically from the DRAGON NARB. We have already put a working copy of this file on each NARB at the following location:

```
/usr/local/tomcat/shared/classes/terce.conf/tedb-intra.xml
```

Please examine this file to understand how it describes the data plane topology of your pod. The following is a screenshot of the blue pod's `tedb-intra.xml` file after being loaded into TextMate. TextMate understands the format of XML files so blocks can be folded or collapsed and the syntax is highlighted, making it easier to read.



```
<?xml version="1.0" encoding="UTF-8" ?>
<topology
  xmlns="http://ogf.ogf/schema/network/topology/ctrlPlane/20071023/"
  id="blue-topology">
  <idcId>https://idc.blue.pod.lan:8443/axis2/services/OSCARS</idcId>
  <domain id="urn:ogf:network:domain=blue.pod.lan">
    <!-- BLUE VLSR1 -->
    <node id="urn:ogf:network:domain=blue.pod.lan:node=vlsr1">
      <address>192.168.2.4</address>
      <!-- intradomain link: blue-vlsr1:3==blue-es1:eth1 -->
      <port id="urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=3">
        <capacity>1000000000</capacity>
        <maximumReservableCapacity>1000000000</maximumReservableCapacity>
        <minimumReservableCapacity>1000000000</minimumReservableCapacity>
        <granularity>100000000</granularity>
        <link id="urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=3:link="*">
          <remoteLinkId>urn:ogf:network:domain=*:node=*:port=*:link="*</remoteLinkId>
          <trafficEngineeringMetric>100</trafficEngineeringMetric>
          <capacity>1000000000</capacity>
          <maximumReservableCapacity>1000000000</maximumReservableCapacity>
          <minimumReservableCapacity>1000000000</minimumReservableCapacity>
          <granularity>100000000</granularity>
          <SwitchingCapabilityDescriptors>
            <switchingcapType>l2sc</switchingcapType>
            <encodingType>ethernet</encodingType>
            <switchingCapabilitySpecficInfo>
              <interfaceMTU>9000</interfaceMTU>
              <vlanRangeAvailability>0,100-200</vlanRangeAvailability>
            </switchingCapabilitySpecficInfo>
          </SwitchingCapabilityDescriptors>
        </link>
      </port>
      <!-- intradomain link: blue-vlsr1:4==blue-vlsr2:3 -->
      <port id="urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=4">
        <!-- intradomain link: blue-vlsr1:5==blue-vlsr3:5 -->
        <port id="urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=5">
      </node>
    <!-- BLUE VLSR2 -->
    <node id="urn:ogf:network:domain=blue.pod.lan:node=vlsr2">
      <!-- BLUE VLSR3 -->
    <node id="urn:ogf:network:domain=blue.pod.lan:node=vlsr3">
  </domain>
</topology>
```

## Step 6: Populate the Scheduling Database

The final step is to import the topology information from your XML file into the OSCARS scheduling database so that it can keep track of which resources are being used on the network. This is done by defining your local domain in the database and running `updateTopology.sh`.

### 1) Define Your Local Domain

You must define your local domain in the scheduling database so the `updateTopology.sh` script in the next step knows which links are local. This is done by running the `idc-domainadd` command in the `/home/tomcat55/dcn-software-suite-0.4/idc/tools/utils` directory. The example below shows the procedure that should be run on the red pod:

```
tomcat55@red-narb:~$ cd /home/tomcat55/dcn-software-suite-0.4/idc/tools/utils
tomcat55@red-narb:~/dcn-software-suite-0.4/idc/tools/utils$ ./idc-domainadd
Topology Identifier (i.e. mydomain.net)*: red.pod.lan
IDC URL*: https://idc.red.pod.lan:8443/axis2/services/OSCARS
Descriptive Name (for display purposes)*: red-pod
Abbreviated Name (for display purposes)*: red
Is this your IDC's local domain? [y/n] y
New domain 'red.pod.lan' added.
```

### 2) Run `updateTopology.sh`

The `updateTopology.sh` script syncs the database with your XML topology files. Follow the procedure below to run this script:

```
tomcat55@red-narb:~$ cd ~/dcn-software-suite-0.4/idc/tools/updatedbterce/
tomcat55@red-narb:~/dcn-software-suite-0.4/idc/tools/updatedbterce$
./updateTopology.sh http://127.0.0.1:8080/axis2/services/TERCE
DOMAIN: red.pod.lan
NODE: vlsr1
PORT: 3
LINK: *
PORT: 4
LINK: 11.1.2.1
PORT: 5
LINK: 11.1.3.1
NODE: vlsr2
PORT: 4
LINK: 11.1.4.1
PORT: 3
LINK: 11.1.2.2
NODE: vlsr3
PORT: 4
LINK: 11.1.4.2
PORT: 5
LINK: 11.1.3.2
PORT: 3
LINK: *
Complete.
tomcat55@red-narb:~/dcn-software-suite-0.4/idc/tools/updatedbterce$
```

If no error is returned, your database is now populated with the topology information from your XML topology file.

## Step 7: Schedule and Provision the intra-domain LSP

Now, let's try to provision an intra-domain LSP via the WBUI provided by OSCARS.

**Before you start provisioning via OSCARS WBUI, please ensure that the time set on your computer is valid – you must have a valid time zone setting with the current time in that time zone. The WBUI uses JavaScript to get the local time on your machine if you leave the “Time” fields blank (see below).**

- 1) Login to the Web-Based User Interface (WBUI)

Open a web browser and connect to “[https://idc.\[yourColor\].pod.lan:8443/OSCARS/](https://idc.[yourColor].pod.lan:8443/OSCARS/)”. Login with username **oscars-admin** and password **oscars**, or whatever username and password you chose when you ran the `idc-useradd` script in the previous step. Again, please note that Internet Explorer is **not** currently supported. You must use a Mozilla-based browser with the current OSCARS release.

**INTERNET<sup>2</sup>** **ESnet** **On-demand Secure Circuits and Advance Reservation System**  
A collaboration between [ESnet](#), [Internet2](#), [DANTE](#), and [ISI East](#)

January 20, 2009 16:19 User logged out.

Login/Logout

User Name:

Password:

Sign in via your OSCARS login and password to access this system. To find out about OSCARS and how it works, go to the [documentation](#). To obtain an account or request more information, email one of the contacts below.

More documentation can be found by clicking on the link at the bottom of this page.

[Documentation](#) | [ESnet](#) | [Berkeley Lab](#) | [Notice to Users](#)

Contacts: [Chin Guok](#), [David Robertson](#)

2) Once you have logged in, create a path reservation.

Click on the tab “Create Reservation”. The “source” and “destination” fields have the following format:

```
urn:ogf:network:domain=[color].pod.lan:node=[vlsr1 or vlsr3]:port=3:link=*
```

For example:

To refer to blue-es1, you would use:

```
urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=3:link=*
```

To refer to yellow-es2, you would use:

```
urn:ogf:network:domain=yellow.pod.lan:node=vlsr3:port=3:link=*
```

Recall that we configured the `lookup.url` parameter in the `oscars.properties` file. This parameter was set to [http://idc.green.pod.lan:9995/perfSONAR\\_PS/services/hLS](http://idc.green.pod.lan:9995/perfSONAR_PS/services/hLS).

To make provisioning easier, hostnames may be generated for the URNs on the edge of each network. This allows a user to enter a string such as `es1.red.pod.lan` instead of `urn:ogf:network:domain=red.pod.lan:node=vlsr1:port=3:link=*`.

This is precisely what the lookup service that is configured in the green pod is designed to do. You may use the URN format, or you may use the hostnames to refer to the source and destination nodes. The screenshot below shows an example of using hostnames for the source/destination fields, but the form could have been filled out using the URN notation. For more information about the Lookup Service, please see section 6.3 of Appendix D.

For example, you can set up an LSP between blue-es1 and blue-es2 with the following filled in the “Create Reservation” form:

```
Source: urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=3:link=*
Destination: urn:ogf:network:domain=blue.pod.lan:node=vlsr3:port=3:link=*
Bandwidth (Mbps): 1000
Description: describe what this LSP is for, i.e. testing
VLAN: [leave blank for any]
```

The following values will also work:

```
Source: es1.blue.pod.lan
Destination: es2.blue.pod.lan
Bandwidth (Mbps): 1000
Description: describe what this LSP is for, i.e. testing
VLAN: [leave blank for any]
```

Once the form is filled out, click the **Create Reservation** button. An example on the blue pod is given below.

January 20, 2009  
15:54

Reservation creation form

Reservations	Reservation Details	Create Reservation	User Profile	User List	Add User	Attributes	Institutions
Authorizations	Authorization Details	Login/Logout					

Required inputs are bordered in green. The source and destination can be topology identifiers, host names, or IP addresses, depending on the layer used. Click on the boxes associated with the start and end dates to bring up a calendar widget. The reservation time slot defaults to now, and now + 4 minutes, respectively, if you leave the dates and times empty.

WARNING: Entering a series of hops in the Path field may alter routing behavior for the selected flow. Hops can be topology identifiers, host names, or IP addresses, depending on the layer used. Note that the path field will expand to the number of lines occurring in the hops list.

Production circuit

Source:

Destination:

Path (series of hops):

Bandwidth (Mbps):  ( 1-10000 )

Description:  ( For our records )

Start date:  1/20/2009

Start time:  15:54

End date:  1/20/2009

End time:  15:58

Use layer 2 parameters  Use layer 3 parameters

VLAN:  tag, or range, e.g. 3000-3100

Source Port:

Destination Port:

You may leave all other fields blank. This will create an LSP with your required bandwidth, starting as soon as possible. Since you haven't put in the end time of this LSP, the default will leave this LSP up for 4 minutes. After 4 minutes, the LSP is automatically torn down. The VLAN field can be left blank, or filled in with the keyword 'any', or with a value between 100 and 200.

Once you click "Create Reservation", you will be taken to a page that shows the summary of your reservation and the status of the LSP that you have just set up.

At the top of the page, it will indicate if your reservation is successful or not. If the reservation is successfully, you will see your LSP in "ACCEPTED" status. If you click "Refresh" button on the top of that page, you will see the updated status of the LSP. If the IDC successfully put the LSP in service, the status will change to "ACTIVE" within about 15 seconds.

Your screen should look something like the following after clicking Create Reservation and Refresh button several times. If your screen indicates that the status is not ACTIVE, try clicking Refresh again.

January 20, 2009  
15:56

Reservation details for blue.pod.lan-3

Reservations	Reservation Details	Create Reservation	User Profile	User List	Add User	Attributes	Institutions
Authorizations	Authorization Details	Login/Logout					

NEW GRI

---

GRI	blue.pod.lan-3
Status	ACTIVE
User	oscars-admin
Description	LSP between blue-es1 and blue-es2
Start date	<input type="text" value="1/20/2009"/>
Start time	<input type="text" value="15:56"/>
End date	<input type="text" value="1/20/2009"/>
End time	<input type="text" value="16:00"/>
Created time	2009/01/20 15:56
Bandwidth (Mbps)	1000
Source	urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=3:link=*
Destination	urn:ogf:network:domain=blue.pod.lan:node=vlsr3:port=3:link=*
Intradomain hops	urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=3:link=*
	urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=5:link=11.2.3.1
	urn:ogf:network:domain=blue.pod.lan:node=vlsr3:port=5:link=11.2.3.2
	urn:ogf:network:domain=blue.pod.lan:node=vlsr3:port=3:link=*
Interdomain path	
VLAN	137
Tagged	true

In the above example, OSCARS has chosen to use VLAN 137, since we let it pick the VLAN tag instead of providing our own.

If you see the status change to FAILED, please check the output of the following files and work with the instructors to debug the problem. Please check that the start/end times are correct and that the time on your computer is valid if you left the “Time” fields blank.

```
/usr/local/tomcat/logs/catalina.out
/usr/local/tomcat/logs/oscars.log
```

After 4 minutes, you should see that the status changes to “FINISHED”.

To actually make use of your LSP, you need to follow the steps described in Step 4 below. The method used in Exercise #1 will not work because the edge ports are now in tagged/trunking mode –



the VLSRs are expecting the end systems to be sending tagged 802.1Q frames so we must use the vconfig command in Linux.

### 3) Other ways to check the LSP status

- a. You can go to your NARB/IDC machine and look at the schedule log file to check the status of the LSP.

```
tomcat55@blue-narb:~$ tail -f /usr/local/tomcat/logs/catalina.out
2008-07-15 22:14:01,273 [INFO] blue-vlsr1-pc>
2008-07-15 22:14:01,274 [INFO] edit lsp blue.pod.lan-1
2008-07-15 22:14:01,275 [INFO] blue-vlsr1-pc(edit-lsp-blue.pod.lan-1)#
2008-07-15 22:14:01,275 [INFO] blue-vlsr1-pc(edit-lsp-blue.pod.lan-1)#
2008-07-15 22:14:01,277 [INFO] set source ip-address 192.168.2.4 tagged-group 137 destination ip-
address 192.168.2.8 tagged-group 137
2008-07-15 22:14:01,278 [INFO] blue-vlsr1-pc(edit-lsp-blue.pod.lan-1)#
2008-07-15 22:14:01,280 [INFO] blue-vlsr1-pc(edit-lsp-blue.pod.lan-1)#
2008-07-15 22:14:01,281 [INFO] set bandwidth gige swcap l2sc encoding ethernet gpid ethernet
2008-07-15 22:14:01,282 [INFO] blue-vlsr1-pc(edit-lsp-blue.pod.lan-1)#
2008-07-15 22:14:01,283 [INFO] blue-vlsr1-pc(edit-lsp-blue.pod.lan-1)#
2008-07-15 22:14:01,284 [INFO] set ero-hop-ipv4 strict ip-address 11.4.3.1
2008-07-15 22:14:01,284 [INFO] blue-vlsr1-pc(edit-lsp-blue.pod.lan-1)#
2008-07-15 22:14:01,284 [INFO] blue-vlsr1-pc(edit-lsp-blue.pod.lan-1)#
2008-07-15 22:14:01,287 [INFO] set ero-hop-ipv4 strict ip-address 11.4.3.2
2008-07-15 22:14:01,287 [INFO] blue-vlsr1-pc(edit-lsp-blue.pod.lan-1)#
2008-07-15 22:14:01,288 [INFO] blue-vlsr1-pc(edit-lsp-blue.pod.lan-1)#
2008-07-15 22:14:01,289 [INFO] set vtag 137
2008-07-15 22:14:01,289 [INFO] blue-vlsr1-pc(edit-lsp-blue.pod.lan-1)#
2008-07-15 22:14:01,289 [INFO] blue-vlsr1-pc(edit-lsp-blue.pod.lan-1)#
2008-07-15 22:14:01,290 [INFO] exit
2008-07-15 22:14:01,290 [INFO] blue-vlsr1-pc>
2008-07-15 22:14:01,291 [INFO] blue-vlsr1-pc>
2008-07-15 22:14:01,310 [INFO] commit lsp blue.pod.lan-1
2008-07-15 22:14:01,310 [INFO] blue-vlsr1-pc>
2008-07-15 22:14:01,310 [INFO] created lsp blue.pod.lan-1
2008-07-15 22:14:01,311 [INFO] blue-vlsr1-pc>
2008-07-15 22:14:01,312 [INFO] show lsp blue.pod.lan-1
2008-07-15 22:14:01,312 [INFO] Src 192.168.2.4/137, dest 192.168.2.8/137
2008-07-15 22:14:01,312 [INFO] Generic TSPEC R=gige, B=gige, P=gige, m=100, M=1500
2008-07-15 22:14:01,312 [INFO] Encoding ethernet, Switching l2sc, G-Pid ethernet
2008-07-15 22:14:01,313 [INFO] Ingress Local ID Type: tagged group, Value: 137
2008-07-15 22:14:01,313 [INFO] Egress Local ID Type: tagged group, Value: 137.
2008-07-15 22:14:01,314 [INFO] E2E LSP VLAN Tag: 137.
2008-07-15 22:14:01,314 [INFO] Status: Commit
2008-07-15 22:14:01,314 [INFO] blue-vlsr1-pc>
2008-07-15 22:14:06,330 [INFO] blue-vlsr1-pc>
2008-07-15 22:14:06,331 [INFO] show lsp blue.pod.lan-1
2008-07-15 22:14:06,331 [INFO] Src 192.168.2.4/137, dest 192.168.2.8/137
2008-07-15 22:14:06,332 [INFO] GRI: 1727633764-2147483649
2008-07-15 22:14:06,332 [INFO] Generic TSPEC R=gige, B=gige, P=gige, m=100, M=1500
2008-07-15 22:14:06,332 [INFO] Encoding ethernet, Switching l2sc, G-Pid ethernet
2008-07-15 22:14:06,333 [INFO] Ingress Local ID Type: tagged group, Value: 137
2008-07-15 22:14:06,333 [INFO] Egress Local ID Type: tagged group, Value: 137.
2008-07-15 22:14:06,334 [INFO] E2E LSP VLAN Tag: 137.
2008-07-15 22:14:06,334 [INFO] Status: In service
2008-07-15 22:14:06,334 [INFO] blue-vlsr1-pc>
2008-07-15 22:14:06,338 [INFO] blue.pod.lan-1 is IN SERVICE
```

You will see the CLI commands that OSCARS generates based on your WBUI request.

- b. You can also go to the “source” or “destination” of the LSP to check the status at DRAGON CLI. For the example above, you can login to blue-vlsr1; telnet to localhost 2611 and check on the LSP.

```

blue-vlsr1-pc> show lsp
                        **LSP status summary**

Name          Status      Dir    Source (IP/LSP ID)  Destination (IP/Tunnel ID)
-----
blue.pod.lan-1
              In service <=> 192.168.2.4         192.168.2.8
                               137                  137

green-vlsr1-pc> show lsp blue.pod.lan-1
Src 192.168.2.4/137, dest 192.168.2.8/137
GRI: 1727633764-2147483649
Generic TSPEC R=gige, B=gige, P=gige, m=100, M=1500
Encoding ethernet, Switching l2sc, G-Pid ethernet
Ingress Local ID Type: tagged group, Value: 137
Egress Local ID Type: tagged group, Value: 137.
E2E LSP VLAN Tag: 137.
Status: In service

```

4) As shown above, **the LSP is in port-to-port/tagged mode, so the data plane interface is eth1.<vlan>, not eth1 as in the previous example.** Take a look at the VLAN that you have been assigned; in the above example, it's **vlan 137**.

Login to green\_es1 as “root”:

```

green_es1_pc:~# /sbin/modprobe 8021q
green_es1_pc:~# vconfig add eth1 137
Added VLAN with VID == 137 to IF -:eth1:-
green_es1_pc:~# ifconfig eth1.137 10.0.0.1 netmask 255.255.255.252

```

Then, login to green\_es2 as “root”, and repeat what you did for green\_es1, except that you should put the other side of the /30 to eth1.137.

Now you should be able to ping across your new LSP.

5) Schedule a “book ahead” reservation

Repeats steps 1-3 above, but fill in the start/end time fields. For example, try to “schedule” an LSP to come up 4 minutes in the future and stay active for 10 minutes.

## Step 8: Provision a circuit using the example Java client

Circuits do not exclusively need to be setup using the Web-Based User Interface. They may be setup by an application, such as a Java program. An example Java client is included in the DCN Software Suite. This Java client is designed to use the Web Services interface to send a signed SOAP message to OSCARS. The client has been pre-loaded with an X.509 user certificate. In order for the client to be able to create a reservation, we must first add a user to OSCARS so that it recognizes the X.509 certificate presented by the client. Run the `idc-useradd` script with the following parameters to create this user for the example Java client:

```

tomcat55@red-narb:~$ cd /home/tomcat55/dcn-software-suite-0.4/idc/tools/utils
tomcat55@red-narb:~/dcn-software-suite-0.4/idc/tools/utils$ ./idc-useradd
* indicates a required field
Login*: alice
Password*: anyPassword (password will not echo to the console)
Confirm Password*: anyPassword (password will not echo to the console)
First Name*: Alice
Last Name*: Alice
Cert Subject: CN=Alice, OU=OASIS Interop Test Cert, O=OASIS
Cert Issuer: [leave blank]

1. Red Pod
2. Green Pod
3. Blue Pod
4. Yellow Pod
Select the user's organization (by number): 1

1. OSCARS-user
2. OSCARS-engineer
3. OSCARS-administrator
4. OSCARS-service
5. OSCARS-operator
6. OSCARS-site-administrator
7. OSCARS-publisher
8. OSCARS-may-specify-path
Select the user's role(s) (numbers separated by spaces): 1
Personal Description: [leave blank]
Email(Primary)*: alice@alicedomain.net
Email(Secondary): [leave blank]
Phone(Primary)*: 1234567890
Phone(Secondary): [leave blank]
New user 'alice' added.

```

Next, setup the Java client by creating a new file in the following directory:

```

tomcat55@red-narb:~ $ cd ~/dcn-software-suite-0.4/idc/examples/javaClients/
tomcat55@red-narb:~/dcn-software-suite-0.4/idc/examples/javaClients$ emacs
myCircuit.properties

```

The **myCircuit.properties** must contain parameters as shown below. Please view the **example.12.properties** (that's 12 as in layer 2, **not** the number 12) in the same directory for a description of all of the available parameters.

```

interactive=0
url1=https://idc.red.pod.lan:8443/axis2/services/OSCARS/
gri=
layer=2
sourceEndpoint=es1.red.pod.lan
destEndpoint=es2.red.pod.lan
vtag=any
bandwidth=100
description=testing-java-client
pathSetupMode=timer-automatic
start_time=
end_time=
duration=0.5

```

Launch the example Java client by using the **createRes.sh** script. You may view the status of the reservation in the WBUI, or with the **listRes.sh** script in the `javaClients` directory.

```
tomcat55@red-narb:~/dcn-software-suite-0.4/idc/examples/javaClients$  
./createRes.sh -pf myCircuit.properties
```

## Exercise #3: Inter-domain Provisioning with OSCARS

Objective: In this exercise, we will configure the OSCARS for inter-domain scheduling and provisioning.

Inter-domain operations include use of the IDC protocol. This protocol is a product of the DICE Control Plane Working Group. DICE is an acronym for an informal collaboration between Dante, Internet2, Canarie and ESNet. The DICE control plane working group has met over the past 2 years to define the architecture and protocol use by an IDC. The protocol has been implemented by Internet2, ESNet, and GEANT and currently interconnects dynamic circuit networks at all three organizations.

The IDC protocol defines messages for reserving network resources, signaling resource provisioning, gathering information about previously requested resources, and basic topology exchange. These messages are defined in a SOAP web service format. Since all messages are defined using SOAP, the protocol also utilizes a few external web service protocols and XML descriptions for features such as security and topology description.

In the previous exercise, we scheduled and provisioned intra-domain LSPs. In this exercise, we will connect the four pod domains as a star:

- Green is the hub.
- Red-vlsr3 is connected to Green-vlsr1.
- Blue-vlsr2 connects to Green-vlsr2.
- Yellow-vlsr1 connects to Green-vlsr3.
- Each pod, except green, will have one inter-domain link, while green has 3 inter-domain links.

Inter-domain signaling works slightly different than intra-domain signaling: signaling occurs end-to-end in an intra-domain LSP. When it comes to inter-domain LSP, the signaling occurs within the intra-domain basis and there is no signaling via the inter-domain link. For example, if you want to provision a link from red-es1 to blue-es2.

- The red-IDC will set up the LSP from red-sw1 port 3 to red-sw3 port 7.
- The green-IDC will setup the LSP from green-sw1 port 7 to green-sw2 port7.
- The blue-IDC will setup the LSP from blue-sw2 port 7 to blue-sw3 port 3.

### Procedure:

#### Step 1: Design and implement the inter-domain data and control plane

Refer to Appendix B. Add the appropriate data circuit IDs, i.e. D11, D12 and D13, to your list of data links created in Exercise #1.

Go to the lab and install the appropriate Ethernet cabling for the desired data plane connectivity.

## Step 2: Configure OSCARS for inter-domain

In Exercise #2, the XML topology file you were given (`tedb-intra.xml`) already contained the inter-domain link. Thus, the inter-domain link has already been added to the OSCARS scheduling database.

In this step, you will generate a certificate signing request, get it signed by the CA, install the signed certificate, tell OSCARS about the other domains, and where exactly it may reach your neighboring IDC.

**Please stop here as we would like to go through the following steps together.**

### 1) Generating a Server Certificate for Inter-Domain Requests

You must generate a certificate that your domain will pass to other domains. This certificate is stored in the following keystore file:

```
/usr/local/tomcat/shared/classes/repo/OSCARS.jks
```

These three steps will create a certificate for your domain:

1. SSH to the NARB/IDC with username `tomcat55`. Generate a certificate and certificate signing request (CSR):

In our exercise, the green pod is the only CA that's signing the certificate for other pods. Use the following commands to verify that the green pod CA is installed in your keystores:

```
> ssh tomcat55@red-narb
tomcat55@red-narb:~$ cd ~/dcn-software-suite-0.4/idc/tools/utils/
tomcat55@red-narb:dcn-software-suite-0.4/idc/tools/utils$ ./idc-certview
Choose the keystore or file with the certificate you'd like to view:
  1. OSCARS.jks - stores the private key for sending and public key for receiving
messages from other IDCs
  2. ssl-keystore.jks - stores SSL certificates of other IDCs running HTTPS
  3. Open certificate file...
Enter choice: 1

Choose the certificate you wish to view:
  1. greenca
  2. ca
  3. root
  4. dcstestroot
  5. esnetroot
  6. doegridsca
  7. geant
Enter choice: 1

-- Certificate details

Alias name: greenCA
Creation date: Jan 14, 2008
Entry type: trustedCertEntry

Owner: CN=idc.green.pod.lan, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=US
Issuer: CN=idc.green.pod.lan, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=US
```

```
Serial number: 89b4999737fcd0ab
Valid from: Mon Jan 14 18:51:54 EST 2008 until: Thu Jan 11 18:51:54 EST 2018
Certificate fingerprints:
    MD5: EB:FC:FD:B9:78:40:61:B1:32:D0:64:CE:37:DD:CD:29
    SHA1: A5:93:01:22:DA:FD:52:B6:F1:88:6C:64:78:97:01:B7:1E:02:BC:6C
```

The above `idc-certview` command shows that **greenCA** has been installed in your machine as a trusted CA. The other listings are default certificates that may be used for testing but should be deleted in production environments. They will not be used in this course and can be ignored.

You will now create your own private key and certificate using the `idc-certadd` script:

```
tomcat55@red-narb:~$ cd ~/dcn-software-suite-0.4/idc/tools/utils/
tomcat55@red-narb:/usr/local/tomcat/shared/classes/repo$ ./idc-certadd
What would you like to do?
    1. Create a new certificate my IDC will use in outgoing messages to other IDCs
    2. Import a certificate created using choice 1 that was signed by a CA
    3. Trust a CA or another IDC's certificate
Enter choice: 1

-- You have chosen to create a new certificate for sending messages to other IDCs.

Enter an alias for this certificate: redidc
How many days will this certificate be valid?: 3650
-- Using keystore password from /usr/local/tomcat/shared/classes/repo/rampConfig.xml
What is your first and last name?
[Unknown]: idc.red.pod.lan
What is the name of your organizational unit?
[Unknown]:
What is the name of your organization?
[Unknown]:
What is the name of your City or Locality?
[Unknown]:
What is the name of your State or Province?
[Unknown]:
What is the two-letter country code for this unit?
[Unknown]: US
Is CN=idc.green.pod.lan, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=US correct?
[no]: yes

-- Certificate created.
Would you like to generate a Certificate Signing Request (CSR) y/n? y
What filename should I give the CSR?: redidc.csr
-- Certificate Signing Request saved in file red-idc.csr
--- Please send red-idc.csr to your CA for signing.
--- You may then import your signed certificate by running idc-certadd and choosing
option 2.
--- Send the following X.509 subject to your neighboring IDCs: CN=idc.red.pod.lan,
OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=US
```

**NOTE: Do not use ~ in the path of the CSR filename, the script has been known to throw errors when this character is used. Also, if you need to generate a new CSR for any reason then you may do so with the `idc-certsignreq` script.**

2. Email `redidc.csr` to the Certificate Authority (CA), in our case the workshop instructors, to get your certificate signed. After the instructors sign your certificate signing request, they will e-mail you back the signed certificate.
3. The instructors should e-mail you back a file with a `.cer` extension. You will import the signed certificate by running `idc-certadd` again:

```
tomcat55@red-narb:~$ cd ~/dcn-software-suite-0.4/idc/tools/utils/
tomcat55@red-narb:/usr/local/tomcat/shared/classes/repo$ ./idc-certadd
What would you like to do?
  1. Create a new certificate my IDC will use in outgoing messages to other IDCs
  2. Import a certificate created using choice 1 that was signed by a CA
  3. Trust a CA or another IDC's certificate
Enter choice: 2

-- You have chosen to import a signed certificate for talking to other domains.

Enter the filename of your signed certificate: /home/tomcat55/redidc.cer
-- Using keystore password from
/usr/local/tomcat/shared/classes/repo/rampConfig.xml
-- Using certificate with the alias redidc
--- If this is not the correct alias please exit (Ctrl-C) and modify the
    <ramp:user> tag in rampConfig.xml

-- Signed certificate imported
--- Send the following X.509 subject to your neighboring IDCs: CN=idc.red.pod.lan,
    OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=US
```

If there are no errors then you may view the signed certificate with the following command:

```
tomcat55@red-narb:~$ cd ~/dcn-software-suite-0.4/idc/tools/utils/
tomcat55@red-narb:dcn-software-suite-0.4/idc/tools/utils$ ./idc-certview
Choose the keystore or file with the certificate you'd like to view:
  1. OSCARS.jks - stores the private key for sending and public key for
    receiving messages from other IDCs
  2. ssl-keystore.jks - stores SSL certificates of other IDCs running HTTPS
  3. Open certificate file...
Enter choice: 1

Choose the certificate you wish to view:
  1. dcstestroot
  2. greenca
  3. ca
  4. esnetroot
  5. geant
  6. test
  7. redidc
  8. root
  9. doegridsca
Enter choice: 7

-- Certificate details

Alias name: redidc
```



```

Creation date: Jan 19, 2008
Entry type: keyEntry
Certificate chain length: 2
Certificate[1]:
Owner: CN=Unknown, OU=Unknown, O=Unknown, ST=Unknown, C=US
Issuer: CN=idc.green.pod.lan, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=US
Serial number: 10000e
Valid from: Sat Jan 19 15:26:32 EST 2008 until: Tue Jan 16 15:26:32 EST 2018
Certificate fingerprints:
    MD5: 7E:AA:97:DE:E2:39:B7:67:35:34:61:1D:ED:84:13:EA
    SHA1: 4C:EC:3D:0C:0D:78:33:DF:5D:42:9B:E6:8E:49:1F:9B:A1:B3:79:EC
Certificate[2]:
Owner: CN=idc.green.pod.lan, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=US
Issuer: CN=idc.green.pod.lan, OU=Unknown, O=Unknown, L=Unknown, ST=Unknown, C=US
Serial number: 89b4999737fcd0ab
Valid from: Mon Jan 14 18:51:54 EST 2008 until: Thu Jan 11 18:51:54 EST 2018
Certificate fingerprints:
    MD5: EB:FC:FD:B9:78:40:61:B1:32:D0:64:CE:37:DD:CD:29
    SHA1: A5:93:01:22:DA:FD:52:B6:F1:88:6C:64:78:97:01:B7:1E:02:BC:6C

```

You should see the display is much longer than what you saw in the output of the first `idc-certview` command for the greenCA's root certificate on the previous page. This output shows that a chain of trust has been established by the greenCA signing your certificate.

## 2) Making your IDC Aware of Other Domains

You must add an entry for both direct neighbors and downstream domains using the `idc-domainadd` command. This will add the corresponding rows to the `domains` table in the `bs` MySQL database on your IDC. For more information about this step, see section 7.1 of Appendix D.

First, add your directly connected neighbor – the green pod. Below is an example of how this is done on the red pod:

```

tomcat55@red-narb:~$ cd /home/tomcat55/dcn-software-suite-0.4/idc/tools/utils
tomcat55@red-narb:~/dcn-software-suite-0.4/idc/tools/utils$ ./idc-domainadd
Topology Identifier (i.e. mydomain.net)*: green.pod.lan
IDC URL*: https://idc.green.pod.lan:8443/axis2/services/OSCARS
Descriptive Name (for display purposes)*: green-pod
Abbreviated Name (for display purposes)*: green

1. Red Pod
2. Green Pod
3. Blue Pod
4. Yellow Pod
5. New...
Select the organization associated with this domain (by number): 2
Is this your IDC's local domain? [y/n] n
New domain 'green.pod.lan' added.

```

From the red pod's perspective, entries must also be added for downstream domains. We can enter Unknown in the IDC URL field, since we may not know (or be able to reach) the IDC directly from our local domain. The red IDC will never need to directly contact the IDC in either the blue or yellow domains. These steps are currently necessary so that the local IDC will recognize the topology identifier of downstream domains during provisioning.

```
tomcat55@red-narb:~$ cd /home/tomcat55/dcn-software-suite-0.4/idc/tools/utis
tomcat55@red-narb:~/dcn-software-suite-0.4/idc/tools/utis$ ./idc-domainadd
Topology Identifier (i.e. mydomain.net)*: blue.pod.lan
IDC URL*: Unknown
Descriptive Name (for display purposes)*: blue-pod
Abbreviated Name (for display purposes)*: blue

1. Red Pod
2. Green Pod
3. Blue Pod
4. Yellow Pod
5. New...
Select the organization associated with this domain (by number): 3
Is this your IDC's local domain? [y/n] n
New domain 'blue.pod.lan' added.
```

```
tomcat55@red-narb:~$ cd /home/tomcat55/dcn-software-suite-0.4/idc/tools/utis
tomcat55@red-narb:~/dcn-software-suite-0.4/idc/tools/utis$ ./idc-domainadd
Topology Identifier (i.e. mydomain.net)*: yellow.pod.lan
IDC URL*: Unknown
Descriptive Name (for display purposes)*: yellow-pod
Abbreviated Name (for display purposes)*: yellow

1. Red Pod
2. Green Pod
3. Blue Pod
4. Yellow Pod
5. New...
Select the organization associated with this domain (by number): 4
Is this your IDC's local domain? [y/n] n
New domain 'yellow.pod.lan' added.
```

Similar to the example Java client, we must now add a user so that OSCARS will recognize requests coming from a directly connected neighbor. Run the `idc-useradd` command and add a new user for the green IDC, using `CN=idc.green.pod.lan, OU=Unknown, O=Unknown, ST=Unknown, C=US` as the certificate subject. This user will need the `OSCARS-engineer` and `OSCARS-service` roles in order to have the correct privileges to signal paths via the Web Services interface.

```
tomcat55@red-narb:~$ cd /home/tomcat55/dcn-software-suite-0.4/idc/tools/utis
tomcat55@red-narb:~/dcn-software-suite-0.4/idc/tools/utis$ ./idc-useradd
* indicates a required field
Login*: greenidc
Password*: anyPassword
Confirm Password*: anyPassword
First Name*: Green
Last Name*: IDC
Cert Subject: CN=idc.green.pod.lan, OU=Unknown, O=Unknown, ST=Unknown, C=US
Cert Issuer: [leave blank]
```

```

1. Red Pod
2. Green Pod
3. Blue Pod
4. Yellow Pod
Select the user's organization (by number): 2

1. OSCARS-user
2. OSCARS-engineer
3. OSCARS-administrator
4. OSCARS-service
5. OSCARS-operator
6. OSCARS-site-administrator
7. OSCARS-publisher
8. OSCARS-may-specify-path
Select the user's role(s) (numbers separated by spaces): 2 4
Personal Description: [leave blank]
Email(Primary)*: admin@green-domain.net
Email(Secondary): [leave blank]
Phone(Primary)*: 1234567890
Phone(Secondary): [leave blank]
New user 'greenidc' added.

```

### 3) Enable inter-domain notifications

The next step is to enable inter-domain notifications. In the OSCARS implementation notification messages are distributed by an external service called the NotificationBroker (NB). The IDC and NB communicate via web service messages and the NB forwards those messages to subscribers that wish to know about the IDC's activities. Potential subscribers not only include neighboring IDCs but also end-users, accounting services, monitoring services, etc. Currently the NotificationBroker and OSCARS run on the same machine but could run on separate machines as they are independent web services. One reason that they commonly run on the same machine is so that they can share an authentication and authorization database. The NB it needs an entry for the IDC in its users table so it can authenticate notifications published by the IDC. We may do this by creating a user that has the certificate subject of the local IDC associated with it using the `idc-useradd` command:

```

tomcat55@red-narb:~$ cd /home/tomcat55/dcn-software-suite-0.4/idc/tools/utils
tomcat55@red-narb:~/dcn-software-suite-0.4/idc/tools/utils$ ./idc-useradd
* indicates a required field
Login*: localidc
Password*: anyPassword
Confirm Password*: anyPassword
First Name*: Local
Last Name*: IDC
Cert Subject: CN=idc.red.pod.lan, OU=Unknown, O=Unknown, ST=Unknown, C=US
Cert Issuer: [leave blank]

1. Red Pod
2. Green Pod
3. Blue Pod
4. Yellow Pod
Select the user's organization (by number): 1

1. OSCARS-user
2. OSCARS-engineer
3. OSCARS-administrator
4. OSCARS-service
5. OSCARS-operator
6. OSCARS-site-administrator

```

```
7. OSCARS-publisher
8. OSCARS-may-specify-path
Select the user's role(s) (numbers separated by spaces): 4 7
Personal Description: [leave blank]
Email(Primary)*: admin@red-domain.net
Email(Secondary): [leave blank]
Phone(Primary)*: 1234567890
Phone(Secondary): [leave blank]
New user 'localidc' added.
```

After adding this user the next step is to open **oscars.properties** and uncomment the line in bold below so that web service notification will be published by the IDC:

```
...
# These properties determine the types of notifications generated by the IDC.
# the modules property can be set to email for email notficiations
# It may also be set to ws for web service notifications.
#
notify.observer.1=net.es.oscars.notify.EmailObserver
notify.observer.2=net.es.oscars.notify.FileWriterObserver
notify.observer.3=net.es.oscars.notify.WSObserver
...
```

#### 4) Enable inter-domain link in XML topology file

In order to enable the inter-domain link to the green pod, you must uncomment a section from the XML topology file. Please load up the following file into a text editor:

```
/usr/local/tomcat/shared/classes/terce.conf/tedb-intra.xml
```

The image below depicts the inter-domain link between the blue and green pods. Find the appropriate section in the `tedb-intra.xml` file for your pod and remove the comment lines, as shown by the square boxes in the figure below:

```

tedb-intra.xml

<!-- intradomain link: blue-vlsr2:3==blue-vlsr1:4 -->
<port id="urn:ogf:network:domain=blue.pod.lan:node=vlsr2:port=3">...

<!-- interdomain link: blue-vlsr2:7==green-vlsr2:7 -->
<!--
<port id="urn:ogf:network:domain=blue.pod.lan:node=vlsr2:port=7">
  <capacity>1000000000</capacity>
  <maximumReservableCapacity>1000000000</maximumReservableCapacity>
  <minimumReservableCapacity>1000000000</minimumReservableCapacity>
  <granularity>100000000</granularity>

  <link id="urn:ogf:network:domain=blue.pod.lan:node=vlsr2:port=7:link=*">
    <remoteLinkId>urn:ogf:network:domain=green.pod.lan:node=vlsr2:port=7:link=*</remoteLinkId>
    <trafficEngineeringMetric>100</trafficEngineeringMetric>
    <capacity>1000000000</capacity>
    <maximumReservableCapacity>1000000000</maximumReservableCapacity>
    <minimumReservableCapacity>1000000000</minimumReservableCapacity>
    <granularity>100000000</granularity>
    <SwitchingCapabilityDescriptors>
      <switchingcapType>l2sc</switchingcapType>
      <encodingType>ethernet</encodingType>
      <switchingCapabilitySpecificInfo>
        <interfaceMTU>9000</interfaceMTU>
        <vlanRangeAvailability>0,100-200</vlanRangeAvailability>
      </switchingCapabilitySpecificInfo>
    </SwitchingCapabilityDescriptors>
  </link>
</port>
-->
</node>

<!-- BLUE VLSR3 -->
<node id="urn:ogf:network:domain=blue.pod.lan:node=vlsr3">...

```

5) Re-run updateTopology.sh

The updateTopology.sh script must be re-run after enabling the inter-domain link in the XML topology file. Follow the procedure below to run this script:

```

tomcat55@red-narb:~$ cd ~/dcn-software-suite-0.4/idc/tools/updatedbterce/
tomcat55@red-narb:~/dcn-software-suite-0.4/idc/tools/updatedbterce$
./updateTopology.sh http://127.0.0.1:8080/axis2/services/TERCE
DOMAIN: red.pod.lan
NODE: vlsr1
PORT: 3
LINK: *
PORT: 4
LINK: 11.1.2.1
PORT: 5
LINK: 11.1.3.1
NODE: vlsr2
PORT: 4
LINK: 11.1.4.1
PORT: 3
LINK: 11.1.2.2
NODE: vlsr3
PORT: 4

```

```
LINK: 11.1.4.2
PORT: 5
LINK: 11.1.3.2
PORT: 3
LINK: *
PORT: 7
LINK: *
Complete.
tomcat55@red-narb:~/dcn-software-suite-0.4/idc/tools/updatedbterce$
```

Please verify that you see `PORT: 7` under `NODE: vlsr3`, which indicates that the inter-domain link to the green pod has correctly been enabled.

#### 6) Restart tomcat

The final step is to restart Tomcat so all the configurations changes in the previous step will take affect. You may do this with `/usr/local/tomcat/bin/shutdown.sh` and `/usr/local/tomcat/bin/startup.sh`.

### **Step 5: Provision Inter-domain LSPs via WBUI or example Java client**

Repeat the last step in Exercise #2, except that you will have your “source” and “destination” from a different pod. You will need to coordinate with your colleagues in the other domains so that you don’t assign a conflicting IP addresses on the point-to-point/tagged layer 2 links.

Feel free to use either the WBUI or the example Java client to create inter-domain circuits. The example below uses the WBUI, but you could just as easily use the Java client to create these circuits.

Below is an example of provisioning a circuit between red-es1 and yellow-es2. First, the create reservation form:

January 21, 2009  
1:51

Reservation creation form

- Reservations
- Reservation Details
- Create Reservation
- User Profile
- User List
- Add User
- Attributes
- Institutions
- Authorizations
- Authorization Details
- Login/Logout

Required inputs are bordered in green. The source and destination can be topology identifiers, host names, or IP addresses, depending on the layer used. Click on the boxes associated with the start and end dates to bring up a calendar widget. The reservation time slot defaults to now, and now + 4 minutes, respectively, if you leave the dates and times empty.

WARNING: Entering a series of hops in the Path field may alter routing behavior for the selected flow. Hops can be topology identifiers, host names, or IP addresses, depending on the layer used. Note that the path field will expand to the number of lines occurring in the hops list.

Production circuit

Source	<input type="text" value="es1.red.pod.lan"/>
Destination	<input type="text" value="es2.yellow.pod.lan"/>
Path (series of hops)	<input type="text"/>
Bandwidth (Mbps)	<input type="text" value="1,000"/> ( 1-10000 )
Description	<input type="text" value="inter-domain LSP from red-es1 to yellow-es2"/> ( For our records )
Start date	<input type="text" value="1/21/2009"/>
Start time	<input type="text" value="1:51"/>
End date	<input type="text" value="1/21/2009"/>
End time	<input type="text" value="1:55"/>
<input checked="" type="radio"/> Use layer 2 parameters <input type="radio"/> Use layer 3 parameters	
VLAN	<input type="text"/> tag, or range, e.g. 3000-3100
Source Port	<input type="button" value="Tagged"/>
Destination Port	<input type="button" value="Tagged"/>

Next, the Reservation Details screen showing that the reservation was successful and that the circuit is now ACTIVE:

January 21, 2009  
1:53

Reservation details for red.pod.lan-5

Reservations	Reservation Details	Create Reservation	User Profile	User List	Add User	Attributes
Institutions	Authorizations	Authorization Details	Login/Logout			

NEW GRI

---

GRI	red.pod.lan-5
Status	ACTIVE
User	oscars-admin
Description	inter-domain LSP from red-es1 to yellow-es2
Start date	<input type="text" value="1/21/2009"/>
Start time	<input type="text" value="01:52"/>
End date	<input type="text" value="1/21/2009"/>
End time	<input type="text" value="01:56"/>
Created time	2009/01/21 01:52
Bandwidth (Mbps)	1000
Source	urn:ogf:network:domain=red.pod.lan:node=vlsr1:port=3:link=*
Destination	urn:ogf:network:domain=yellow.pod.lan:node=vlsr3:port=3:link=*
Intradomain hops	urn:ogf:network:domain=red.pod.lan:node=vlsr1:port=5:link=11.1.3.1 urn:ogf:network:domain=red.pod.lan:node=vlsr3:port=5:link=11.1.3.2 urn:ogf:network:domain=red.pod.lan:node=vlsr3:port=7:link=*
Interdomain path	urn:ogf:network:domain=red.pod.lan:node=vlsr1:port=3:link=* urn:ogf:network:domain=red.pod.lan:node=vlsr3:port=7:link=* urn:ogf:network:domain=green.pod.lan:node=vlsr1:port=7:link=*
VLAN	139
Tagged	true



If we login to any of the six VLSRs included in this path, we can see the status of the LSP. For example, since this LSP transits green-vlsr1, we see an LSP with the same GRI if we login to the DRAGON CLI on green-vlsr1 and run show lsp:

```
green_vlsr1_pc:~# telnet 127.0.0.1 2611
...
green-vlsr1-pc> show lsp
                        **LSP status summary**

Name          Status      Dir   Source (IP/LSP ID)  Destination (IP/Tunnel ID)
-----
red.pod.lan-5
              In service <=> 192.168.4.4         192.168.4.8
                               100                 100

green-vlsr1-pc> show lsp red.pod.lan-5
Src 192.168.4.4/100, dest 192.168.4.8/100
GRI: 1877564093-2147483649
Generic TSPEC R=gige, B=gige, P=gige, m=100, M=1500
Encoding ethernet, Switching l2sc, G-Pid ethernet
Ingress Local ID Type: tagged group, Value: 100
Egress Local ID Type: tagged group, Value: 100.
E2E LSP VLAN Tag: 100.
Status: In service
```

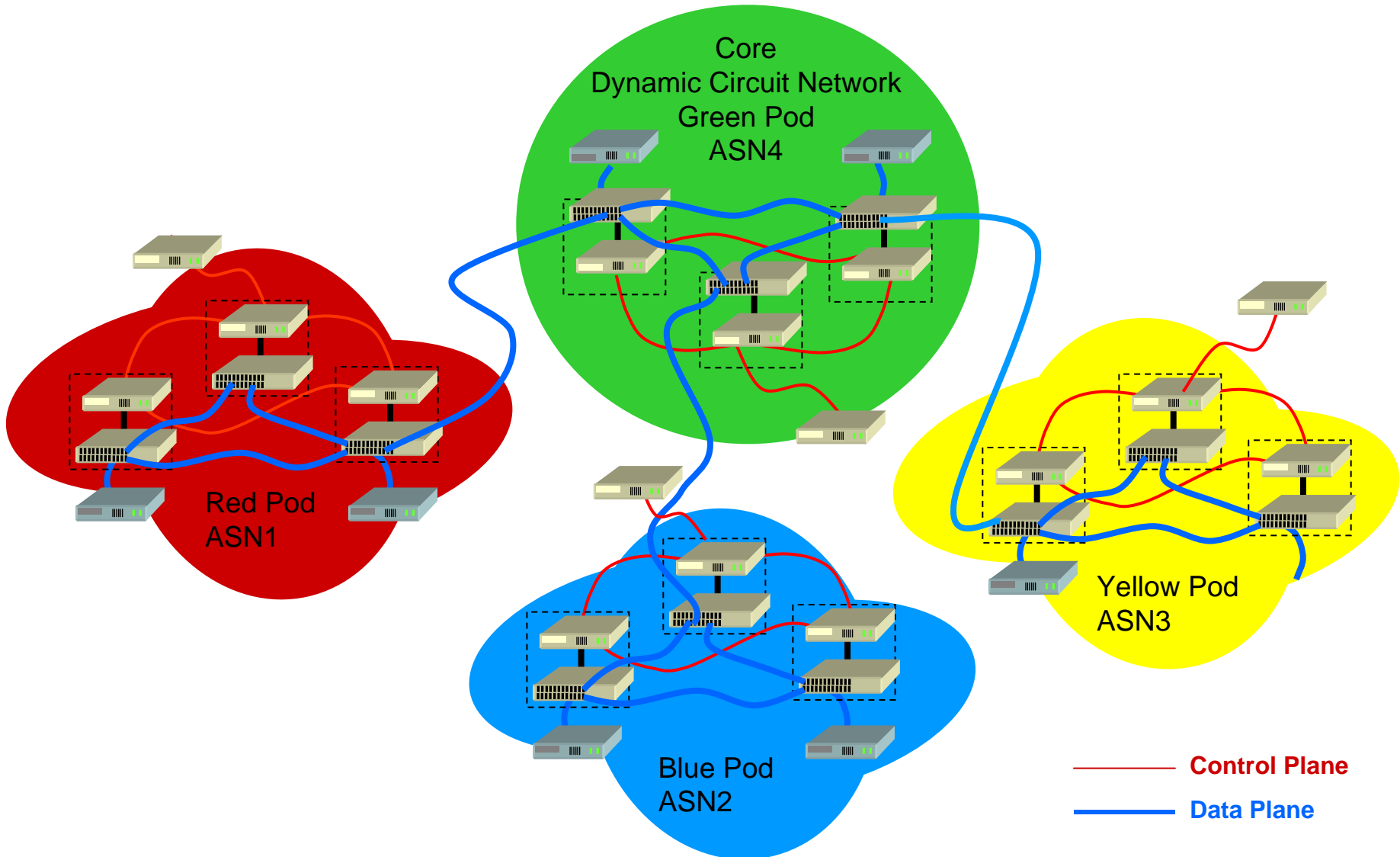


# **Appendix A**

## **Basic Pod Architecture**



# DCN Workshop Architecture

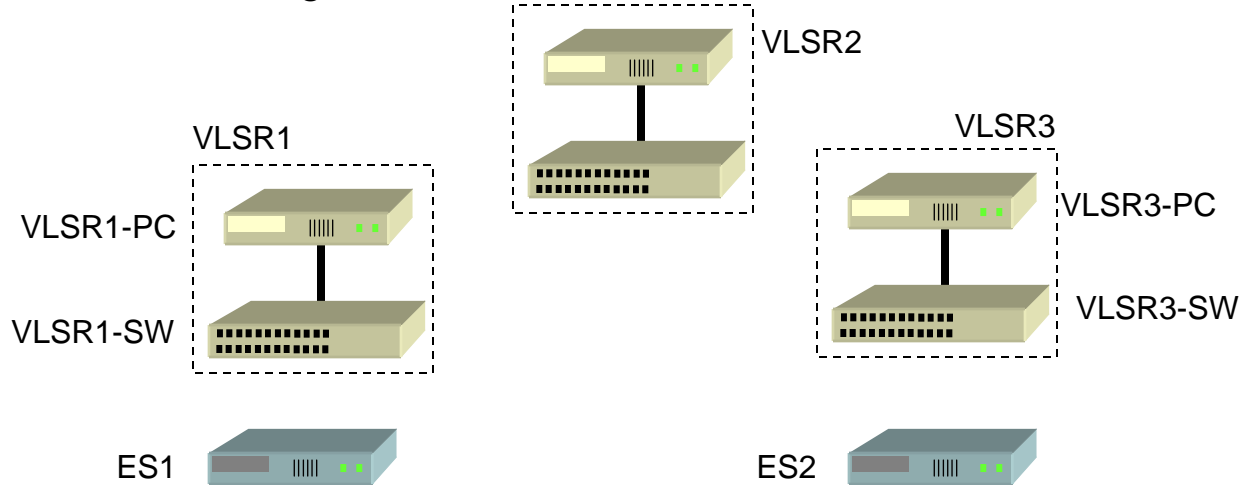


# Pod Network Elements

Inter-Domain Controller – “IDC”  
Network Aware Resource Broker- “NARB”



Virtual Label Switching Router- “VLSR”

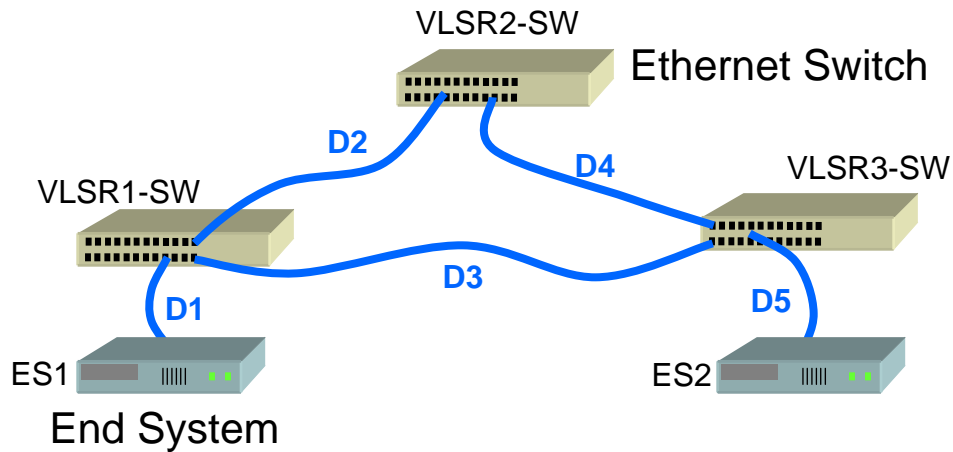


 Control Plane PC (VLSR#-PC, NARB, IDC)

 Data Plane Ethernet Switch (VLSR#-SW)

 End System (ES#)

# Basic Pod Data Plane



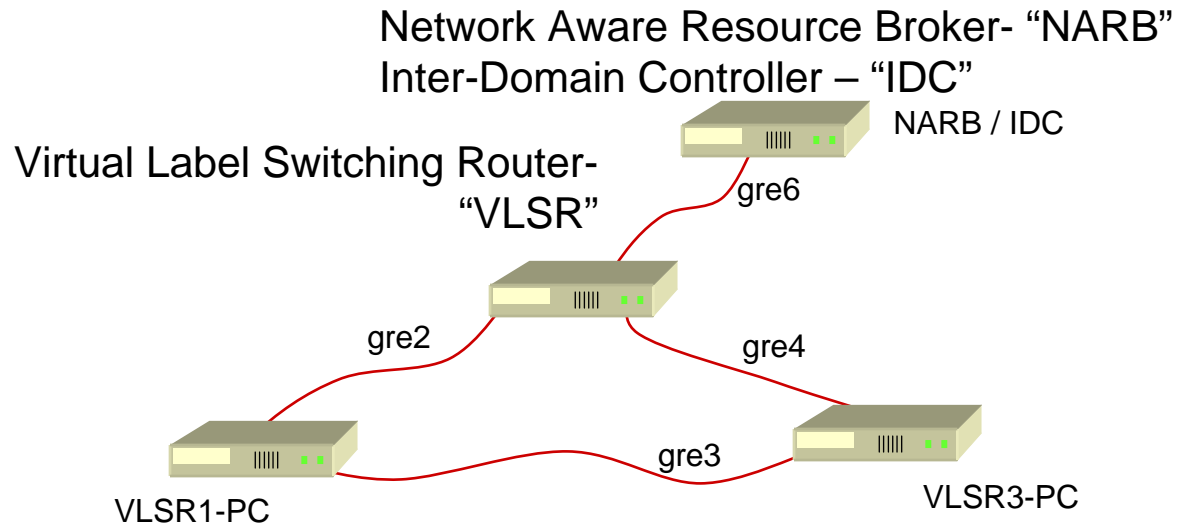
Data Plane via Cat5 Patch Cable

 Data Plane Ethernet Switch (VLSR#-SW)

 End System (ES#)

 Data Plane (D#)

# Basic Pod Control Plane

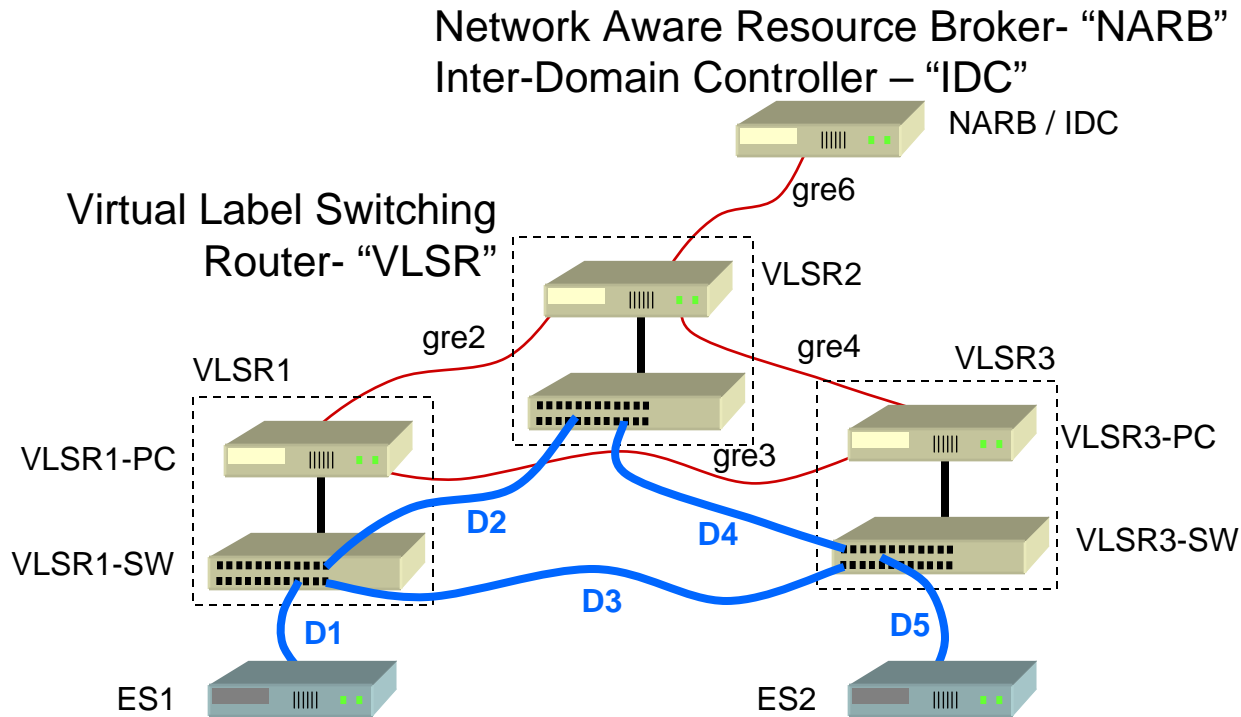


Control Plane PC (VLSR#-PC, NARB, IDC) End System (ES#)




# Pod Network Elements

## Control and Data Planes



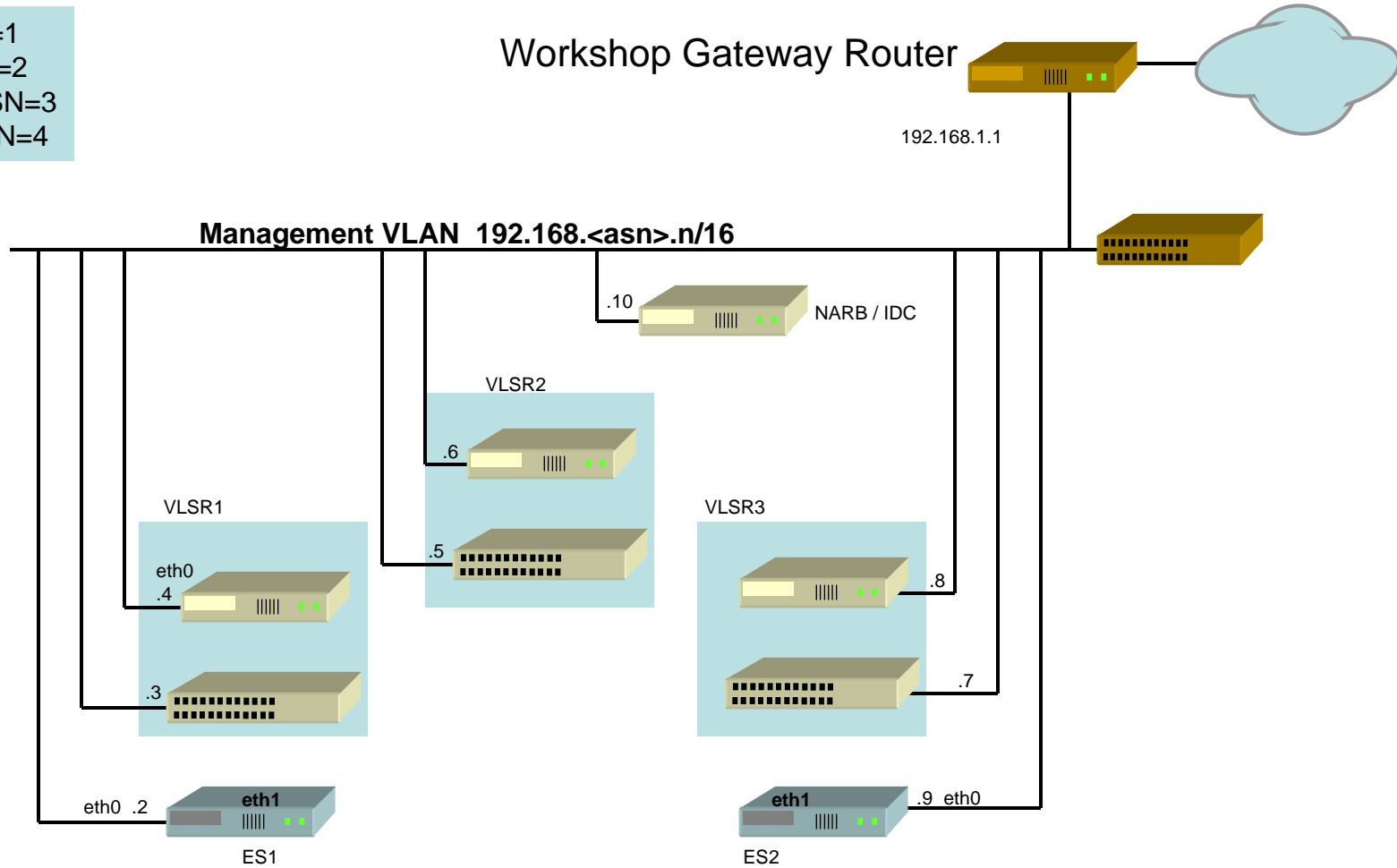
 Control Plane PC (VLSR#-PC, NARB, IDC)

 Data Plane Ethernet Switch (VLSR#-SW)

 End System (ES#)

# Pod Management Addressing













“Red” pod: ASN=1  
“Blue” pod: ASN=2  
“Yellow” pod: ASN=3  
“Green” pod: ASN=4















eth0 - Management Plane Interface and Control Channel (PCs)

eth1 - Data Plane Interfaces (PCs)

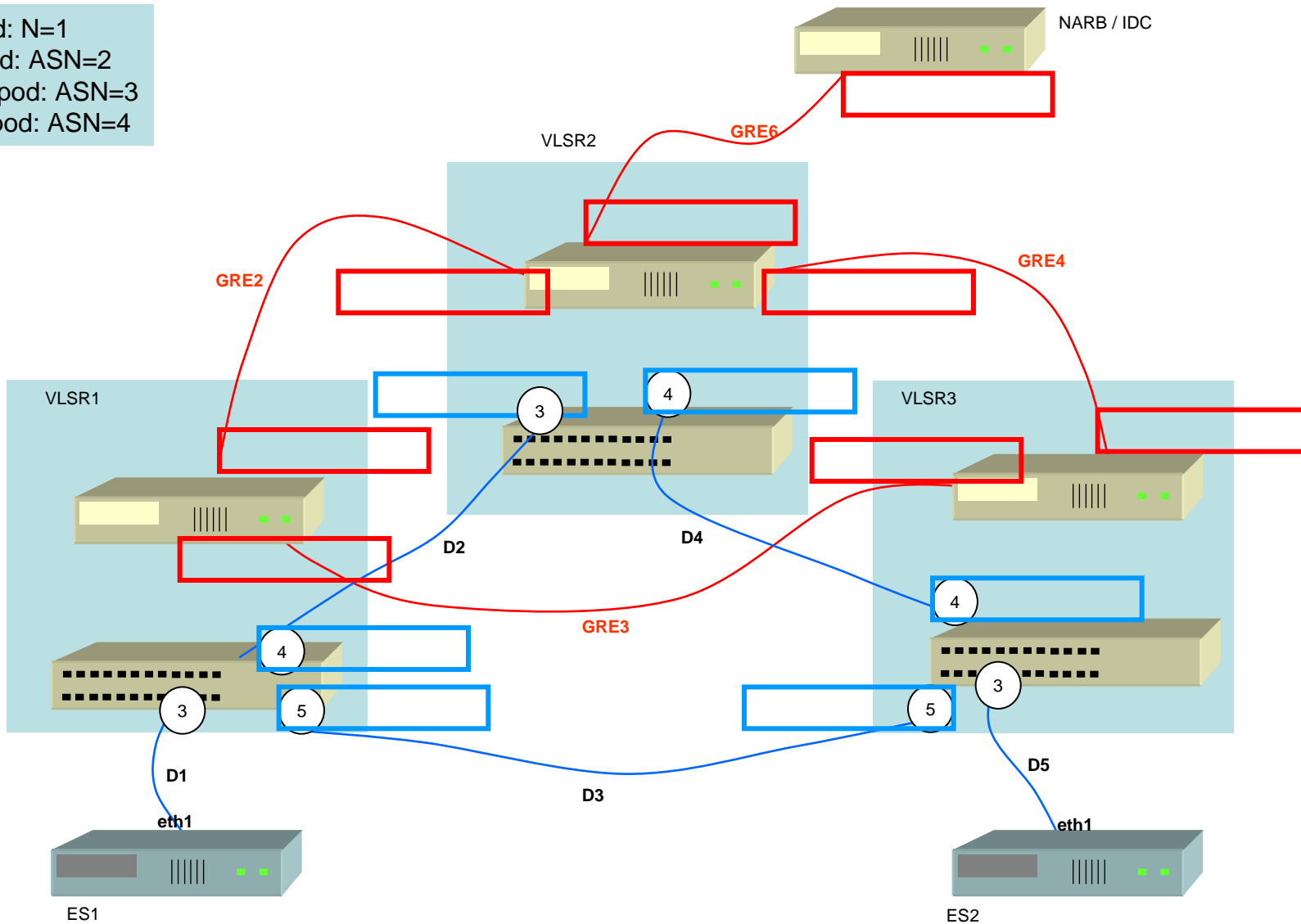
# Rack Layout

GW1
SW1
NARB
VLSR1-SW
VLSR1-PC
VLSR2-SW
VLSR2-PC
VLSR3-SW
VLSR3-PC
ES1
ES2
NARB
VLSR1-SW
VLSR1-PC
VLSR2-SW
VLSR2-PC
VLSR3-SW
VLSR3-PC
ES1
ES2
     
     

GW2
SW2
NARB
VLSR1-PC
VLSR1-SW
VLSR2-PC
VLSR2-SW
VLSR3-PC
VLSR3-SW
ES1
ES2
NARB
VLSR1-PC
VLSR1-SW
VLSR2-PC
VLSR2-SW
VLSR3-PC
VLSR3-SW
ES1
ES2
     
     

# Exercise #1 Data and Control links

“Red” pod: N=1  
“Blue” pod: ASN=2  
“Yellow” pod: ASN=3  
“Green” pod: ASN=4



# Login information

- **Wireless Network:**
  - SSID: DCNworkshop
  - WPA Personal Key: Workshop!
- **Login to all VLSRs, NARBs, and End Systems:**
  - ssh port 22
  - username: user[1-16], password: Workshop!
  - username: root, password: rootme
- **Login to all Dell PowerConnect switches**
  - telnet port 23
  - username: admin, password: admin
- **OSCARS configuration; login to the NARB/IDC machine:**
  - ssh port 22
  - username: tomcat55, password: dragon
- **OSCARS axis2 login:**
  - <https://idc.<color>.pod.lan:8443/axis2/axis2-admin/>
  - username: admin, password: axis2
- **OSCARS web user interface:**
  - <https://idc.<color>.pod.lan:8443/OSCARS/>
  - username: oscars-admin, password: oscars

- **Command Line Interface ports**

- dragond 2611
- ospfd 2604 (intra-domain)
- narb 2626
- rce 2688

```
> telnet localhost 2611
```

```
> password: dragon
```

## **Appendix B**

### **Detailed Pod Architecture and Configuration**





# Workshop Pods

laurel.dragon.maxgigapop.net  
management access



192.168.1.1  
192.168.2.1  
192.168.3.1  
192.168.4.1

Data Plane (TE)  
11.asn.n.n/30

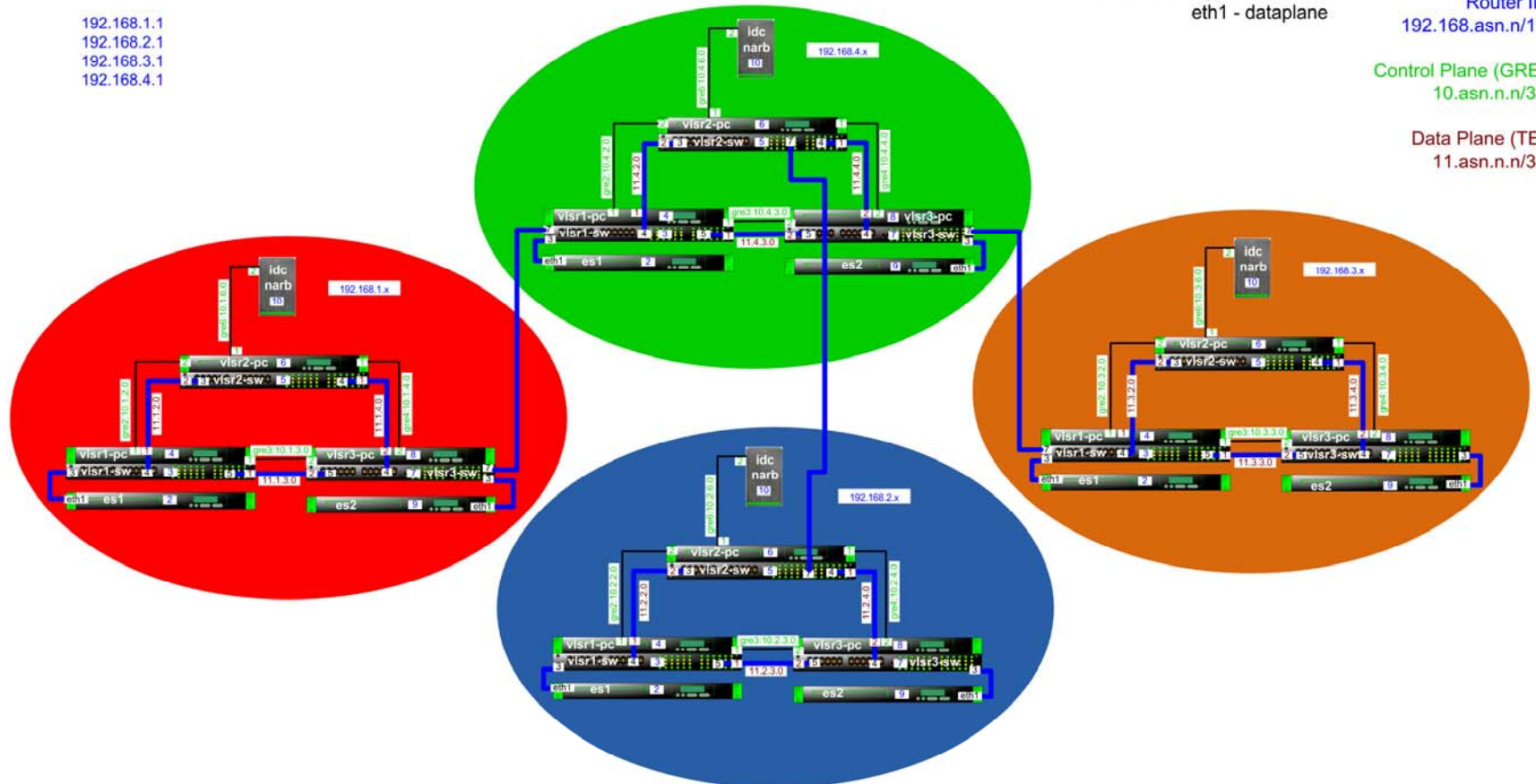
switch port/es interface  
x  
eth0 - management/control  
eth1 - dataplane

Green: ASN 4  
Red: ASN 1  
Blue: ASN 2  
Yellow: ASN 3

Management Network  
Router ID  
192.168.asn.n/16

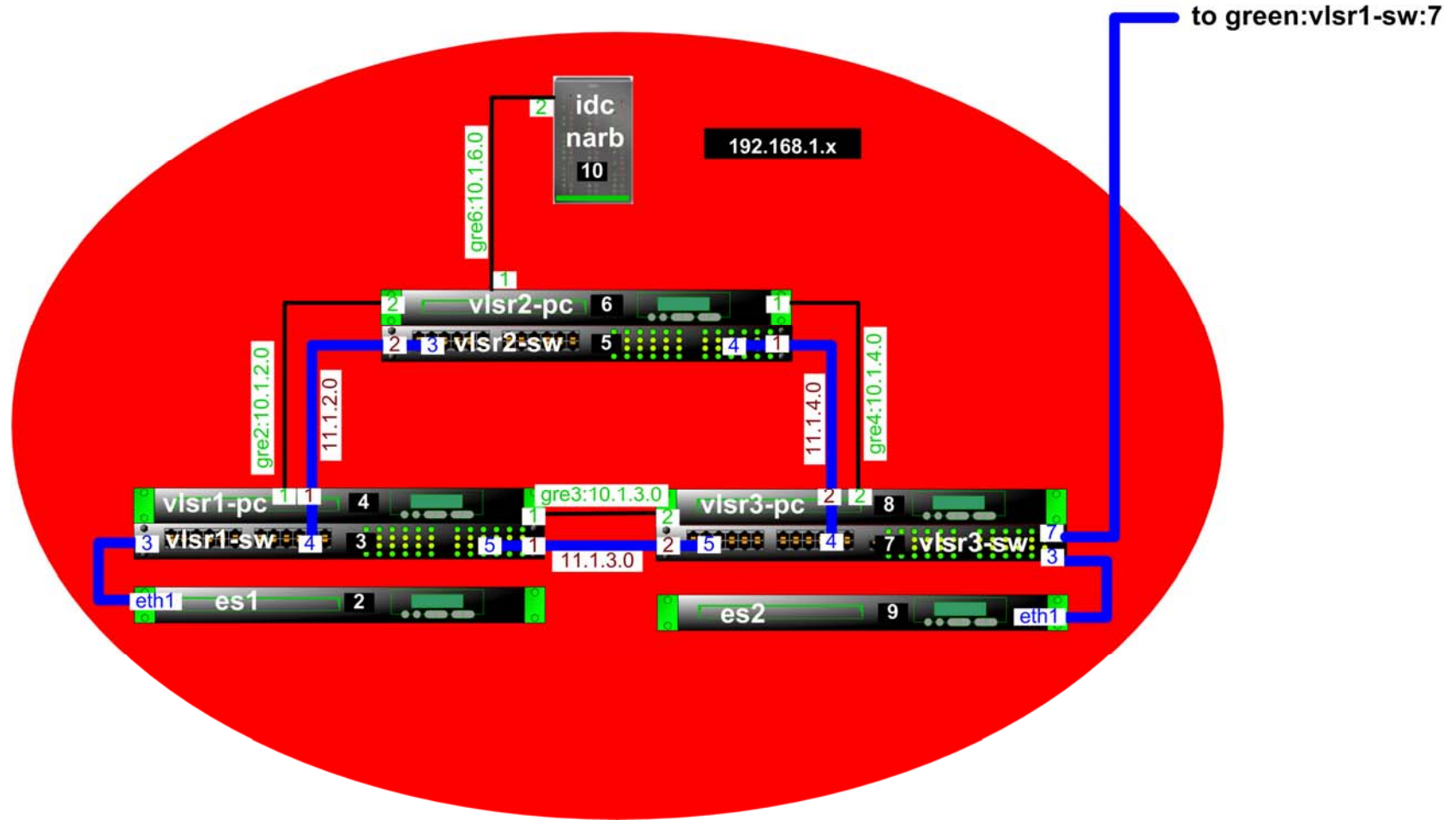
Control Plane (GRE)  
10.asn.n.n/30

Data Plane (TE)  
11.asn.n.n/30

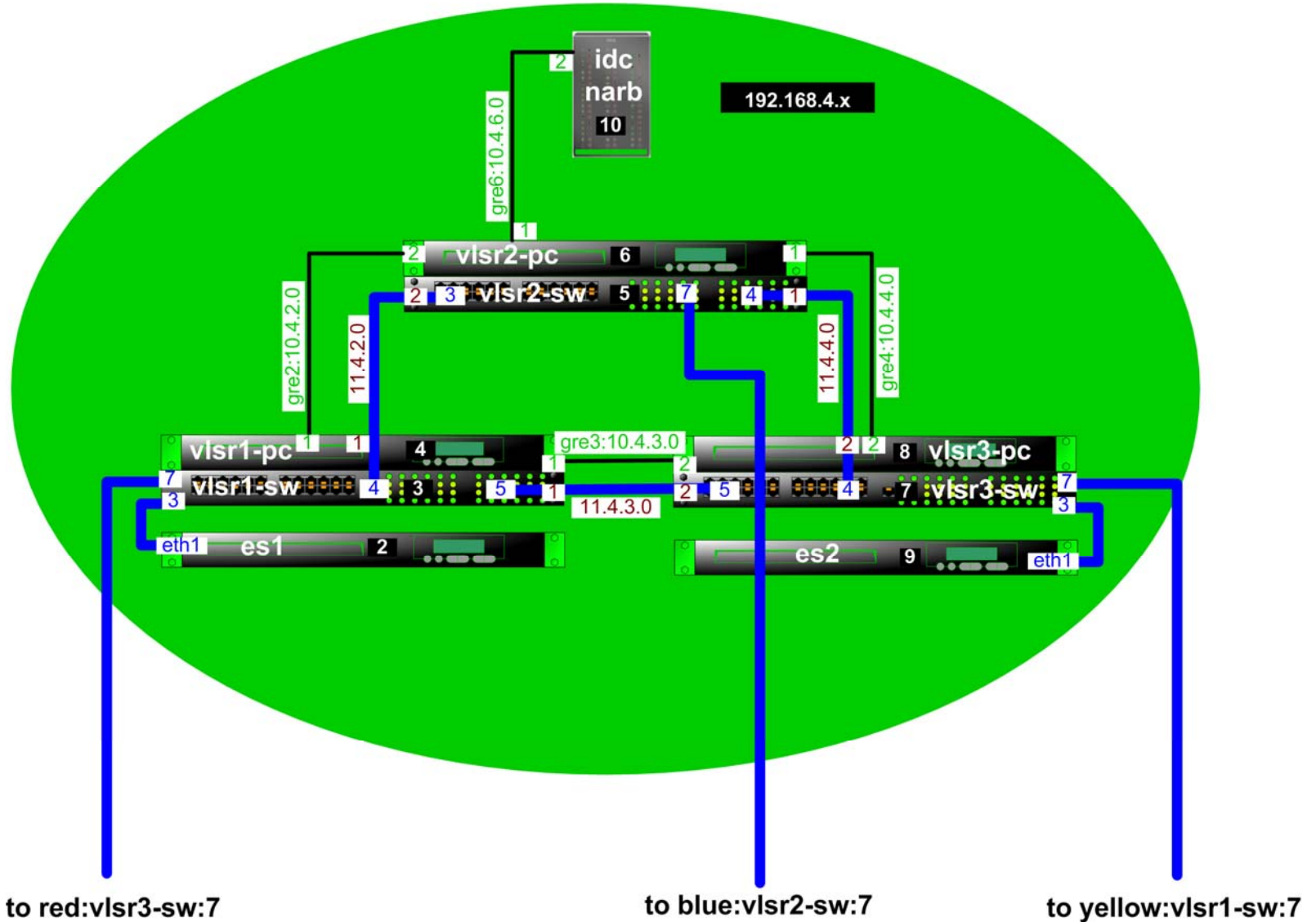




# Red Pod

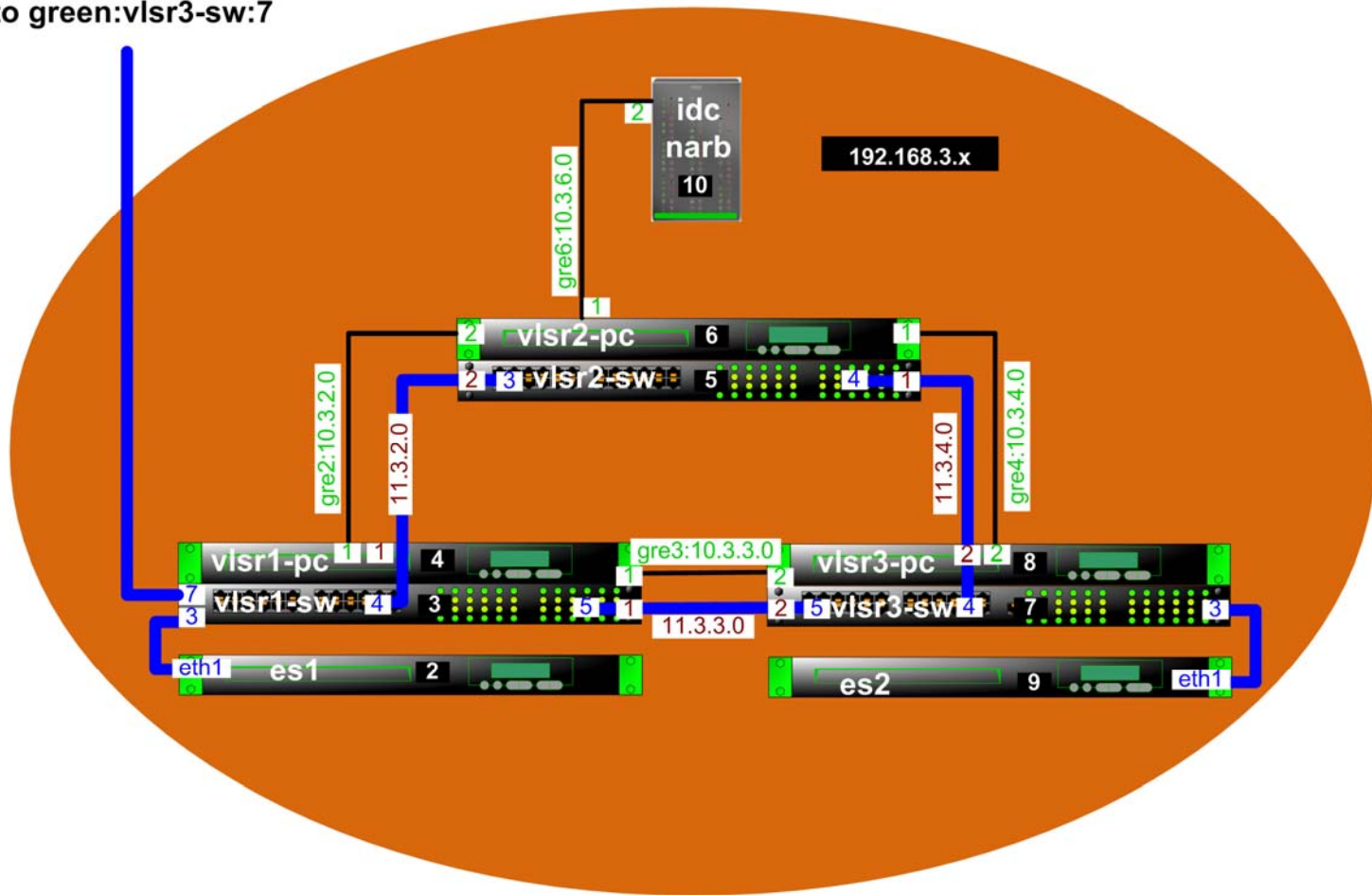


# Green Pod

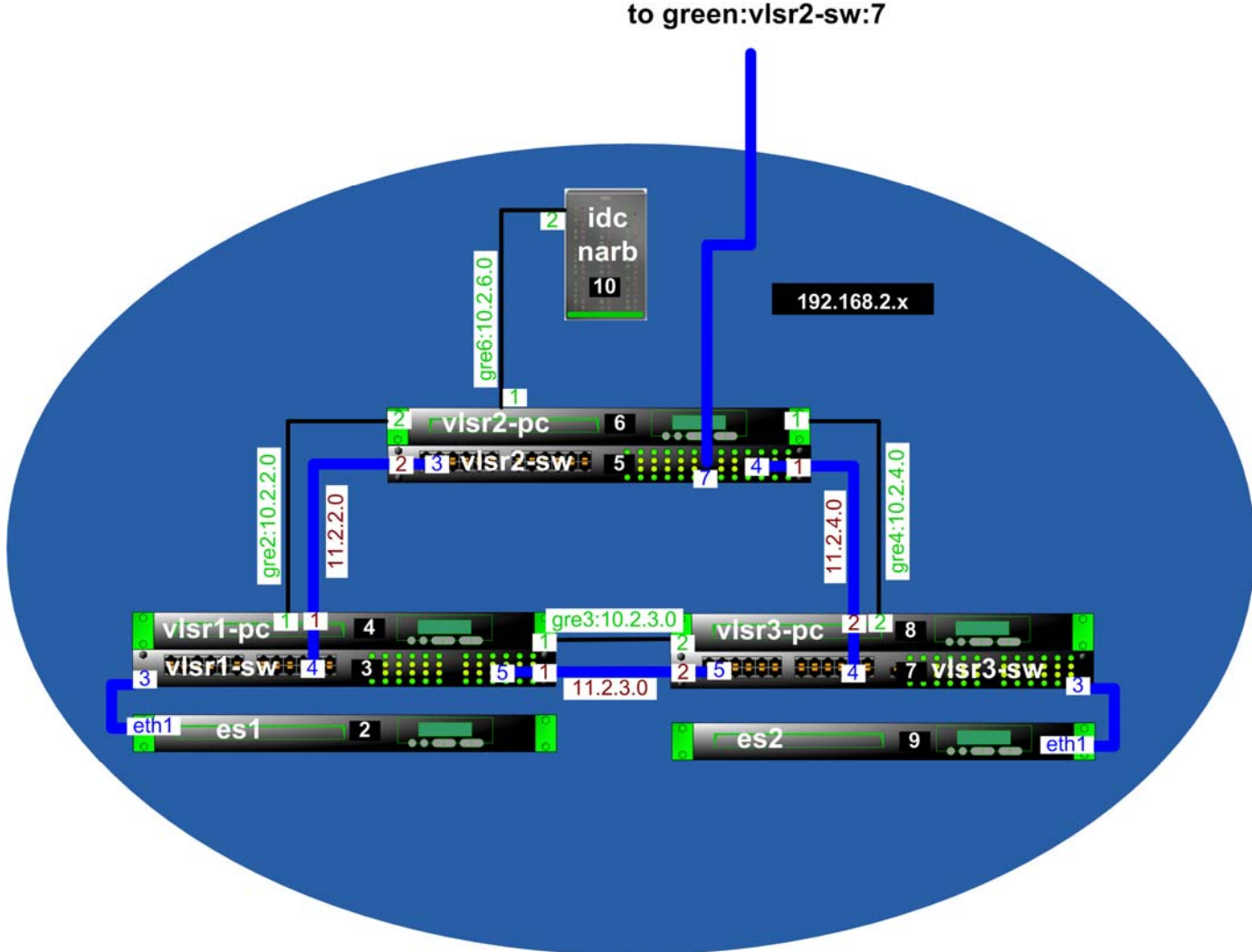


# Yellow Pod

to green:vlsr3-sw:7



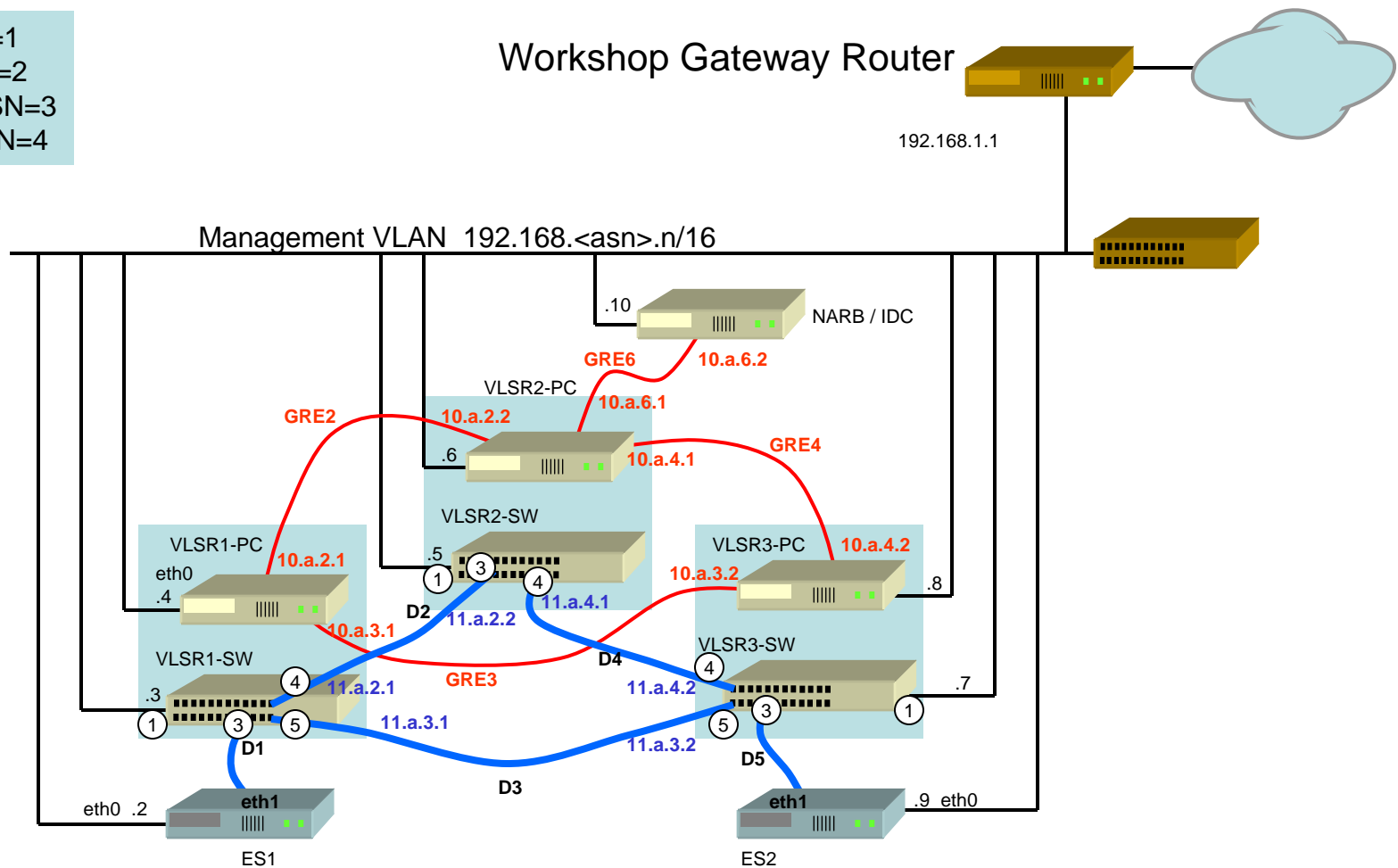
# Blue Pod



# Exercise #1 Intra-Domain Detail

## (Answer Sheet)

"Red" pod: ASN=1  
 "Blue" pod: ASN=2  
 "Yellow" pod: ASN=3  
 "Green" pod: ASN=4



Dynamic Data plane port group = g3-g24  
 Dynamic VLAN range = 100...200

**Management VLAN 192.168.<asn>.n/16**  
**GRE<x> = 10.<asn>.<x>.n / 30**  
**GRE7 = 10.1.7.0 / 30**  
**TEaddr = 11.<asn>.<x>.n / 30**





DRAGON Control Plane Control Plane (GRE tunnels) and DataPlane (TE Links) Configuration									
network device	network component	management address	control channel id	data channel id	Control plane (gre) local address	remote addresses control/management	Data Plane (TE Link) local address	data plane port	Remote network component:control channel
vlsr1	vlsr1-pc	192.168.1.4	red-gre2	D2	10.1.2.1/30	10.1.2.2/192.168.1.6	11.1.2.1 ←	vlsr1-sw:4 ←	vlsr1-pc:red-gre2 ←
			red-gre3	D3	10.1.3.1/30	10.1.3.2/192.168.1.8	11.1.3.1 ←	vlsr1-sw:5 ←	vlsr1-pc:red-gre3 ←
	vlsr1-sw	192.168.1.3							
vlsr2	vlsr2-pc	192.168.1.6	red-gre2	D2	10.1.2.2/30	10.1.2.1/192.168.1.4	11.1.2.2 ←	vlsr2-sw:3 ←	vlsr2-pc:red-gre2 ←
			red-gre4	D4	10.1.4.1/30	10.1.4.2/192.168.1.8	11.1.4.1 ←	vlsr2-sw:4 ←	vlsr2-pc:red-gre4 ←
	vlsr2-sw	192.168.1.5	red-gre6	---	10.1.6.1/30	10.1.6.2/192.16.1.10	---	---	narb:red-gre6 ←
vlsr3	vlsr3-pc	192.168.1.8	red-gre3	D3	10.1.3.2/30	10.1.3.1/192.168.1.4	11.1.3.2 ←	vlsr3-sw:5 ←	vlsr3-pc:red-gre3 ←
			red-gre4	D4	10.1.4.2/30	10.1.4.1/192.168.1.6	11.1.4.2 ←	vlsr3-sw:4 ←	vlsr3-pc:red-gre4 ←
	vlsr3-sw	192.168.1.7							
narb	narb	192.168.1.10	red-gre6	---	10.1.6.2/30	10.1.6.1/192.168.1.6	---	---	vlsr2-pc:red-gre6 ←

VLSR Edge Port Configuration				
Edge Type	edge port	remote connection point	remote management ip	data channel id
es	vlsr1-sw:3	es1:eth1	192.168.1.2	D1
es	vlsr3-sw:3	es2:eth1	192.168.1.9	D2
interdomain	vlsr3-sw:7	green:vlsr1-sw:7	192.168.4.3	---

DRAGON Control Plane Control Plane (GRE tunnels) and DataPlane (TE Links) Configuration									
network device	network component	management address	control channel		Control plane (gre) local address	remote addresses control/management	Data Plane (TE Link) local address	data plane port	Remote network component:control channel
vlsr1	vlsr1-pc	192.168.2.4	blue-gre2		10.2.2.1/30	10.2.2.2/192.168.2.6	11.2.2.1	vlsr1-sw:4	vlsr2-pc:blue-gre2
	vlsr1-sw	192.168.2.3	blue-gre3		10.2.3.1/30	10.2.3.2/192.168.2.8	11.2.3.1	vlsr1-sw:5	vlsr3-pc:blue-gre3
vlsr2	vlsr2-pc	192.168.2.6	blue-gre2		10.2.2.2/30	10.2.2.1/192.168.2.4	11.2.2.2	vlsr2-sw:3	vlsr1-pc:blue-gre2
			blue-gre4		10.2.4.1/30	10.2.4.2/192.168.2.8	11.2.4.1	vlsr2-sw:4	vlsr3-pc:blue-gre4
	vlsr2-sw	192.168.2.5	blue-gre6		10.2.6.1/30	10.2.6.2/192.16.1.10	---	---	narb:blue-gre6
vlsr3	vlsr3-pc	192.168.2.8	blue-gre3		10.2.3.2/30	10.2.3.1/192.168.2.4	11.2.3.2	vlsr3-sw:5	vlsr1-pc:blue-gre3
	vlsr3-sw	192.168.2.7	blue-gre4		10.2.4.2/30	10.2.4.1/192.168.2.6	11.2.4.2	vlsr3-sw:4	vlsr2-pc:blue-gre4
narb	narb	192.168.2.10	blue-gre6		10.2.6.2/30	10.2.6.1/192.168.2.6	---	---	vlsr2-pc:blue-gre6

VLSR Edge Port Configuration				
Edge Type	edge port	remote connection point	remote management ip	data channel id
es	vlsr1-sw:3	es1:eth1	192.168.2.2	D1
es	vlsr3-sw:3	es2:eth1	192.168.2.9	D2
interdomain	vlsr2-sw:7	green:vlsr2-sw:7	192.168.4.5	---

DRAGON Control Plane Control Plane (GRE tunnels) and DataPlane (TE Links) Configuration									
network device	network component	management address	control channel		Control plane (gre) local address	remote addresses control/management	Data Plane (TE Link) local address	data plane port	Remote network component:control channel
vlsr1	vlsr1-pc	192.168.3.4	ylw-gre2		10.3.2.1/30	10.3.2.2/192.168.3.6	11.3.2.1 ←	vlsr1-sw:4 ←	vlsr1-pc:ylw-gre2 ←
			ylw-gre3		10.3.3.1/30	10.3.3.2/192.168.3.8	11.3.3.1 ←	vlsr1-sw:5 ←	vlsr1-pc:ylw-gre3 ←
	vlsr1-sw	192.168.3.3							
vlsr2	vlsr2-pc	192.168.3.6	ylw-gre2		10.3.2.2/30	10.3.2.1/192.168.3.4	11.3.2.2 ←	vlsr2-sw:3 ←	vlsr1-pc:ylw-gre2 ←
			ylw-gre4		10.3.4.1/30	10.3.4.2/192.168.3.8	11.3.4.1 ←	vlsr2-sw:4 ←	vlsr3-pc:ylw-gre4 ←
	vlsr2-sw	192.168.3.5	ylw-gre6		10.3.6.1/30	10.3.6.2/192.16.1.10	---	---	narb:ylw-gre6 ←
vlsr3	vlsr3-pc	192.168.3.8	ylw-gre3		10.3.3.2/30	10.3.3.1/192.168.3.4	11.3.3.2 ←	vlsr3-sw:5 ←	vlsr1-pc:ylw-gre3 ←
			ylw-gre4		10.3.4.2/30	10.3.4.1/192.168.3.6	11.3.4.2 ←	vlsr3-sw:4 ←	vlsr2-pc:ylw-gre4 ←
	vlsr3-sw	192.168.3.7							
narb	narb	192.168.3.10	ylw-gre6		10.3.6.2/30	10.3.6.1/192.168.3.6	---	---	vlsr2-pc:ylw-gre6 ←

VLSR Edge Port Configuration				
Edge Type	edge port	remote connection point	remote management ip	
es	vlsr1-sw:3	es1:eth1	192.168.3.2	D1
es	vlsr3-sw:3	es2:eth1	192.168.3.9	D2
interdomain	vlsr1-sw:7	green:vlsr3-sw:7	192.168.4.7	---

DRAGON Control Plane Control Plane (GRE tunnels) and DataPlane (TE Links) Configuration									
network device	network component	management address	control channel		Control plane (gre) local address	remote addresses control/management	Data Plane (TE Link) local address	data plane port	Remote network component:control channel
vlsr1	vlsr1-pc	192.168.4.4	grn-gre2		10.4.2.1/30	10.4.2.2/192.168.4.6	11.4.2.1 ←	vlsr1-sw:4 ←	vlsr2-pc:grn-gre2 ←
	vlsr1-sw	192.168.4.3	grn-gre3		10.4.3.1/30	10.4.3.2/192.168.4.8	11.4.3.1 ←	vlsr1-sw:5 ←	vlsr3-pc:grn-gre3 ←
vlsr2	vlsr2-pc	192.168.4.6	grn-gre2		10.4.2.2/30	10.4.2.1/192.168.4.4	11.4.2.2 ←	vlsr2-sw:3 ←	vlsr1-pc:grn-gre2 ←
			grn-gre4		10.4.4.1/30	10.4.4.2/192.168.4.8	11.4.4.1 ←	vlsr2-sw:4 ←	vlsr3-pc:grn-gre4 ←
	vlsr2-sw	192.168.4.5	grn-gre6		10.4.6.1/30	10.4.6.2/192.16.1.10	---	---	narb:grn-gre6 ←
vlsr3	vlsr3-pc	192.168.4.8	grn-gre3		10.4.3.2/30	10.4.3.1/192.168.4.4	11.4.3.2 ←	vlsr3-sw:5 ←	vlsr1-pc:grn-gre3 ←
			grn-gre4		10.4.4.2/30	10.4.4.1/192.168.4.6	11.4.4.2 ←	vlsr3-sw:4 ←	vlsr2-pc:grn-gre4 ←
	vlsr3-sw	192.168.4.7							
narb	narb	192.168.4.10	grn-gre6		10.4.6.2/30	10.4.6.1/192.168.4.6	---	---	vlsr2-pc:grn-gre6 ←

VLSR Edge Port Configuration				
Edge Type	edge port	remote connection point	remote management ip	
es	vlsr1-sw:3	es1:eth1	192.168.4.2	D1
es	vlsr3-sw:3	es2:eth1	192.168.4.9	D2
interdomain	vlsr1-sw:7	red:vlsr1-sw:7	192.168.1.7	---
interdomain	vlsr2-sw:7	blue:vlsr2-sw:7	192.168.2.5	---
interdomain	vlsr2-sw:7	yellow:vlsr1-sw:7	192.168.3.3	---

## **Appendix C**

### **DCN Software Suite v0.4: OSCARS Inter-Domain Controller (IDC) Installation Guide**

For the latest version of the DCN Software Suite,  
visit the following URL:

<https://wiki.internet2.edu/confluence/display/DCNSS>





[www.internet2.edu](http://www.internet2.edu)

# DCN Software Suite v0.4.0: OSCARS Inter-Domain Controller (IDC) Installation Guide





# Table of Contents

<a href="#">1 Overview.....</a>	<a href="#">1</a>
<a href="#">1.1 About this Document.....</a>	<a href="#">1</a>
<a href="#">1.2 Hardware and Software Requirements.....</a>	<a href="#">1</a>
<a href="#">1.2.1 System Requirements.....</a>	<a href="#">1</a>
<a href="#">1.2.2 Network Requirements.....</a>	<a href="#">1</a>
<a href="#">1.2.3 Firewall Requirements.....</a>	<a href="#">1</a>
<a href="#">1.2.4 Third-Party Library and Package Requirements.....</a>	<a href="#">1</a>
<a href="#">1.3 Downloading the IDC software .....</a>	<a href="#">2</a>
<a href="#">2 Preparing your Environment.....</a>	<a href="#">2</a>
<a href="#">2.1 MySQL.....</a>	<a href="#">3</a>
<a href="#">2.1.1 Install Option 1: Manual Installation.....</a>	<a href="#">3</a>
<a href="#">2.1.2 Install Option 2: Automatic Installation with a Package Manager.....</a>	<a href="#">3</a>
<a href="#">2.2 Java Development Kit (JDK).....</a>	<a href="#">3</a>
<a href="#">2.2.1 Do I already have the right version of Java?.....</a>	<a href="#">3</a>
<a href="#">2.2.2 Download and Installation.....</a>	<a href="#">4</a>
<a href="#">2.2.3 Setting the JAVA_HOME Environment Variable.....</a>	<a href="#">4</a>
<a href="#">2.2.4 Optional: Adding JAVA_HOME/bin To Your PATH Variable.....</a>	<a href="#">4</a>
<a href="#">2.3 Tomcat.....</a>	<a href="#">5</a>
<a href="#">2.3.1 Download and Installation.....</a>	<a href="#">5</a>
<a href="#">2.3.2 Setting the CATALINA_HOME Environment Variable.....</a>	<a href="#">5</a>
<a href="#">2.3.3 Starting/Stopping the Tomcat Server.....</a>	<a href="#">6</a>
<a href="#">2.3.4 Verifying a Successful Installation.....</a>	<a href="#">6</a>
<a href="#">2.3.5 Configuring SSL.....</a>	<a href="#">6</a>

2.4 Ant.....	6
2.4.1 Download and Installation.....	6
2.4.2 Setting the ANT_HOME Environment Variable.....	7
2.4.3 Adding ANT_HOME/bin to Your PATH Variable.....	7
2.5 SMTP Server.....	7
3 Installing the Inter-Domain Controller (IDC) Software.....	7
3.1 Installing the IDC for the First Time.....	8
3.2 Upgrading an Existing IDC from Version 0.3.X.....	10
3.3 Installing the TERCE.....	17
3.4 Creating the First User Account.....	18
3.5 Changing Keystore Passwords.....	19
3.6 Verifying the Web Service API Installation.....	19
3.7 Verifying the Web User Interface (WBUI) Installation.....	20
4 Creating and Managing User Accounts.....	20
4.1 Adding Users to the Database.....	20
4.1.1 Creating New User Accounts.....	20
4.1.2 Modifying Users.....	21
4.1.3 Deleting Users.....	21
4.2 Managing X.509 Certificates from Users and Other IDCs.....	21
4.2.1 Trusted Certificate Authorities.....	22
4.2.2 Associating an X.509 Certificate with a User Account.....	24
5 Describing Your Network Topology.....	25
5.1 Generating an XML Topology Description.....	25
5.1.1 Example XML files.....	25
5.1.2 Domains, Nodes, Ports and Links.....	26

5.1.3 Fully-Qualified Identifiers.....	26
5.1.4 <topology> Element.....	27
5.1.5 <domain> Element.....	27
5.1.6 <node> element.....	28
5.1.7 <port> Element.....	28
5.1.8 <link> Element.....	29
5.1.9 <switchingCapabilityDescriptors> Element.....	30
5.1.10 <switchingCapabilitySpecificInfo> Element.....	30
5.2 Configuring the terce-ws.properties file.....	31
5.3 Populating the Scheduling Database.....	31
5.3.1 Defining Your Local Domain.....	31
5.3.2 Run updateTopology.sh.....	32
5.4 Registering with a Topology Service.....	32
6 Verifying that You Can Build Local Circuits.....	33
7 Inter-Domain Configuration.....	33
7.1 Generating an IDC Certificate for Sending Inter-Domain Requests.....	34
7.1.1 Creating a Certificate and Private Key.....	34
7.1.2 Getting the IDC Certificate Signed.....	36
7.2 Making your IDC Aware of Other Domains.....	37
7.3 Activating Notifications.....	39
7.4 Trusting SSL Certificates of Other IDCs Running HTTPS.....	41
8 Appendix A: oscars.properties.....	42
8.1.1 General.....	42
8.1.2 MySQL Database Properties.....	42
8.1.3 Authentication, Authorization, and Accounting (AAA) Properties.....	43

8.1.4 Topology Exchange and Pathfinding Properties.....	44
8.1.5 Path Setup Properties.....	45
8.1.6 Notifications.....	47
8.1.7 External Services.....	49
8.1.8 Micellaneous Properties.....	50
9 Appendix B: Using Hostnames Instead of URNs in Requests.....	52
10 Appendix C: Manually Installing Axis2.....	52
10.1.1 Download and Installation.....	52
10.1.2 Setting the AXIS2_HOME Environment Variable.....	53
10.1.3 Verifying a Successful Installation.....	53
10.2 Manually Installing Rampart.....	53
10.2.1 Download and Installation.....	53
10.2.2 Verifying a Successful Installation.....	54
11 Appendix D: Becoming a Certificate Authority (CA).....	54
11.1.1 Generating Your Own CA Certificate.....	54
11.1.2 Signing User ‘Certificate Signing Requests’ (CSRs).....	55
12 Appendix E: Static Path Calculation.....	55
12.1 Creating a Static List of Local Paths.....	55
12.2 Building Your Inter-Domain Routing Table.....	56
12.2.1 Populating the <remoteLinkId> fields in tedb-intra.xml and tedb-inter.xml.....	57
12.2.2 Defining a Default Route.....	57
12.2.3 Defining a Path with Only an Egress.....	58
12.2.4 Defining a Path with Multiple Hops.....	58





# 1 Overview

## 1.1 About this Document

This document is intended to be a guide for installing the OSCARS Inter-Domain Controller (IDC) as part of the DCN Software Suite. It specifically targets those interested in installing an IDC on a network running the DRAGON software (also included in the DCN Software Suite). The document assumes basic familiarity with DRAGON and experience with a Unix-like operating system. It does not assume experience with XML or building Java software but such experience may be useful.

## 1.2 Hardware and Software Requirements

### 1.2.1 System Requirements

The OSCARS IDC software requires a single PC that will act as a web server for processing requests. Most modern PCs should be suitable for running the software. The following specifications are the minimum requirements for most installations:

- 1Ghz Processor (preferably X86 architecture), 1GB memory
- Linux/Unix Operating System
- Basic Internet connectivity
- System clock running the Network Time Protocol (NTP)

Requirements may be greater for systems running the OSCARS IDC concurrently on the same machine as other components of the DCN Software Suite.

### 1.2.2 Network Requirements

A network running the DRAGON control plane software is required for this installation. See the DRAGON documentation for more information. See **Section 1.3** for information on obtaining DRAGON.

### 1.2.3 Firewall Requirements

The IDC runs on port 8080 and port 8443 by default.

### 1.2.4 Third-Party Library and Package Requirements

Installing and running the OSCARS IDC requires the following software packages:

Name	Supported Version	Download Location
MySQL	4.1+	<a href="http://dev.mysql.com/downloads/mysql/">http://dev.mysql.com/downloads/mysql/</a>
Java Development Kit (JDK)	5.0	<a href="http://java.sun.com/javase/downloads/index_jdk5.jsp">http://java.sun.com/javase/downloads/index_jdk5.jsp</a>
Tomcat	5.5	<a href="http://tomcat.apache.org/download-55.cgi">http://tomcat.apache.org/download-55.cgi</a>
Axis2	1.4.1	<a href="http://ws.apache.org/axis2/download/1_4_1/download.cgi">http://ws.apache.org/axis2/download/1_4_1/download.cgi</a>
Rampart	1.4.1 SNAPSHOT	<a href="https://wiki.internet2.edu/confluence/download/attachments/19074/rampart-SNAPSHOT.tar.gz">https://wiki.internet2.edu/confluence/download/attachments/19074/rampart-SNAPSHOT.tar.gz</a>
Ant	1.7	<a href="http://ant.apache.org/bindownload.cgi">http://ant.apache.org/bindownload.cgi</a>

### 1.3 Downloading the IDC software

The IDC software is part of the DCN Software Suite. It can be downloaded at:

- <https://wiki.internet2.edu/confluence/display/DCNSS>

After downloading the DCN software suite, you may unpack it with the following commands:

```
% gunzip dcn-software-suite-0.4.tar.gz
% tar -xvf dcn-software-suite-0.4.tar
```

This will create a directory called `dcn-software-suite-0.4`. The IDC software, called OSCARS, is located in the subdirectory `dcn-software-suite-0.4/idc`. The TERCE software is located in `dcn-software-suite-0.4/terce`. This document describes the installation of both the IDC and TERCE. The DRAGON software is located in the subdirectory `dcn-software-suite-0.4/dragon`. DRAGON software installation instructions are located in `dcn-software-suite-0.4/dragon/docs/DRAGON-INSTALL-0.3.pdf`. The remainder of this document will focus on the IDC installation.

## 2 Preparing your Environment

This section details how to install and configure prerequisite software on the machine that will be running the IDC. The following prerequisite steps are detailed:

1. Install MySQL
2. Install the Java Development Kit and set the `JAVA_HOME` environment variable
3. Install the Tomcat web server and set the `CATALINA_HOME` environment variable



4. Install Ant and add it to your PATH environment variable

In addition the Axis2 and Rampart libraries from Apache must be installed. This will be done automatically by the OSCARS installation script as described in Section 3. If you would like to perform this installation manually please see section .

## 2.1 MySQL

MySQL is the database used to maintain user accounts and track reservations. You may install MySQL in one of two ways: manually, by installing a package downloaded from the MySQL web site OR automatically, using your operating system's package manager:

### 2.1.1 Install Option 1: Manual Installation

Download the MySQL package from the MySQL web site at:

- <http://dev.mysql.com/downloads/mysql>

Installing MySQL in this manner is beyond the scope of this document but (English) installation instructions may be found at:

- <http://dev.mysql.com/doc>

### 2.1.2 Install Option 2: Automatic Installation with a Package Manager

Download and install MySQL through a package manager if your operating system runs such a service. A few common package managers are up2date (RedHat), apt-get (Debian), and yum. You may install MySQL using a command such as:

```
% up2date mysql-server
```

Consult specific package managers for the exact command and package name.

## 2.2 Java Development Kit (JDK)

Java is the programming language in which the OSCARS IDC software was created and provides the environment in which it runs. In addition to running the software, the JDK also contains utilities required for compiling the source code and generating user certificates. This section details installation and configuration related to this package.

### 2.2.1 Do I already have the right version of Java?

Many systems come pre-installed with Java. To install the IDC, your system must not only have Java Runtime Environment (JRE) version 5 but also the various compilers and utilities. To verify that you have the necessary Java environment, issue the following command:

```
% javac -version
```

If the first line of output reads `javac 1.5.0_X`, you should not need to install the Java Development Kit and may skip to section **2.2.3 Setting the JAVA\_HOME Environment Variable**. If you get “command not found” or the version number is less than 1.5, you may need to install JDK 5.0 and should proceed to **2.2.2 Download and Installation**.

**NOTE: If you are not running the SUN distribution of Java you may encounter issues. The GNU and IBM versions of Java are not fully tested and some users have reported problems. It is recommended you run the SUN distribution of Java.**

## 2.2.2 Download and Installation

You may download JDK 5.0 from Sun’s web site at:

- [http://java.sun.com/javase/downloads/index\\_jdk5.jsp](http://java.sun.com/javase/downloads/index_jdk5.jsp)

It is recommended that you download the latest update of **JDK Version 5**. Choose the package most suitable for your operating system.

Once downloaded, unpack the file; this should create a new folder named something similar to “jdk1.5.0\_X”. The final step of installation is to move this folder to an easily accessible place. We recommend renaming the folder to `java5` in `/usr/local` with the following command:

```
% sudo mv jdk1.5.0_X /usr/local/java5
```

The location may be anywhere you choose – just make sure you note the location as it is required for setting the `JAVA_HOME` environment variable in the next section.

## 2.2.3 Setting the JAVA\_HOME Environment Variable

Once Java is installed, you need to set the `JAVA_HOME` environment variable with its location. This variable is required by the Tomcat web server (see section **2.3 Tomcat**) to run. To set this environment variable, issue these commands:

```
% JAVA_HOME=/usr/local/java5
% export JAVA_HOME
```

You may permanently set this variable (recommended) by adding the above commands to the profile file in your home directory (i.e. `.bash_profile` or `.profile`).

## 2.2.4 Optional: Adding JAVA\_HOME/bin To Your PATH Variable

This step is optional but may make issuing commands easier in later steps. You should add the folder `JAVA_HOME/bin` to your `PATH` environment variable so that you can easily access the

`keytool` command for issuing certificates. To update your `PATH` variable, issue these commands:

```
% PATH=$PATH:$JAVA_HOME/bin
% export PATH
```

You may permanently set this variable (recommended) by adding the above commands to the profile file in your home directory (i.e. `.bash_profile` or `.profile`).

## 2.3 Tomcat

Tomcat is a Java-based application container in which the OSCARS IDC software runs. This section details installation and basic configuration of Tomcat.

### 2.3.1 Download and Installation

You may download Tomcat from the project's web site at:

- <http://tomcat.apache.org/download-55.cgi>

It is recommended you download **Tomcat Version 5.5**.

**NOTE: Tomcat 6.0 is NOT currently supported by the software.**

Once downloaded, unpack the downloaded file; this should create a new folder named something similar to “`apache-tomcat-5.5.X`”. The final step of installation is to move this folder to an easily accessible place. We recommend renaming the folder to `tomcat` in `/usr/local` with the following command:

```
% sudo mv apache-tomcat-5.5.X /usr/local/tomcat
```

The location may be anywhere you choose – just make sure you note the location as it is required for setting the `CATALINA_HOME` environment variable in the next section.

**NOTE: It is NOT RECOMMENDED that you download Tomcat with a package manager such as `up2date`, `yum`, or `apt-get`. Many users have reported difficulty with this method. If you install Tomcat using this method be aware that some of the environment variables and other settings may vary from what is contained within this document.**

### 2.3.2 Setting the `CATALINA_HOME` Environment Variable

Once Tomcat is installed, you need to set the `CATALINA_HOME` environment variable with its location. This variable is required by the Tomcat web server to run. To set this environment variable, issue these commands:

```
% CATALINA_HOME=/usr/local/tomcat
% export CATALINA_HOME
```

You may permanently set this variable (recommended) by adding the above commands to the profile file in your home directory (i.e. `.bash_profile` or `.profile`).

### 2.3.3 Starting/Stopping the Tomcat Server

You may start Tomcat with the following command:

```
% $CATALINA_HOME/bin/startup.sh
```

You shutdown the Tomcat server with the following command:

```
% $CATALINA_HOME/bin/shutdown.sh
```

### 2.3.4 Verifying a Successful Installation

To verify installation was successful, startup the Tomcat server with the following command:

```
% $CATALINA_HOME/bin/startup.sh
```

After starting the server, point a web browser to port 8080 of the machine on which you installed Tomcat with the following URL:

- <http://your-machine-name:8080>

If installation was successful, a web page will load with the Tomcat logo and a message that reads “If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!”

### 2.3.5 Configuring SSL

You may configure Tomcat to use SSL so that all requests and responses to the server are encrypted. This is not required but highly recommended. Information on this process can be found at:

- <http://tomcat.apache.org/tomcat-5.5-doc/ssl-howto.html>

## 2.4 Ant

Ant is a tool that is used to build the IDC code and deploy various configuration files (think of it as “make” for Java). This section details how to install and configure Ant.

### 2.4.1 Download and Installation

Download Ant from the project’s web site at:

- <http://ant.apache.org/bindownload.cgi>

Most IDC testing has been done with **Version 1.7**. Unpack and install Ant with the following commands:

```
% unzip apache-ant-1.7.0-bin.zip
% sudo mv apache-ant-1.7.0 /usr/local/ant
```

You are not required to install the downloaded folder in /usr/local/ant but you should note where it is installed as this information is needed in later steps.

## 2.4.2 Setting the ANT\_HOME Environment Variable

Once Ant is installed, you need to set the ANT\_HOME environment variable with Ant's location. To set this environment variable, issue these commands:

```
% ANT_HOME=/usr/local/ant
% export ANT_HOME
```

You may permanently set this variable (recommended) by adding the above commands to the profile file in your home directory (i.e. .bash\_profile or .profile).

## 2.4.3 Adding ANT\_HOME/bin to Your PATH Variable

It is recommended you add the Ant bin directory to your PATH environment variable. This will allow ant command-line tools to be found when you type-in the command name. To set this environment variable, issue these commands:

```
% PATH=$PATH:$ANT_HOME/bin
% export PATH
```

You may permanently set this variable (recommended) by adding the above commands to the profile file in your home directory (i.e. .bash\_profile or .profile).

## 2.5 SMTP Server

You may **optionally** install an SMTP server on the same machine as the IDC and it will send email notification of circuit activity. A standard installation of `sendmail` should be suitable for the IDC's purposes. The IDC sends mail by transmitting SMTP packets to localhost so it is not dependent on the underlying software. Installing such software is beyond the scope of this document.

## 3 Installing the Inter-Domain Controller (IDC) Software

This section details how to install the OSCARS IDC Software. It assumes you have installed all the prerequisites as described in the previous section, Preparing your Environment. This section will cover installing an IDC for the first time, upgrading from a previous version of the IDC, and installing the TERCE component.

### 3.1 Installing the IDC for the First Time

This section describes how to install OSCARS on a machine that does not have another version of the software already running. **If you have a previous version of the OSCARS IDC installed that you wish to upgrade, please proceed to the next section of this document.** The steps in this section will install both the OSCARS web page interface for human users and the web service interface for applications. To install OSCARS for the first time run the `do_build.sh` script followed by `do_install.sh`. These two scripts will compile the code, build the necessary MySQL database tables, and copy configuration files to the appropriate locations. Below is a detailed description of their use.

First run `do_build.sh` and you should see the following:

```
$ ./do_build.sh

--- Checking prerequisites...
    We seem to be in the correct directory
    Found ant
    Environment variable CATALINA_HOME is set to
    /usr/local/tomcat
    Environment variable DOMAIN_HOME is not set. Continuing
    without it.
    Found jta.jar under lib
    Axis2 library not found.
    Rampart library not found.
```

If you do not see the above output verify that you have installed all the prerequisites described in **2 Preparing your Environment**. You should answer ‘y’ to the next two questions and proceed as shown below:

```
- Axis2 installation not detected. Should I build it for you
y/n? y
    OK, will build Axis2 for you.

- Axis2 is not deployed. Should I do this for you y/n? y
    OK, will deploy Axis2 for you.
--- Downloading Axis2...
```

You should then see output of Axis2 and the module Rampart downloading. The script will then restart your Tomcat server. The output for the Axis2/Rampart download and the Tomcat restart is not shown in this document as it is rather lengthy. You should know that if your Tomcat server is not running you may see a harmless “Connection refused” exception. This only displays because the script first tries to shutdown the Tomcat server, which it can only do if Tomcat is running. This exception should not affect you installation so please ignore it if it appears. After restarting Tomcat you should see the following:

```
--- Your kit looks good.
```

- Input the hostname for this IDC. Leave blank for "your-server.com":

If the host name displayed is not correct for your machine then enter the correct one, otherwise you may just hit the return key. You will then be asked if you would like to setup your MySQL database. You should answer 'y' to the questions and provide information for an account capable of creating and granting privileges on databases as shown below:

- Install databases y/n? **y**  
OK, will install databases.  
Found mysql client at /usr/bin/mysql
- Input the MySQL server hostname. Leave blank for localhost:  
Using localhost .
- Input a privileged MySQL username on that host. Leave blank for root:  
Using root .
- Input the password for the privileged account:  
Privileged account access verified.

The next of questions will ask for information on the MySQL user that will be used by the IDC to connect to the database. Please remember the information you specify for later as it will be important when configuring your IDC. Example output is shown below:

- Input a MySQL username the IDC will use to connect to the databases.  
-- This name and password must match the hibernate.connection.username and password specified in oscars.properties.  
Leave blank for "oscars":  
Using oscars .
- Input the password for the IDC account:  
IDC account access verified.
- Got all information. Press return to create the databases...  
Creating databases *bss, aaa, notify, testbss, testaaa*  
Databases created...  
Initializing databases...  
Databases initialized...  
Granting privileges to IDC account...  
IDC account authorized.  
Modifying conf/server/aaa.cfg.xml ...  
Modifying conf/server/bss.cfg.xml ...

You are now ready to build your IDC as indicated by the following output:

- Press return to build IDC...

As the last line indicates pressing return will build (compile) the IDC. You will know the build was successful when you see the following output:

```
BUILD SUCCESSFUL
Total time: 30 seconds
--- IDC built.
```

At this point you have successfully built the IDC. The next step is to deploy the software on your system with `do_install.sh`. The script will start by stopping your Tomcat server and prompting you to tell it if it should copy the OSCARS configuration files to their proper locations:

```
% ./do_install.sh
--- Checking prerequisites...
    We seem to be in the correct directory
    CATALINA_HOME is set to /usr/local/tomcat

--- Stopping Tomcat...
Using CATALINA_BASE:   /usr/local/tomcat
Using CATALINA_HOME:   /usr/local/tomcat
Using CATALINA_TMPDIR: /usr/local/tomcat/temp
Using JRE_HOME:        /usr/local/java5

Do you wish to copy key files and oscars configuration files
to Tomcat now? [y/n] y
```

You should answer ‘y’ and what follows is a long stream of output that moves the OSCARS software and configuration files to their correct locations. You will know installation was successful when you see the following:

```
IDC deployed, server configured
```

A final prompt will appear asking if you would like to build the IDC tools. You should answer ‘y’ to this prompt. It will compile a number of utilities important for configuring and running your IDC. This includes the circuit scheduler and commands for managing the database. Below is a snippet of the output you will see:

```
Should I build the OSCARS tools for you y/n?y
...
--- Tools built.
```

You are now ready to install the TERCE component and begin configuring the OSCARS software. Proceed to **3.3 Installing the TERCE** to continue installation.

## 3.2 Upgrading an Existing IDC from Version 0.3.X

Upgrading an existing IDC that is running version 0.3.0 or 0.3.1 of the DCN Software Suite requires you to run the `do_upgrade.sh` script, reissue your certificate with `idc-certadd`, and run the `do_install.sh` script. There are also a few additional steps required for inter-



domain reservations to work properly with the new version of the software that are detailed in this section. **If you are running a version of the software prior to 0.3 it is recommended that you first upgrade to version 0.3 or 0.3.1 before trying these steps.**

The first step is to run the `do_upgrade.sh` script. The commands to do this and subsequent output are as follows (NOTE: The output you see may vary depending on your installation as it checks if any steps have already been completed. This will be especially true if you run this script multiple times):

```
% cd dcn-software-suite-0.4/idc
% ./do_upgrade.sh

--- Checking prerequisites...
    We seem to be in the correct directory
    CATALINA_HOME is set to /usr/local/tomcat

- An old keystore has been detected at
  /usr/local/tomcat/shared/classes/server/sec-server.jks
  --Version 0.4 consolidates the contents of sec-
  server.jks and sec-client.jks to a new keystore,
  OSCARS.jks
  --This script will automatically copy and rename sec-
  server.jks to OSCARS.jks
  --NOTE: You will need to have your certificate in sec-
  client.jks re-issued because it uses a DSA private key
  and the WS-Policy spec now used by OSCARS requires an
  RSA private key. Please run tools/utils/idc-certadd after
  this script completes.

<Press any key to continue>
```

The above will appear if this is the first time you've run this script. It will consolidate your keystores as required for version 0.4 of this software. Pressing any key will resume the installation.

```
Keystore successfully upgraded
unzip found.
wget found.
    Found axis2 library at lib/axis2. Axis2 webapp is not
    deployed.
    Rampart library not found.

- Axis2-1.4.1 with Rampart-SNAPSHOT installation not
  detected. Should I build it for you y/n? y
```

OK, will build Axis2-1.4.1 for you.

- Axis2-1.4.1 is not deployed. Should I do this for you y/n? **y**

The above output indicates that your Axis2 library needs to be updated. Answering yes to both of the above questions will result in an automatic upgrade of Axis2 to version 1.4. This results in a lot of output as the different components download automatically. When finished the following will appear:

```
Would you like to update your Axis2 configuration for
version 1.4 y/n? y
Axis2 ready, no restart required...
Found axis2 modules at
/usr/local/tomcat/webapps/axis2/WEB-INF/services
Copying terce to /usr/local/tomcat/webapps/axis2/WEB-
INF/services
Not copying terce configuration files because they
already exist
Restarting Tomcat...
...
TERCE installed successfully!
--- Axis2 configuration upgraded.
```

Answering yes will upgrade your TERCE and perform any final steps required for the Axis2 upgrade. Once complete the script will begin prompting you to update your databases.

```
Would you like to upgrade your bss MySQL tables y/n? y
Please enter your mysql user name: oscars
--- mysql tables upgraded
Would you like to upgrade your aaa database y/n? y
Please enter your mysql user name: oscars
--- mysql tables upgraded
```

The first set of prompts upgrade the structure of the aaa database with user information and the bss database with scheduling/topology information. If this is your first time running this script you MUST answer 'y' to both of the above prompts.

```
Would you like to create the notify database y/n? y
NOTE: This action requires you to specify a privileged
MySQL user account such as 'root'
Please input a privileged mysql user (i.e. root): root
Please input password for privileged mysql user:
"Please enter the mysql user name OSCARS uses (i.e.
oscars): oscars
```

```
Please enter the password fo the mysql user name OSCARS
uses:
--- mysql tables upgraded
```

The next set of prompts creates a new database called 'notify' that is used to track where notifications of IDC activity should be sent. It requires the login and password of a privileged MySQL user so that a new table can be created and access to that new table can be created to the database user used by OSCARS (as indicated by the hibernate.connection.username property in oscars.properties).

```
Would you like to update your topology description to
match the latest schema version y/n? y
```

This version of OSCARS uses a newer version of the NMWG control plane topology schema than previous versions of the software. Answering 'y' to this prompt will bring current topology descriptions (such as those in your tedb-intra.xml and tedb-inter.xml files) up-to-date automatically.

```
- Updating oscars.properties...
Enter your IDC's URL: https://your-
idc.net:8443/axis2/services/OSCARS
-- Adding default topology service URL
-- Adding default hLS URL
-- Adding IDC PEP to notification broker
```

The above prompt adds the idc.url and the notify.broker.ws.url properties to oscars.properties. After the prompts some additional properties are set to provide default locations for the lookup service and topology service.

```
Would you like to use the new perfsonar pathfinder? [y/n]
y
```

The above configures OSCARS to use the new “perfsonar” pathfinder which automatically queries other domain's topology from an external topology service and calculates a path. It is recommended you answer 'y' at this step. Answering 'y' also requires the same answer when prompted later if you'd like to register with the topology service.

```
Would you like to activate WS-Notifications? [y/n] y
```

The above is required to do interdomain provisioning. It tells the IDC to send notification messages.

```
Would you like to subscribe to other IDCs notifications?
[y/n] y
```

The above is required to do interdomain provisioning. It tells the IDC to subscribe to other IDC's notification messages.

```
Would you like to register the local topology with the
topology service? [y/n] y
```

The above is required if you are using the perfSONAR pathfinder. It tells the IDC to register the local topology with an external topology service. The default topology service provided by Internet2 is used if you answer 'y'. You can change this by altering the `perfsonar.topology_url` and `lookup.topology.1` properties in `oscars.properties`.

```
Would you like to register this IDC and
NotificationBroker with the lookup service so other IDCs
can automatically discover it? [y/n] y
```

The above prompt will register information about the IDC and NotificationBroker (such as the URL) with the Lookup Service. It is recommended you answer 'y' because it allows other IDCs and services to automatically locate your IDC. Internet2 provides a home Lookup Service that you can use by default. You may change this location by modifying the `lookup.home.1` property in `oscars.properties`. You may optionally provide additional information about your IDC such as longitude and latitude by using the `lookup.reg` properties. See Appendix A: `oscars.properties` for more details.

```
Would you like to automatically discover the URL of
neighboring IDCs and NotificationBrokers using the lookup
service? [y/n] y
```

The above prompt asks if you'd like to automatically discover the URLs of neighboring IDCs and NotificationBrokers registered with the Lookup Service. It will also automatically update the URLs of neighboring domains if their services move to a different location. If a service is not registered with the Lookup Service you can still add the URL using the `idc-domainadd` or `idc-serviceadd` scripts. It is recommended you answer 'y' to this prompt as it may save you configuration time later.

```
May this script copy jdom.jar to
$CATALINA_HOME/webapps/axis2/WEB-INF/lib/? [y/n] y
```

The above is required to run the topology registration. It resolves a library dependency that doesn't exist otherwise.

```
#####
#####
```

UPGRADE NOTES

You may now run `./do_install.sh` to complete your upgrade.

You also need to complete the following steps:

1. Create a user account for the local domain as described in section '7.3 Activating Notifications' of the DCNSS install document

2. Add the URL of each neighboring domain's NotificationBroker by running tools/utils/idc-serviceadd as described at the end of section '7.2 Making your IDC Aware of Other Domains' of the DCNSS install document

3. Associate each neighboring domain with an Institution by running tools/utils/idc-siteadd

```
#####
#####
```

When the script is complete the above prompt displays. Prior to running `./do_install.sh` you need to re-issue your server's X.509 certificate. Version 0.4 introduces some changes that require an RSA private key and all previous versions defaulted to a DSA private key. To re-issue your certificate follow the instructions in **7.1 Generating an IDC Certificate for Sending Inter-Domain Requests**. Once completed with those steps then you may return to this section.

As the script indicates you may now run `do_install.sh` to complete the installation. Steps 2 and 3 indicated at the end of `do_upgrade.sh` are also described in this section. Running `do_install.sh` looks like the following:

```
% ./do_install.sh

--- Checking prerequisites...
    We seem to be in the correct directory
    CATALINA_HOME is set to /usr/local/tomcat

--- Stopping Tomcat...
Using CATALINA_BASE:  /usr/local/tomcat
Using CATALINA_HOME:  /usr/local/tomcat
Using CATALINA_TMPDIR: /usr/local/tomcat/temp
Using JRE_HOME:       /usr/local/java5

Do you wish to copy key files and oscars configuration
files to Tomcat now? [y/n]
n
--- Deploying IDC...
...
...
BUILD SUCCESSFUL
```

```
Total time: 39 seconds
```

```
--- Restarting Tomcat...
Using CATALINA_BASE:   /usr/local/tomcat
Using CATALINA_HOME:   /usr/local/tomcat
Using CATALINA_TMPDIR: /usr/local/tomcat/temp
Using JRE_HOME:        /usr/local/java5
```

```
IDC deployed
```

The above means that the new IDC software is deployed. Another prompt will then appear as follows:

```
Should I build the OSCARS tools for you y/n?y
```

```
--- Building tools...
Buildfile: build.xml
```

```
prepare:
```

```
compile:
```

```
idcutil:
```

```
all:
```

```
BUILD SUCCESSFUL
Total time: 3 seconds
```

```
--- Tools built.
```

The above prompts if you like to compile the tools located in the tools/utills directory. It is highly recommended you answer 'y' as these are very useful for managing the IDC.

There are a few final steps you must complete before finishing the upgrade. First, you must create a user account for the local IDC if you have not already done so. The steps for this (including a screenshot of the WBUI “Add User” form with required fields populated) is described in section **7.3 Activating** .

You must also add the URL of each of your neighboring IDC's “NotificationBrokers” to the OSCARS database. NotificationBroker is a new service that comes installed with OSCARS by default and runs at the same URL as OSCARS accepts it ends with OSCARSNotify. **YOU MAY BE ABLE TO SKIP THIS STEP AND PROCEED TO THE NEXT PARAGRAPH.** If you

answered 'y' to the `do_upgrade.sh` prompt “*Would you like to automatically discover the URL of neighboring IDCs and NotificationBrokers using the lookup service?*” AND all you neighbors are registered with a Lookup Service then this information will automatically be retrieved. Otherwise you need to manually add this URL. An example of the commands required to manually add a NotificationBroker URL for a domain are as follows:

```
% cd dcn-software-suite-0.4/idc/tools/utils
% ./idc-serviceadd

1. blue.pod.lan
2. red.pod.lan
Select the service's domain (by number): 2
Choose service type:
  1. NotificationBroker
  2. Other...
Enter Choice*: 1
Service URL*: https://their-
machine:8443/axis2/services/OSCARSNotify
New 'NB' service at 'https://their-
machine:8443/axis2/services/OSCARS' added.
```

Finally, you will need to associate each domain with an organization so that neighboring IDCs can be properly authorized to see notifications about events on their network. You can do this using the `idc-siteadd` command as follows:

```
% cd dcn-software-suite-0.4/idc/tools/utils
% ./idc-siteadd
1. blue.pod.lan
2. red.pod.lan
Select the service's domain (by number): 2

1. Energy Sciences Network
2. Internet2
3. MY ORGANIZATION
4. New...
Select the organization associated with this domain (by
number): 4
Organization Name*: Red Organization
New site created that associates domain 'red.pod.lan'
with organization 'Red Organization'
```

After completing the above your software should be completely upgraded and ready to use.

### 3.3 Installing the TERCE

The TERCE stores XML topology and the static list of local routes used by the IDC. Currently these components are kept in manually generated XML files, but in the future they will be automatically generated. You must install the TERCE to run OSCARS. The TERCE is included

in the DCN Software Suite and can only be installed after Axis2 is installed (i.e. after you run `do_build.sh`). The installation can be done by running the following commands:

```
% cd dcn-software-suite-0.4/terce
% ./do_install.sh
```

Configuring the TERCE is detailed in section **5 Describing Your Network Topology**. If this is your first time installing the software then continue to section **3.4 Creating the First User Account**.

### 3.4 Creating the First User Account

You need to create a user account so you can login to the OSCARS web page, called the web user interface (WBUI), and verify your installation. You must create the first account with a provided command-line script but additional users can also be created from the WBUI. The tools to create a user can be found in the `tool/Utils` directory. You must first change to this directory with the following command:

```
% cd dcn-software-suite-0.4/idc/tools/Utils
```

Before creating the user account, you must first add your local organization to the OSCARS database. To do this, run the following commands:

```
% ./idc-orgadd
Organization Name*: MY ORGANIZATION
New organization 'MY ORGANIZATION' added.
```

The commands to create the first user are as follows:

```
% ./idc-useradd
* indicates a required field
Login*: oscars-admin
Password*:
Confirm Password*:
First Name*: John
Last Name*: Smith
Cert Subject:
Cert Issuer:

1. Energy Sciences Network
2. Internet2
3. MY ORGANIZATION
Select the user's organization (by number): 3
```

```
1. OSCARS-user
2. OSCARS-engineer
3. OSCARS-administrator
4. OSCARS-service
5. OSCARS-operator
6. OSCARS-site-administrator
7. OSCARS-publisher
8. OSCARS-may-specify-path
Select the user's role(s) (numbers separated by spaces): 2 3
Personal Description:
```



```
Email(Primary)*: myemail@mydomain.net
Email(Secondary):
Phone(Primary)*: 5555555555
Phone(Secondary):
```

The fields with a \* are required. It is recommended at this time you just fill-in the minimum requirements. You can always modify this information later when you log-in to the WBUI. It's important to note that the user you create should have at least **OSCARS-administrator** privileges so it can create new users. You may also want to give it **OSCARS-engineer** privileges so you can view and manage all circuits. After creating this user you are almost ready to test your installation. The final step before testing your installation is to change your keystore passwords as described in the next section (*NOTE: At this point you have enough configuration done to complete the steps in sections 3.6 and 3.7 but you will run into conflicts later if you do not complete 3.5*).

### 3.5 Changing Keystore Passwords

Keystores are special files that act as a database of X.509 certificate and private keys for your IDC. It is important to keep the information in the OSCARS keystore (OSCARS.jks) protected so you **MUST** change the default passwords. The `idc-kspasswd` command can be used to change the passwords. The command to change the OSCARS.jks password is as follows: (*NOTE: the default "old" password for the keystore is **password***):

```
% cd dcn-software-suite-0.4/idc/tools/utils
% ./idc-kspasswd
Enter the old keystore password:
Enter the new keystore password:
Re-enter the new keystore password:
-- Password changed
```

After changing the keystore passwords you are ready to test your installation. If this is your first time installing the software then continue to section **3.6 Verifying the Web Service API Installation**.

### 3.6 Verifying the Web Service API Installation

You may verify that the web service API deployed correctly through the Axis2 administrator interface by doing the following:

1. Open <http://your-machine:8080/axis2/axis2-admin/> in your browser
2. Log-in to the page that loads (default user: **admin**, default password: **axis2**)
3. Click "Available Services" on the left-hand side of the screen

You should see an entry for “OSCARS” and “OSCARSNotify” on the page that loads, followed by a list of web service calls. This indicates that installation was successful. If you do not see it, try re-installing the IDC.

### 3.7 Verifying the Web User Interface (WBUI) Installation

The Web User Interface (WBUI) is a set of web pages provided with OSCARS for managing reservations and users. To verify that the WBUI installed correctly visit the following URL:

- <http://your-machine:8080/OSCARS>

A login page should appear. You can login using the account you created in section **3.4 Creating the First User Account**. Once logged-in a blank page should load (reservations will be listed on this page once you begin actively using OSCARS). At this time you cannot create circuits until you do more configurations but you can manage users (see section **4 Creating and Managing User Accounts** for more information).

***NOTE: Web User Interface DOES NOT work with INTERNET EXPLORER. Mozilla-based browsers such as Firefox or Safari are the most thoroughly tested.***

## 4 Creating and Managing User Accounts

The OSCARS IDC has a built-in system for authenticating and authorizing requests. User information is kept in the MySQL *aaa* database. The WBUI associates a username and password with accounts kept in this database. In contrast, web service requests are authenticated using X.509 certificates that are associated with the accounts in the database. Creating user accounts can be done via the WBUI as described in section **4.1**. When you are ready to begin accepting requests from other IDCs or applications using the web service API read section **4.2** on what is required to accept X.509 certificates.

### 4.1 Adding Users to the Database

Users and permissions are stored in a MySQL database. This section details how to add users to the database and assign them permissions.

#### 4.1.1 Creating New User Accounts

You can create new users from the OSCARS web page called the web user interface (WBUI). The five steps for creating a new user via this method are:

1. Visit <http://your-server:8080/OSCARS> – where *your-server* is the name of the server on which the WBUI is running

2. Login with a user account that has administrator privileges (such as the one you created in **3.4 Creating the First User Account**)
3. Click the **Users** tab on the top of the page that loads
4. Click the **Add User** button
5. Complete all the fields outlined in green. Most of the fields are self-explanatory but a few are worth mentioning:
  - *X.509 Subject Name* – Use this field to associate an X.509 certificate with a particular user. This field is not required to allow the user to provision via the WBUI. It is required if the entity associated with this account will send requests using the web service API (such as another IDC). See section **4.2.2 Associating an X.509 Certificate with a User Account** for more information on how to use this field.
  - *X.509 issuer name* – Generally the X.509 Subject Name is enough but you may also use this field to indicate the issuer of the certificate. In most cases you do not need to use this field.
  - *Choose Role(s)* – These determine what permissions users have. More complicated permissions are beyond the scope of this document.
6. Click the **Add** button on the top of the screen

#### **4.1.2 Modifying Users**

You may modify user accounts via the WBUI under the **Users** tab. Clicking on the users last name will display a form for editing the user's profile. The form should be self-explanatory as it is similar to the add users form.

#### **4.1.3 Deleting Users**

You may delete user accounts via the WBUI under the **Users** tab. Clicking on DELETE will remove the user after a confirmation.

### **4.2 Managing X.509 Certificates from Users and Other IDCs**

X.509 certificates are passed in web service messages to your IDC as a means to authenticate users and other IDCs sending requests. **You do not need to create any X.509 certificates to setup circuits on the local domain via the WBUI. You may skip this section until you are ready to begin accepting requests from applications using the web service API or accepting requests from other IDCs.** Each certificate you accept needs to be signed by a certificate authority (CA) that your server trusts. The X.509 subject of the certificate must also be

associated with a user account on your IDC. This section describes the information you need from users or other IDCs to **accept** requests. Instructions for generating a certificate your IDC inserts in messages it sends to other IDCs is described in section 7 **Inter-Domain Configuration**.

### 4.2.1 Trusted Certificate Authorities

X.509 certificates included in requests to your IDC must be issued by a certificate authority (CA) that your IDC trusts. This is important for the IDC to verify a particular request is from who the sender actually claims to be. The CAs trusted is determined by which CA certificates are stored in the `$CATALINA_HOME/shared/classes/repo/OSCARS.jks` keystore. A keystore is a file that acts as a database for storing certificates and their associated keys. By default a number of CA certificates are already installed in OSCARS (including those for Internet2's test CA and ESnet). You may view the certificates already installed by running the provided `idc-certview` script as follows:

```
% cd dcn-software-suite-0.4/idc/tools/utils
% ./idc-certview
Choose the keystore with the certificate you'd like to view:
  1. OSCARS.jks - stores the private key for sending and
public key for receiving messages from other IDCs
  2. ssl-keystore.jks - stores SSL certificates of other
IDCs running HTTPS
  3. Open certificate file...
Enter choice: 1
```

As previously mentioned, the keystore named `OSCARS.jks` maintains the CA certificates trusted to sign incoming messages. For this reason you should choose option 1 when you wish to look at the certificates already installed for this purpose. Option 2 is described in sections 7.1 and 7.4 respectively. Option 3 is described in section 4.2.2. After choosing option 1 output similar to the following displays:

```
Choose the certificate you wish to view:
  1. oasistestca
  2. oasistestrootca
  3. esnetroot
  4. doegridsca
  5. geant
  6. dcsca
  7. nortel
Enter choice: 6
```

This set of output is the list of certificates currently installed in your keystore. Your output may look slightly different depending on what you have installed. If you wish to view the details of one of the certificates then enter the number next to it and hit 'return'. Output such as the following will display:

```
-- Certificate details
```

```
Alias name: dcsca
Creation date: Mar 29, 2007
Entry type: trustedCertEntry

Owner: CN=DCSTest, OU=DCS Test, O=DCS Test, L=Ann Arbor,
ST=Michigan, C=US
Issuer: CN=DCSTest, OU=DCS Test, O=DCS Test, L=Ann Arbor,
ST=Michigan, C=US
Serial number: 0
Valid from: Thu Mar 29 11:14:29 EDT 2007 until: Sun Mar 26
11:14:29 EDT 2017
Certificate fingerprints:
    MD5:  92:21:D8:26:10:73:0A:CE:36:56:7D:F8:6C:65:9E:C6
    SHA1:
05:25:DF:20:69:78:EC:89:40:12:E7:70:01:B3:FB:D7:9E:44:9C:FF
```

If you wish to accept a request that is signed by a CA other than those currently in the keystore then you can import a new CA certificate using the provided `idc-certadd` script. **You do not need to worry about this if you only plan to accept requests from certificates signed by one of the default CAs and may progress to the next section.** Assuming you do want to add a new CA, the CA must first send you their certificate (via email, download it from a web site, etc) and you must then save it in a file. Assuming you saved it in a file called “ca.cer” you may use the following commands to import it:

```
% cd dcn-software-suite-0.4/idc/tools/utils
% ./idc-certadd What would you like to do?
  1. Create a new certificate my IDC will use in outgoing
messages to other IDCs
  2. Import a certificate created using choice 1 that was
signed by a CA
  3. Trust a CA or another IDC's certificate
Enter choice: 3
```

When you run `idc-certadd` it will prompt you for the operation to perform. Since you would like trust an incoming certificate you should choose option 3. If you’re interested, options 1 and 2 are described in section **7.1 Generating an IDC Certificate for Sending Inter-Domain Requests**. After choosing option 3 the following prompts are displayed:

```
-- You have chosen to import a trusted certificate.

Enter certificate filename: ca.cer
Enter certificate alias (This value is used to reference the
certificate in some configuration files. It may be any valid
string.): caname
```

The first prompt asks for the filename of the certificate to be imported. The second prompt asks for a certificate “alias”. The alias is only used as a reference to the certificate entry in the

keystore so it may be any value that helps you remember the CA to which the certificate belongs. You will see the alias again in commands such as `idc-certview` so make sure it is something that will help you remember the certificate. After specifying these values you will see the following output:

```
OSCARS will trust this certificate when it's used to sign...
...an incoming request y/n? y
...the SSL certificate of another IDC's web server y/n? n
```

Answer ‘y’ to the first question so that you will trust incoming requests with the certificate or another certificate signed by it. For the second question asked by the script about trusting “the SSL certificate of another IDC's web server” you may want to answer ‘y’ if you plan to connect to another IDC running HTTPS with an SSL certificate signed by the CA. More information on this is provided in section **7.4 Trusting SSL Certificates of Other IDCs Running HTTPS**. If you are not sure at this time, then you may answer ‘n’ and add the certificate later by running the script again if needed. These prompts lead to the final set of output:

```
-- Using keystore password from
/Library/Tomcat/shared/classes/server/sec-server.properties
...CERTIFICATE OUTPUT HERE...

-- Certificate imported into
/Library/Tomcat/shared/classes/server/sec-server.jks. Messages
containing this certificate or another certificate issued by
the CA it represents will now be trusted.

-- Complete
```

The output above indicates you have successfully installed the certificate. You should now trust messages signed with certificates signed by the new CA. If you ever need to delete a certificate from `sec-cerver.jks` you may do so by running `idc-certdel`. The use of this command is very similar to that of `idc-certview`. After installing the new certificate, the next step is to associate specific certificates with user accounts. This is described in the next section.

## 4.2.2 Associating an X.509 Certificate with a User Account

When a user or another IDC sends your IDC a request it needs to be able to associate the certificate in that request with an OSCARS user account. Creating user accounts via the WBUI is described in section **4.1.1 Creating New User Accounts** and modifying existing accounts is described in section **4.1.2 Modifying Users**. Copying the “subject” of the certificate being sent in the *X.509 Subject Name* field of the WBUI will associate a certificate with a user account. You may obtain the X.509 subject in a few ways. First, the user may be able to send you the subject of their certificate. If they cannot, have them send you their certificate (i.e. via email) and use `idc-certview` to extract the subject. You may use `idc-certview` to extract the subject as follows:

```

Choose the keystore or file with the certificate you'd like to
view:
  1. OSCARS.jks - stores the private key for sending and
public key for receiving messages from other IDCs
  2. ssl-keystore.jks - stores SSL certificates of other
IDCs running HTTPS
  3. Open certificate file...
Enter choice: 3
Enter the certificate filename: /home/oscars/usercert.cer
-- Certificate Details
--- NOTE: The 'Owner' field is the X.509 subject name
[CERTIFICATE OUTPUT DISPLAYS HERE]

```

When you run `idc-certview` you will be prompted for the location of the certificate. Since the certificate is in a file that the user gave you choose option **3**. You will then be prompted for the certificate filename and once provided the certificate in the file prints. As noted in the output, the *X.509 subject name* is the same as the *Owner* field that displays. Copy the value of the *Owner* field into the WBUI as previously described and your IDC will correctly associate the certificate containing the given subject and the user account you choose.

## 5 Describing Your Network Topology

### 5.1 Generating an XML Topology Description

OSCARS currently requires you to manually generate an XML file that describes your network's topology in the Open Grid Forum (OGF) Network Measurement Working Group (NMWG) control plane topology schema<sup>1</sup>. The topology description you generate describes what is *possible* on your network. For example, it is not concerned with what VLANs are currently provisioned on a network, rather the possible VLANs that could be provisioned on the network.

You can generate both an inter-domain topology description and an intra-domain topology description. The inter-domain topology description will be advertised to other networks, and the intra-domain topology description will be used for internal purposes. Currently, it is recommended you use the same topology for both. Generating the XML topology description is currently one of the most time-consuming portions of the install process.

#### 5.1.1 Example XML files

The easiest way to generate this file is to start from the two examples provided with the DCN Software Suite. Both files describe the same topology of the "blue pod" used in Internet2 DCN workshops<sup>2</sup>. You can find the example XML files in the following locations on your system:

---

<sup>1</sup> <http://anonsvn.internet2.edu/svn/nmwg/trunk/nmwg/schema/>

<sup>2</sup> <http://events.internet2.edu/2007/DCN/>

- `$CATALINA_HOME/shared/classes/terce.conf/tedb-inter.xml`
- `$CATALINA_HOME/shared/classes/terce.conf/tedb-intra.xml`

Both contain the same topology so either will be fine. You may edit them directly or edit a copy in another location on your system. You will need to point the TERCE to the final copy of your files. You may do this by modifying the following file:

- `$CATALINA_HOME/shared/classes/terce.conf/terce-ws.properties`

Edit the `tedb.static.db.interdomain` and `tedb.static.db.intradomain` properties to point to the location of each file.

### 5.1.2 Domains, Nodes, Ports and Links

The NMWG topology schema consists of a hierarchy of domains, nodes, ports, and links. The table below describes each of these elements.

Element	Child Element	Description
<b>domain</b>	node	Represents an administratively-similar set of devices.
<b>node</b>	port	Represents a network device. For this installation, it represents a VLSR
<b>port</b>	link	Represent a physical or virtual port on the network. Corresponds to a physical port number and/or DRAGON local-id for this installation.
<b>link</b>	-	Represents a connection between two ports. For the purposes of this installation, you will likely have one port per link.

### 5.1.3 Fully-Qualified Identifiers

Every element in the domain-node-port-link hierarchy has an “id” attribute. The ID takes the value of a Uniform Resource Name (URN) that contains not only an ID for the element defining it, but also its parent elements. This type of identifier is referred to as a fully-qualified identifier. These IDs always begin with the prefix “urn:ogf:network:”. This prefix is followed by a colon-delimited list of identifiers appropriate for that hierarchical level. For example, a fully-qualified port ID contains a domain ID, a node ID, and the port ID. The hierarchical level of each portion



is indicated by either a “domain=”, “node=”, “port=”, or “link=” prefix. Examples of different fully-qualified link ID types from the example topology files that come with the software are shown below:

Type	Fully-Qualified Identifier
<b>domain ID</b>	urn:ogf:network:domain=blue.pod.lan
<b>node ID</b>	urn:ogf:network:domain=blue.pod.lan:node=vlsr1
<b>port ID</b>	urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=3
<b>link ID</b>	urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=3:link=11.2.1.2

#### 5.1.4 <topology> Element

The <topology> element is the top-level element of the XML description. It contains the following important attributes and values.

Element/Attribute	Example	Description
<i>Id</i>	blue-topology	An identifier for this element. It has no significance currently and may be any string.
<idcId>	https://idc.blue.pod.lan:8443/axis2/services/OSCARS	The URL to the IDC advertising this topology. In most cases, you will only need to replace ‘idc.blue.pod.lan’ with the host on which you have installed the IDC.

#### 5.1.5 <domain> Element

The <domain> element contains a set of commonly administered nodes. In addition to < node >, <domain> contains the following attributes and children elements:

Element/Attribute	Example	Description
<i>Id</i>	urn:ogf:network:domain=blue.pod.lan	A URN representing the domain’s ID. The portion after “domain=” MUST be globally unique and SHOULD take the form of a DNS name.

### 5.1.6 <node> element

The <node> element represents a VLSR. In addition to a list of <port> elements, <domain> contains the following attributes and children elements:

Element/Attribute	Example	Description
<i>Id</i>	urn:ogf.network:domain=blue.pod.lan:node=vlsr1	A URN representing the node's ID. The portion after "node=" may be any valid string and is not required by the IDC to have any specific value.
<address>	192.168.2.4	An IP address that corresponds to the "router-id" field in the VLSR's ospfd.conf file. See DRAGON install documents for more info about router-id's and ospfd.conf.

### 5.1.7 <port> Element

The <port> element represents a connection point between VLSRs. **The ID of this element must be set a specific way for OSCARS to correctly provision circuits.** The ID field corresponds to a physical port and/or local-ID controlled by DRAGON. The ID attribute's format is different depending on the network devices being controlled.

Element/Attribute	Example	Description
<i>id</i>	urn:ogf.network:domain=blue.pod.lan:node=vlsr1:port=3	This type of ID maps directly to the local-ID or physical port of an Ethernet switch.
<i>Id</i>	urn:ogf.network:domain=anna.internet2.edu:node=vlsr1:port=1-2-1	This type of ID maps to a port on an Ethernet switch. The format is (chassis+shelf)-slot-subport
<i>Id</i>	urn:ogf.network:domain=dcn.internet2.edu:node=CHIC:port=S23647	This type of ID map is applied to ports on an ESLM card in a Ciena CoreDirector. The format is S(Subnet Interface ID). See DRAGON documentation for more information about subnet interface IDs.
<i>Id</i>	urn:ogf.network:domain=dcn.internet2.edu:node=CHIC:port=DTL1	This type of IDC maps to internal SONET links between

		CienaCoreDirectors. The DTL values map to those in the DRAGON configuration files.
<capacity>	1000000000	Maximum capacity of the port in bits per second
<maximumReservableCapacity>	1000000000	Maximum amount of capacity that can be reserved on a link in bits per second. It MUST be <= capacity
<minimumReservableCapacity>	1000000000	Minimum amount of capacity that can be reserved on a link in bits per second. It MUST be <= maximumReservableCapacity
<granularity>	1000000000	The minimum increment in which bandwidth can be reserved.

### 5.1.8 <link> Element

Element/Attribute	Example	Description
<i>id</i>	urn:ogf.network:domain=blue.pod.lan:node=vlsr1:port=3:link=11.2.1.2	The link portion of this ID MUST map to the TE address used in the DRAGON ospfd.conf files. <b>For Ethernet interfaces on the edge of your network the link-id is only used as a label and may be any value you wish (i.e. 1, *, myport, etc)</b>
<remoteLinkId>	urn:ogf.network:domain=*.node=*.port=*.link=*	The remote link to which the link is connected. Values may all be "*" if connected to an end-host or domain without a topology description.
<trafficEngineeringMetric>	1000000000	A metric associated with this link
<capacity>	1000000000	Maximum capacity of the link in bits per second. Must be <= the parent <port> capacity
<maximumReservableCapacity>	1000000000	Maximum amount of capacity that can be reserved on a link in bits per second. It

		MUST be <= capacity
<minimumReservableCapacity>	100000000	Minimum amount of capacity that can be reserved on a link in bits per second. It MUST be <= maximumReservableCapacity
<granularity>	100000000	The minimum increment in which bandwidth can be reserved.

### 5.1.9 <switchingCapabilityDescriptors> Element

This element is a child of link and contains information about its parent's switching capability. This information can be derived from DRAGON's ospfd.conf or narb.conf.

Element/Attribute	Example	Description
<switchingcapType>	l2sc	"l2sc" for Ethernet links and "tdm" for SDH/SONET links
<encodingType>	ethernet	"Ethernet" for Ethernet links and "sdh" for SDH/SONET links

### 5.1.10 <switchingCapabilitySpecificInfo> Element

This element is a child of <switchingCapabilityDescriptors> and contains information specific to the switching capability type. Currently, only elements for l2sc links are defined.

Element/Attribute	Example	Description
< interfaceMTU >	9000	The maximum transmission unit (MTU) of this link
< vlanRangeAvailability >	0, 3000-3100, 3150-3200	The range of VLANs available for provisioning on a link. Use a "-" to separate continuous ranges and a "," to separate discontinuous ranges. <b>A value of 0 indicates that untagged interfaces are allowed on the interface represented by this link. Untagged interfaces will not be allowed if a value of 0 is not in the range.</b>

## 5.2 Configuring the `terce-ws.properties` file

**NOTE:** You may skip this section if your `static-routes.xml`, `tedb-intra.xml`, and `tedb-inter.xml` file are in `$CATALINA_HOME/shared/classes/terce.conf`.

The TERCE is a web service that acts as the topology exchange and route computation element for OSCARS. In the future, it will act as the intermediary between OSCARS and the NARB, but only static topology description and route files are supported in this release. To ensure the TERCE properties file can locate the static topology and route file you have generated, it must be edited as follows:

1. Open `$CATALINA_HOME/shared/classes/terce.conf/terce-ws.properties` in a text editor
2. Change the properties file to look like the following (where *location-of-your-topology-file* is replaced with the custom path to your topology file and likewise for *location-of-your-static-routes-file*)

```
#static tedb properties
tedb.type=static
tedb.static.db.interdomain=location-of-your-topology-file
tedb.static.db.intradomain=location-of-your-topology-file

#static rce properties
rce.type=static
rce.static.file=location-of-your-static-routes-file
```

## 5.3 Populating the Scheduling Database

The final step is to import the topology information from your XML file into the OSCARS scheduling database so that it can keep track of which resources are used on the network. This is done by defining your local domain in the database and running `updateTopology.sh`.

### 5.3.1 Defining Your Local Domain

You must define your local domain so that OSCARS know which links are on your network. This information is stored in MySQL under the `bss.domains` table but you add it using the `idc-domainadd` script. You may run this script with the following commands:

```
% cd dcn-software-suite-0.4/idc/tools/utils/
% ./idc-domainadd
Topology Identifier (i.e. mydomain.net)*: blue.pod.lan
IDC URL*: https://your-machine:8443/axis2/services/OSCARS
Descriptive Name (for display purposes)*: Blue Pod
Abbreviated Name (for display purposes)*: blue
```

```

1. Energy Sciences Network
2. Internet2
3. MY ORGANIZATION
4. New...
Select the organization associated with this domain (by
number): 3
Is this your IDC's local domain? [y/n] y

```

The first field, Topology Identifier, MUST match the domain ID you use in your URNs. Also, make sure you select the organization that represents your local domain. You should have defined this value in 3.4 Creating the First User Account. If not you may create it by selecting “New...” from the menu of choices. the above commands should result in a message indicating the domain was added.

### 5.3.2 Run `updateTopology.sh`

The `updateTopology.sh` script syncs the database with your Intra-Domain topology file. The three steps for running this script are:

1. Verify Tomcat is running
2. Change your current directory to `dcn-software-suite-0.4/idc/tools/updatedbterce`

```
% cd dcn-software-suite-0.4/idc/tools/updatedbterce
```
3. Finally, run the `updateTopology.sh` script and point it to your TERCE service:

```
% ./updateTopology.sh
```

If no errors are returned, your database is now populated with the appropriate topology information.

## 5.4 Registering with a Topology Service

OSCARS can register the topology of your local domain with an external service so that other domains can consume the data. It also make it easily accessible to other tools such as those used for network monitoring. OSCARS also provides a pathfinding module that can consume your local information as well as other domain's topology information and use that to automatically calculate paths. It is HIGHLY RECOMMENDED that you enable topology registraion although it is not required unless you plan to use the above module (which is used by default). If you do not plan to register you topology with an external service please see 12.1 Creating a Static List of Local Paths.

Registering your topology with the external topology service is done by modifying values in the `oscars.properties` file. More information is provided on this file in section 8 Appendix A: `oscars.properties`. The change that you need to make to register your topology are as follows:

1. Open `$CATALINA_HOME/shared/classes/server/oscars.properties` in a text editor
2. Find the line “`#external.service.2=topology`” and remove the “`#`” symbol at the beginning of the line.
3. By default Internet2 provides a topology service located at the URL specified for the `personar.topology_url` property. If you are running your own topology service you may change this property, otherwise you can proceed to the next step
4. Restart Tomcat with the following commands to complete this task:

```
% $CATALINA_HOME/bin/shutdown.sh
% $CATALINA_HOME/bin/startup.sh
```

## 6 Verifying that You Can Build Local Circuits

If you have completed all the previous steps in this document you may now try provisioning a local circuit via the WBUI. The following steps will allow you to do this:

1. Open a browser and go to <https://your-server:8443/OSCARS> (or port 8080 if SSL is not installed)
2. Login using a user account you created
3. Click the “Create Reservation” tab
4. Enter the following value into the form (at a minimum):
  - a. Source – a fully qualified link-id or lookup service name of one edge of your network
  - b. Destination - a fully qualified link-id or lookup service name of another edge of your network
  - c. Bandwidth – the amount of bandwidth to reserve
  - d. Description – a brief description of your reservation
5. Click the “Reserve Bandwidth” button.

Assuming everything works correctly, you should see a page indicating the reservation was made. Click the “Refresh” button until you see the status of “ACTIVE” indicating that the scheduler built the circuit. Once you see ACTIVE you may begin using your circuit.

## 7 Inter-Domain Configuration

This section details the steps required to configure your IDC to communicate with other IDCs. The basic process involves generating an X.509 certificate for communication with other IDCs, adding neighboring domains to the OSCARS database, and activating notifications. Once completing this section you will be able to request circuits that span across multiple domains.

## 7.1 Generating an IDC Certificate for Sending Inter-Domain Requests

### 7.1.1 Creating a Certificate and Private Key

You must generate a certificate and private key that your domain will use to sign messages passed to other domains. This certificate and associated key are stored in the following keystore file:

- `$CATALINA_HOME/shared/classes/repo/OSCARS.jks`.

You may generate a new certificate and private key for your IDC using the `idc-certadd` script provided with OSCARS. An example of an `idc-certadd` session is shown below:

```
% cd dcn-software-suite-0.4/idc/tools/utils
% ./idc-certadd
What would you like to do?
  1. Create a new certificate my IDC will use in outgoing
  messages to other IDCs
  2. Import a certificate created using choice 1 that was
  signed by a CA
  3. Trust a CA or another IDC's certificate
Enter choice: 1
```

At this point you want to create a new certificate for your IDC so you should select option **1**. This results in the following output:

```
-- You have chosen to create a new certificate for sending
messages to other IDCs.

Enter an alias for this certificate: idccert
How many days will this certificate be valid?: 3650
```

You will be prompted for the “alias” and the number of days the certificate will be valid. The alias is an identifier used for reference purposes. It has no real meaning so choose a value that allows you to easily remember what certificate this is. The second value determines when the certificate will expire. When the certificate expires you either need to renew your current certificate or create a new one. In the example the value of 3650 indicates the certificate will not expire for about 10 years. If you are not sure what value to specify then something like 3650 is adequate. After these prompts the following output appears:

```
-- Using keystore password from
/usr/local/tomcat/shared/classes/repo/rampConfig.xml
What is your first and last name?
[Unknown]: youridc.yourdomain.net
What is the name of your organizational unit?
[Unknown]: Networking
What is the name of your organization?
[Unknown]: Your Organization
```



```

What is the name of your City or Locality?
[Unknown]: Ann Arbor
What is the name of your State or Province?
[Unknown]: MI
What is the two-letter country code for this unit?
[Unknown]: US

```

When it asks for your first and last name you don't actually have to add a first and last name. You should add a unique identifier that represents this IDC. It's your decision what this value is, but using your IDC's hostname is common. The values for the fields that follow may matter if your CA has restrictions on them. Check with your CA if you have questions about these fields (*NOTE: The Internet2 test CA allows these fields to be any value*). After entering these values the following output displays:

```

Is CN=youridc.yourdomain.net, OU=Networking, O=Your
Organization, L=Ann Arbor, ST=MI, C=US correct?
[no]: yes

-- Certificate created.

```

You will need to confirm that the values you entered are correct. If they are not then you may respond 'no' and re-enter the values.

```

Would you like to generate a Certificate Signing Request (CSR)
y/n? y
What filename should I give the CSR?: myidc.csr
Enter key password for <alias>password

```

The final two prompts ask if you'd like to create a certificate signing request (CSR). You need to do this if your certificate is going to be signed by a CA. If you answer 'y' then you will be prompted for a file in which to save the request (*Note: If you answer 'n' you may generate a CSR later by running the `idc-certsignreq` command*). You may also be prompted for the key password that is ALWAYS the value "**password**" in current OSCARS implementations. **You will not see this prompt if you keystore password is also "password"**. After your certificate signing request is generated the following output appears:

```

-- Certificate Signing Request saved in file myidc.csr
--- Please send myidc.csr to your CA for signing.
--- You may then import your signed certificate by running
idc-certadd and choosing option 2.
--- Send the following X.509 subject to your neighboring IDCs:
CN=youridc.yourdomain.net, OU=Networking, O=Your Organization,
L=Ann Arbor, ST=MI, C=US

```

These lines indicate that the certificate was successfully created and a CSR generated. If no CSR was generated then you will see output similar to the above (but indicating the lack of CSR). Most useful from this output is probably the last line that prints your certificate subject. You may send that to other IDCs and they can use it to map your IDC's user account to your IDC's

certificate. If you ever need to view the certificates subject again then run `idc-certview`, select “OSCARS.jks”, and select your certificate. The subject will be displayed in the *Owner* field. If you are not going to get your certificate signed then you are done with this section and may progress to another section. If you are going to get your certificate signed then you should continue to **7.1.2 Getting the IDC Certificate Signed**.

## 7.1.2 Getting the IDC Certificate Signed

The first step to getting your certificate signed is to send a certificate signing request (CSR) to a certificate authority (CA). This will be done via email or some other mechanism. You may have generated the CSR when you ran `idc-certadd` or you may generate it by running `idc-certsigreq`. After your CA responds to the request and gives you a signed certificate file you may import it with `idc-certadd`. An example session is shown below:

```
% cd dcn-software-suite-0.4/idc/tools/utils
% ./idc-certadd
What would you like to do?
  1. Create a new certificate my IDC will use in outgoing
messages to other IDCs
  2. Import a certificate created using choice 1 that was
signed by a CA
  3. Trust a CA or another IDC's certificate
Enter choice: 2
```

When you run the `idc-certadd` script you are given the choice of what operation you’d like to perform. You want to import a signed version of the previously created certificate so choose option 2. This leads to the following output:

```
-- You have chosen to import a signed certificate for talking
to other domains.

Enter the filename of your signed certificate:
/home/oscars/idccert.cer
```

You will then be prompted to provide the filename of the signed certificate. After specifying the filename, the script verifies a copy of the certificate belonging to the CA that signed your certificate is in OSCARS.jks. If it is not (which is likely if this is your first time creating a certificate) then you will need to import it (*NOTE: If it is then you will not see the next couple sections of output*). The CA will be able to provide you with their certificate. Example output of the situation where you need to import the CA certificate is shown below:

```
-- Using keystore password from
/usr/local/tomcat/shared/classes/repo/rampConfig.xml
-- Using certificate with the alias idccert
--- If this is not the correct alias please exit (Ctrl-C) and
modify the <user> tag in axis2.xml
```

```
-- You need to import the root certificate of the CA that
signed your certificate.
--- Your CA should have given you this file. If they did not
then they can provide it.
--- The CA certificate will have the subject 'CN=DCSTest,
OU=DCS Test, O=DCS Test, L=Ann Arbor, ST=Michigan, C=US'
Enter the filename of your CA's certificate:
/home/oscars/dcn_ca.cer
Enter an alias for your CA's certificate: i2dcnca
```

The prompts ask you for the filename of the CA certificate and the alias to identify it. The certificate alias may be any value that helps you remember it belongs to the CA. After providing these values you will see something like the following:

```
Owner: CN=DCSTest, OU=DCS Test, O=DCS Test, L=Ann Arbor,
ST=Michigan, C=US
Issuer: CN=DCSTest, OU=DCS Test, O=DCS Test, L=Ann Arbor,
ST=Michigan, C=US
Serial number: 0
Valid from: Thu Mar 29 11:14:29 EDT 2007 until: Sun Mar 26
11:14:29 EDT 2017
Certificate fingerprints:
    MD5:  92:21:D8:26:10:73:0A:CE:36:56:7D:F8:6C:65:9E:C6
    SHA1:
05:25:DF:20:69:78:EC:89:40:12:E7:70:01:B3:FB:D7:9E:44:9C:FF
Trust this certificate? [no]:  yes
Certificate was added to keystore
-- CA certificate imported
```

The top output is the CA certificate and you must verify that you want to install it. After answering 'yes' your CA certificate will be installed and you should see the following output(NOTE: If prompted for a password literally enter the string 'password'):

```
Enter key password for <idccert>password
Certificate reply was installed in keystore
-- Signed certificate imported
--- Send the following X.509 subject to your neighboring IDCs:
CN=youridc.yourdomain.net, OU=Networking, O=Your Organization,
L=Ann Arbor, ST=MI, C=US
```

This indicates that you have successfully installed a signed certificate for use by your domain.

## 7.2 Making your IDC Aware of Other Domains

You must add an entry for each domain to the bss.domains table in MySQL. An entry is needed for both direct neighbors and downstream domains with which dynamic circuit requests may traverse. This is can be done using the idc-domainadd script and the following commands:

```
% cd dcn-software-suite-0.4/idc/tools/utils/
% ./idc-domainadd
```

```
Topology Identifier (i.e. mydomain.net)*: red.pod.lan
IDC URL*: https://their-machine:8443/axis2/services/OSCARS
Descriptive Name (for display purposes)*: Red Pod
Abbreviated Name (for display purposes)*: red
```

```
1. Energy Sciences Network
2. Internet2
3. MY ORGANIZATION
4. New...
Select the organization associated with this domain (by
number): 4
Organization Name*: Red Domain Inc
New organization 'Red Domain Inc' added.
Is this your IDC's local domain? [y/n] n
```

Replace ‘red.pod.lan’ with the local part of the full-qualified domain-id URN (i.e. the part after “domain=”), ‘Red Pod’ and ‘red’ with easily recognizable strings (can be anything), and add the URL of their IDC (NOTE: you may enter ‘Unknown’ if the domain is not a direct neighbor). Also, please associate the domain with an organization. If you have not already created an organization for this domain choose the “New...” option, otherwise select an existing organization. The organization is important in allowing neighbors to retrieve information from your local IDC about reservations that cross their domain. In general, every domain should be associated with a unique organization.

You will also need to point to your neighbor's *notification broker* service. This is the service that distributes notifications about IDC activity. If your neighbor is also running OSCARS (as opposed to another flavor of IDC), the URL will be the same as the IDC URL except with the last part containing OSCARSNotify instead of OSCARS. The commands to add this service are as follows (assuming you are in the directory `dcn-software-suite-0.4/idc/tools/utills/`):

```
% ./idc-serviceadd

1. blue.pod.lan
2. red.pod.lan
Select the service's domain (by number): 2
Choose service type:
  1. NotificationBroker
  2. Other...
Enter Choice*: 1
Service URL*: https://their-
machine:8443/axis2/services/OSCARSNotify
New 'NB' service at 'https://their-
machine:8443/axis2/services/OSCARS' added.
```

You will also need to create a user account for each neighboring domain using the process described in section 4 *Creating and Managing User Accounts*. The most important piece is that the certSubject field matches the X.509 certificate subject they have provided.

The final step is to update your topology description and import it into the OSCARS scheduling database. To do this open your `tedb-intra.xml` file (found in the directory `$CATALINA_HOME/shared/classes/terce.conf` unless you did a special configuration) and update the appropriate *remoteLinkId* field(s) with the URN(s) of your neighbor. After updating and saving the file run the following commands:

```
% cd dcn-software-suite-0.4/idc/tools/updatedbterce
% ./updateTopology.sh
```

### 7.3 Activating Notifications

IDCs communicate by sending notifications between each other as important events occur. These notifications are distributed by a *notification broker* service that is installed with OSCARS by default. The IDC service must authenticate itself to the notification broker service even though they run on the same machine so that not just anyone can send notifications for distribution. This means you must create a user account for the local IDC through the WBUI or the `idc-useradd` script. In both cases you will need to set the “X.509 Subject Name” to the subject of the local IDCs certificate that was created in section **7.1 Generating an IDC Certificate for Sending Inter-Domain Requests**. You will also need to give it the “OSCARS-service” and “OSCARS-publisher” roles. An example screenshot of the WBUI “Add User” form is shown below:

Required fields are outlined in green.

Login Name

Password (Enter twice)

Password Confirmation

First Name

Last Name

X.509 subject name

X.509 issuer name

Organization

Choose Roles [Documentation](#)

None

OSCARS-user -> make reservations

OSCARS-engineer -> manage all reservations, view and update topology

OSCARS-administrator -> manage all users

OSCARS-service -> make reservations and view topology

OSCARS-publisher -> publish events to external services

OSCARS-may-specify-path -> an add-on attribute to allow specification of path elements

OSCARS-operator -> view all reservations

OSCARS-site-administrator -> manage all reservations starting or ending at site

Personal Description

E-mail (Primary)

E-mail (Secondary)

Phone Number (Primary)

Phone Number (Secondary)

After adding the new user you need to modify the `oscars.properties` to indicate that notifications should be sent. You may do this with the following steps:

1. Open `$CATALINA_HOME/shared/classes/server/oscars.properties` in a text editor
2. Remove the '#' symbol from the line `#notify.observer.2=net.es.oscars.notify.WSObserver` (or add the line “`notify.observer.N=net.es.oscars.notify.WSObserver`” if it does not contain the aforementioned line)
3. Finally, restart Tomcat with the following commands:

```
% $CATALINA_HOME/bin/shutdown.sh
% $CATALINA_HOME/bin/startup.sh
```

## 7.4 Trusting SSL Certificates of Other IDCs Running HTTPS

Many IDCs run HTTP over SSL (HTTPS) on their servers. This means that when you connect to their IDC to forward a request to them that you must trust the SSL certificate to establish the connection. Essentially, this verifies to your IDC that the remote IDC is who they say they are. Your IDC trusts the SSL certificate they give you if it is signed by a certificate authority (CA) installed in the following keystore:

- `$CATALINA_HOME/shared/classes/repo/ssl-keystore.jks`

By default certificates such as the Internet2 test CA and those from ESnet are installed. If you would like to trust SSL certificates signed by other CAs you may run `idc-certadd` to install their certificates. An example of an `idc-certadd` session to insert this type of certificate is shown below:

```
% cd dcn-software-suite-0.4/idc/tools/utils
% ./idc-certadd
What would you like to do?
  1. Create a new certificate my IDC will use in outgoing
messages to other IDCs
  2. Import a certificate created using choice 1 that was
signed by a CA
  3. Trust a CA or another IDC's certificate
Enter choice: 3
```

The script prompts you for the operation you'd like to perform. Choose option 3 since you'd like to import a new trusted certificate. The following output then displays:

```
-- You have chosen to import a trusted certificate.

Enter certificate filename: ca.cer
Enter certificate alias (This value is used to reference the
certificate in some configuration files. It may be any valid
string.): caname
```

You are then prompted for the file name of the ca certificate and an alias. Give the filename of the certificate from your CA that you wish to install. The alias can be any value that helps you remember to which CA the certificate belongs. After providing the requested values the following output is shown:

```
OSCARs will trust this certificate when it's used to sign...
...an incoming request y/n? n
...the SSL certificate of another IDC's web server y/n? y
```

You are asked which situations in which you want to trust a certificate signed by the CA. You **MUST** answer 'y' to the second prompt for the certificate to be installed in the correct keystore. You may also answer 'y' to the first question if you want to trust incoming requests that have certificates signed by the CA (see section **4.2.1 Trusted Certificate Authorities** for more

information on when to answer ‘y’ to the first question). After answering these prompts the following output will appear (more will appear if you answered ‘y’ to both).

```
-- Using keystore password from
/usr/local/tomcat/shared/classes/server/sec-server.properties
Certificate was added to keystore

-- Certificate imported into
/usr/local/tomcat/shared/classes/repo/ssl-keystore.jks. HTTPS
servers using this certificate or another certificate issued
by the CA it represents will now be trusted.

-- Complete
```

The above output indicates that the certificate was successfully installed.

## 8 Appendix A: `oscars.properties`

The `oscars.properties` file is the main area in which the OSCARS IDC retrieves installation-specific settings. These include settings for accessing the MySQL database, AAA, the perfSONAR Lookup Service, interacting with DRAGON, and more. The `oscars.properties` file is located on your system at:

- `$CATALINA_HOME/shared/classes/server/oscars.properties`

Your installation comes with a default `oscars.properties` file. This subsection details the properties required for an installation of the OSCARS IDC on a DRAGON-controlled network.

### 8.1.1 General

Property	Example Value	Description
<code>idc.url</code>	<code>http://your-idc:8443/axis2/services/OSCARS</code>	The URL of the local IDC. If not specified then OSCARS will default to the first hostname found on the current machine and assume https is running on port 8443.

### 8.1.2 MySQL Database Properties

The required properties related to the MySQL database are:

Property	Example Value	Description
<code>hibernate.connection.username</code>	<code>oscars</code>	The MySQL username the OSCARS IDC uses to access the database



<b>hibernate.connection.password</b>	mypass	The MySQL password the OSCARS IDC uses to access the database
--------------------------------------	--------	---

(Note: Hibernate is the name of the library used to manage database connections.)

### 8.1.3 Authentication, Authorization, and Accounting (AAA) Properties

The required properties related to AAA are:

Property	Example Value	Description
<b>aaa.salt</b>	os	The value with which encrypted passwords are salted. A value of 'os' means a password could be generated with <code>crypt('password','os')</code>
<b>aaa.userName</b>	oscars	The username for accessing a secure cookie. Required if running the WBUI. It may be any valid string value.
<b>aaa.sessionName</b>	oscarssess	The session name for a secure cookie. Required if running the WBUI. It may be any valid string value.
<b>aaa.secureCookie</b>	1	If set to 1 it will require users to use SSL to access the WBUI. For ease of installation this defaults to 0 but <b>it is recommended you change this value to 1 after setting-up SSL.</b>
<b>aaa.useSignalTokens</b>	1	If set to 1 then signaling tokens will be generated for each createReservation request. Signaling tokens can be transferred between a user who sends a createReservation request and another who wants to send the createPath. <b>Note: If you running the IDC on a PowerPC you MUST set this property to 0.</b>
<b>aaa.guestLogin</b>	guest	The value of this optional property is a username that can be used to assign a guest login to the WBUI. The difference between a guest and a regular user is that a guest can login to the WBUI from multiple machines. In the example given the user "guest" has this ability.

**An IDC administrator should be very careful about what permissions it gives a guest account as it is NOT secure.**

### 8.1.4 Topology Exchange and Pathfinding Properties

The required properties related to topology exchange and pathfinding are:

Property	Example Value	Description
<b>pathfinder.findPath</b>	1	In general, always set to 1. If set to 0, the user must always provide the path.
<b>pathfinder.pathMethod</b>	perfsonar,terce	This property indicates the pathfinding component to use. If a comma-separated list then it will try each pathfinder until an inter-domain path is found. Set to perfsonar if want automatic path calculation using an external topology service. Set to terce for static path calculation.
<b>tedb.tedbMethod</b>	terce	Always set to 'terce' for DRAGON installations. This property indicates the traffic engineering database type (i.e. where to get topology info).
<b>terce.url</b>	http://127.0.0.1:8080/axis2/services/TERCE	The URL to the TERCE service. In most cases, the example URL can be used exactly as shown.
<b>topo.defaultSwcapType</b>	tdm	Defaults to tdm if not specified. Indicates the default switching capability type for non-ethernet links.
<b>topo.defaultEncodingType</b>	sdh/sonet	Defaults to sdh/sonet if not specified. Indicates the default switching capability

		type for non-ethernet links.
--	--	------------------------------

### 8.1.5 Path Setup Properties

The required properties related to path setup are:

Property	Example Value	Description
<b>pss.method</b>	dragon	The method for setting up circuits. A value of 'dragon' means that the IDC will try to use DRAGON to create the path. A value of 'stub' simulates circuit setup but does not perform any action.
<b>pss.dragon.password</b>	dragon	The telnet password used to access the dragon VLSR.
<b>pss.dragon.ssh.portForward</b>	1	Sends telnet traffic over an SSH tunnel if set to 1. If 0, telnet traffic is sent directly to VLSR.
<b>pss.dragon.setERO</b>	0	Explicitly sets every link on the local path if set to 1. This is highly recommended as it ensures DRAGON uses exactly the same path reserved in OSCARS. <b>You can only set this value to 1 if you are running the version of DRAGON in DCNSS 0.3 or later.</b>
<b>pss.dragon.tunnelMode</b>	0	Optional property that will cause untagged circuits on Ciena CoreDirectors to be setup in tunnel mode if equals 1. Tunnel-mode means that an interface will accept any Ethernet packet (tagged or untagged) whereas a regular untagged circuit

		only accepts untagged packets.
<b>pss.dragon.ssh.user</b>	oscars	Required if pss.dragon.ssh.portForward= 1. The SSH user must be the same for all VLSR machines.
<b>pss.dragon.ssh.key</b>	/home/oscars/.ssh/id_rsa	Required if pss.dragon.ssh.portForward= 1. The public key used for SSH login must be the same for all VLSR machines. See ssh-keygen man pages for more information.
<b>pss.dragon.remotePort</b>	2611	The telnet port on which the DRAGON CLI is running. Most DRAGON installations will use 2611.
<b>pss.dragon.nodeId<sup>3</sup></b>	127.0.0.1	The address used by telnet to access the VLSR with <i>nodeId</i> . If pss.dragon.ssh.portForward= 1, you should set this to 127.0.0.1.
<b>pss.dragon.nodeId.ssh</b>	192.168.2.4	Required if pss.dragon.ssh.portForward= 1. The SSH address to access a VLSR with the given <i>nodeId</i> .
<b>pss.dragon.nodeId.ssh.port</b>	22	Optional. Defaults to 22. Indicates the SSH port number used to access the VLSR represented by <i>nodeId</i> via SSH. Use this property if SSH runs on a non-standard port number.
<b>pss.dragon.promptPattern</b>	vlsr	Optional. Defaults to “vlsr”. This value is a string that will be used in

<sup>3</sup> Not required if pss.dragon.ssh.portForward= 0 AND the nodeAddress field in the XML topology description is a routable address that can be used by telnet.

		the telnet prompt of all the VLSRs. You will get a large hexdump in your logs if this property is not set properly.
<b>pss.dragon.hasNarb</b>	1	Optional. Defaults to 1. Indicates whether the domain is running a NARB. If set to 0 it causes OSCARS to use tunnel-id/lsp-id local-ids when sending commands to the VLSR.
<b>pss.dragon.delay</b>	30	The number of seconds between setup/teardown operations on a VLSR

### 8.1.6 Notifications

Property	Example Value	Description
<b>notify.observer.N</b>	net.es.oscars.notify.EmailObserver	Sets a notification component. Current valid componets include one for send Email messages (net.es.oscars.notify.EmailObserver) and another for sending WS-Notification messages(net.es.oscars.notify.WSObserver). The latter must be included for an IDC to support inter-domain messaging. The value <i>N</i> is a number starting at 1.
<b>notify.ws.broker.url</b>	https://your-idc.net:8443/axis2/services/OSCAR SNotify	The URL of the local IDC's NotificationBroker
<b>notify.ws.broker.url.private</b>	https://10.0.0.2:8443/axis2/services/ OSCARSNotify	A private URL that your IDC should use to reach the NB. This can be different from the public URL you advertise. Useful for

		sites behind proxies.
<b>notify.ws.broker.regis terRetryAttempts</b>	10	The number of times to try registering with the IDC before failing. The default is 10.
<b>notify.ws.broker.seco ndsBetweenRegistrat ionRetries</b>	60	The number of seconds between retry attempts of registering with the local notification broker
<b>notifybroker.subscrip tions.maxExpireTime</b>	3600	The maximum amount of time a subscription can remain active without expiring in the notification broker
<b>notifybroker.publish ers.maxExpireTime</b>	3600	The maximum amount of time a publisher registration can remain active without expiring in the notification broker
<b>notifybroker.pep.N</b>	net.es.oscars.notify.ws.policy.IDCEv entPEP	The type of “policy enforcement point” to use. The PEP controls who can receive what notifications. If inactive any user can receive any notification. The only valid value is currently “net.es.oscars.notify.ws.policy.IDCEventPEP”
<b>mail.webmaster</b>	webmaster@blue.pod.lan	Email address of OSCARS administrator who will receive notifications from the server
<b>mail.recipients</b>	user1@my.net:user2@my.net	A colon delimited list of recipients of email notifications from OSCARS

### 8.1.7 External Services

Property	Example Value	Description
<b>external.service.N</b>	subscribe	Tells OSCARS to use an external service of some type. Valid values are subscribe and topology. Subscribe tells it to subscribe to other IDC's notifications and topology tells it to register with the topology service. <i>N</i> is a value starting at 1.
<b>perfsnar.topology_url</b>	<a href="http://packrat.internet2.edu:8012/perfSONAR_PS/services/topology">http://packrat.internet2.edu:8012/perfSONAR_PS/services/topology</a>	The URL of the topology service with which to register topology
<b>lookup.hints</b>	<a href="http://www.perfsnar.net/gls.root.hints">http://www.perfsnar.net/gls.root.hints</a>	A URL to a hints file containing a list of global lookup services.
<b>lookup.global.N</b>	<a href="http://ndb1.internet2.edu:9990/perfSONAR_PS/services/gLS">http://ndb1.internet2.edu:9990/perfSONAR_PS/services/gLS</a>	A URL to a global lookup service. <i>N</i> is a number starting at 1.
<b>lookup.home.N</b>	<a href="http://ndb1.internet2.edu:8005/perfSONAR_PS/services/hLS">http://ndb1.internet2.edu:8005/perfSONAR_PS/services/hLS</a>	A URL to a home lookup service to use for lookups if lookup.useGlobal is set to 0. <i>N</i> is a number that starts at 1.
<b>lookup.topology.N</b>	<a href="http://packrat.internet2.edu:8012/perfSONAR_PS/services/topology">http://packrat.internet2.edu:8012/perfSONAR_PS/services/topology</a>	The URL of a topology service from which to pull topology information when using the perfSONAR pathfinding module. <i>N</i> is a number that starts at 1.
<b>lookup.useGlobal</b>	1	If 1 then any lookup requests to will first try to discover the home lookup service with the data by contacting a global lookup service (as found in the hints file or set with lookup.global.N). If 0

		then it will only contact the hope lookup services specified by lookup.home.N.
<b>external.service.subscribe.termTimeWindow</b>	.2	The window of time as a percent of the total time that the IDC should renew a subscribe message. (i.e. a value of .2 means that a subscription that expires in 60 minutes will be renewed 12 minutes (.2 * 60 minutes) before the expiration time. Default is .2.
<b>external.service.subscribe.retryInterval</b>	1800	The number of seconds to wait before retrying a subscribe message to another IDC. Default is 30 minutes
<b>external.service.subscribe.topics</b>	idc:IDC	The topics to subscribe to from other IDCs Default is idc:IDC.
<b>external.service.topology.renewTime</b>	1800	The number of seconds between topology registration messages. Default is 1800 (30 minutes).

### 8.1.8 Micellaneous Properties

Other various properties are:

Property	Example Value	Description
<b>logging.rsvlogdir</b>	/usr/local/tomcat/logs	Location to keep log information on individual reservations.
<b>timeout.default</b>	600	The default number of seconds to wait for a notification from another IDC in response to a request. May be overridden by any of the other timeout properties.



<b>timeout.[create  modify  cancel]Resv.confirm</b>	600	The number of seconds to wait for a createReservation, modifyReservation, or cancel reservation confirm notification before timing out. (i.e. timeout.createResv.confirm=600)
<b>timeout.[create  modify  cancel]Resv.complete</b>	600	The number of seconds to wait for a createReservation, modifyReservation, or cancel reservation complete message before timing out. (i.e. timeout.createResv.confirm=600)
<b>timeout.[create  teardown]Path.confir m</b>	600	The number of seconds to wait for a createPath or teardownPath confirm notification before timing out. (i.e. timeout.createPath.confirm=600)
<b>timeout.[create  teardown]Path.retry Attempts</b>	10	The number of times to try to update the status of a reservation in response to a createPath or teardownPath event received from another IDC. This property is useful for domains that have clocks slightly out of sync and one may send notification of a create/teardown before the other is ready.
<b>timeout.[create  teardown]Path.retry Wait</b>	30	The number of seconds to wait before trying to update the status of a reservation in response to a confirm or complete event. This property is useful for domains that have clocks slightly out of sync and one may send notification of a create/teardown

		before the local IDC is ready.
<b>policy.vlanFilter.scope</b>	node	Defines the scope to which a VLAN must be unique. May be domain, node, port, or link. Default is node.
<b>rmi.serverPort</b>	1098	The port on which to run the RMI server. Default is 0 (0 means an arbitrary port).
<b>rmi.registryPort</b>	1099	The port on which to run the RMI registry. Default is 1099.

## 9 Appendix B: Using Hostnames Instead of URNs in Requests

An optional step that will make provisioning easier is to generate hostnames for the URNs on the edge of your network. This will prevent users from having to enter long URNs into requests for circuits on your network. Currently this functionality is provided by a **Lookup Service**. The Lookup Service is analogous to DNS in the IP world. You may run your own instance of the Lookup Service or you may use an instance Internet2 has running. To add lookup service names to the lookup service please email [deploy-dcn@internet2.edu](mailto:deploy-dcn@internet2.edu) with the list of names and URNs you would like added to the Lookup Service maintained by Internet2.

## 10 Appendix C: Manually Installing Axis2

Axis2 provides the IDC with a web service framework. It is installed in Tomcat as its own web application. This section details installation and configuration of Axis2.

### 10.1.1 Download and Installation

Download Axis2 from the project's web site at:

- [http://ws.apache.org/axis2/download/1\\_4\\_1/download.cgi](http://ws.apache.org/axis2/download/1_4_1/download.cgi)

You only need to download the **WAR (Web Archive) Distribution** of the software. **You MUST download version 1.4.1 for the IDC to function properly.**

Unpack and install the downloaded components with the following commands:

```
% unzip -d axis2-1 axis2-1.4.1-war.zip
% mv axis2-1/axis2.war $CATALINA_HOME/webapps
```

After moving the WAR file to the correct location, restart the Tomcat server:

```
% $CATALINA_HOME/bin/shutdown.sh
% $CATALINA_HOME/bin/startup.sh
```

### 10.1.2 Setting the AXIS2\_HOME Environment Variable

Once Axis2 is installed, you need to set the AXIS2\_HOME environment variable with its location. To set this environment variable, issue these commands:

```
% AXIS2_HOME=$CATALINA_HOME/webapps/axis2/WEB-INF
% export AXIS2_HOME
```

You may permanently set this variable (recommended) by adding the above commands to the profile file in your home directory (i.e. `.bash_profile` or `.profile`).

### 10.1.3 Verifying a Successful Installation

Validate that installation was successful by pointing your web browser to the following URL:

- <http://your-machine-name:8080/axis2> **OR**
- <https://your-machine-name:8443/axis2> (if SSL is configured – see section 2.3.5 **Configuring SSL**)

On the page that loads, click **Validate**. A page will appear that indicates the status of your Axis2 installation. If you see a message that reads “The core axis2 libraries are present.” Your installation was successful.

## 10.2 Manually Installing Rampart

Rampart is an Axis2 module that provides a number of the IDC’s security features. This section details its installation.

### 10.2.1 Download and Installation

Download the required version of the Rampart module from the DCNSS web site at:

- <https://wiki.internet2.edu/confluence/download/attachments/19074/rampart-SNAPSHOT.tar.gz>

Unpack and install rampart with the following commands:

```
% unzip rampart.zip
% cp rampart-1.4.1/rampart-1.4.1.mar $AXIS2_HOME/modules/
% cp rampart-1.4.1/lib/*.jar $AXIS2_HOME/lib/
```

After installing Rampart, you must restart the Tomcat server with the following commands:

```
% $CATALINA_HOME/bin/shutdown.sh
% $CATALINA_HOME/bin/startup.sh
```

## 10.2.2 Verifying a Successful Installation

You may verify that Rampart was installed correctly through the Axis2 administrator interface. To use this interface point a web browser to the following URL:

- <http://your-machine-name:8080/axis2/axis2-admin/>

You may login using the default username and password (**user: admin, password: axis2**). Click on the **Available Modules** link on the left side of the screen after logging-in. If installation was successful, you will see Rampart in the list that loads.

# 11 Appendix D: Becoming a Certificate Authority (CA)

It's possible to act as a certificate authority (CA) and sign user certificates. Currently, there are no recommendations regarding who should become a CA as a true dynamic circuit authentication and authorization (AA) infrastructure is yet to exist. This section is intended to be a reference for some of the commands needed when running your own CA.

## 11.1.1 Generating Your Own CA Certificate

You may generate your own CA certificate. This is optional and requires the OpenSSL library. There are various issues with safeguarding root certificates that are beyond the scope of this document. The five basic steps to allow you to generate a root certificate are:

1. Create a directory for your CA certificates

```
% mkdir ca_certs
```

2. Create an openssl.conf file in the new directory (search the Internet for examples)
3. Create files for tracking created certificates in directory created in step 1

```
% touch certindex.txt
% echo '100001' > serial
```

4. Create a private key

```
% openssl genrsa -des3 -out ca_private.key 1024
```

5. Create a public key certificate from the private key

```
% openssl req -new -x509 -days 3650 -key ca_private.key -out ca.cer
```

### 11.1.2 Signing User ‘Certificate Signing Requests’ (CSRs)

If you do decide to run your own CA certificate, you will be responsible for signing user Certificate Signing Requests (CSRs). This requires the **user** to run the following commands:

```
% keytool -genkey -alias user1 -keystore OSCARS.jks -storepass
password -validity 3650
% keytool -certreq -alias user1 -keystore OSCARS.jks -file
~/oscars_certs/dcstest_ca/requests/user1.csr
```

Then, the user sends you the CSR file (`user1.csr`). You can then sign the CSR and convert it to an X.509 certificate with the following commands:

```
% openssl ca -config openssl.conf -cert ca.cer -in
requests/user1.csr -keyfile ca_private.key -days 3650 -out
signed_certs/user1.cer
% openssl x509 -in signed_certs/user1.cer -out signed_certs/user1-
X509.cer
```

Afterwards, you can send the user the signed certificate and they can import it **into the same keystore they used to generate the CSR** with the following command:

```
% keytool -import -keystore OSCARS.jks -alias user1 -file user1-
X509.cer
```

## 12 Appendix E: Static Path Calculation

### 12.1 Creating a Static List of Local Paths

The current version of the OSCARS IDC requires you to statically generate intra-domain paths in an XML file. You only need to define those paths that are between the edges of your network. Defining inter-domain paths that extend beyond the local network are discussed section **7 Inter-Domain Configuration**. An example of the static paths file can be found at the following location on your system:

- `$CATALINA_HOME/shared/classes/terce.conf/static-routes.xml`

**NOTE: If your network only runs one VLSR then you do not need to generate a static-routes.xml file and may skip this section. This file is NOT used to calculate the path from edge to edge of the same node. For this reason, you also do not need to define local paths that start and terminate on the same node.**

You may edit this file directly or make a copy. You will need to point the TERCE to the final copy of this file. You may do this by modifying the following file:

- `$CATALINA_HOME/shared/classes/terce.conf/terce-ws.properties`

Edit the `rce.static.file` property to point to the location of `static-routes.xml`.

Here are some helpful guidelines when populating this file:

- `<staticPathEntry>` elements contain the paths. You will need a `<staticPathEntry>` for every source/destination combination you wish to use (including separate paths for the forward and reverse direction).
- `<srcEndpoint>` and `<destEndpoint>` elements contain **fully-qualified link-IDs**. Every time these link-IDs are seen, the associated path will be returned. If a lookup service name is used, it will get converted to a fully-qualified link-ID before making the path calculation.
- The `<path>` element carries the path to be returned when the associated source and destination are given by the user or as the ingress and egress to your network. The `<path>` contains a list of `<hop>` elements. Each of these `<hop>` elements contains a `<linkIdRef>`, which is a fully-qualified link ID.
- Each local hop in a path should be followed by the link to which it is connected. Every local hop should be indicated in the path.

*NOTE: Previous versions of OSCARS required both intra-domain and inter-domain paths to be specified in `static-routes.xml`. You can no longer define inter-domain paths in `static-routes.xml`, see section **12.2 Building Your Inter-Domain Routing Table** for information on defining inter-domain routes.*

## 12.2 Building Your Inter-Domain Routing Table

Currently OSCARS maintains static routing tables that determine the path a circuit will take to reach another domain. You can build these routing tables using the `idc-route` command found in the “tools/utlis” directory. The command allows you to indicate a destination in another domain and the “path” you will use to reach that destination. A destination may be a domain, node, port or link URN. The path to your destination may be as simple as the egress link from your domain (a *loose* inter-domain path) or it may be as specific as every ingress and egress in the path to the destination (a *strict* interdomain-path). The types of routes you enter depend on the needs of your network. Every IDC administrator should read **Section 12.2.1** and then use the list below to help decide which other sections to read and what types of routes to add:

1. If your IDC is going to manage a regional network, campus, or lab with a **single** connection to another domain running the DCN Software Suite 0.3+ (i.e. Internet2 DCN) see **12.2.2 Defining a Default Route**.
2. If your IDC is managing a network with **multiple** connections to other domains running the DCN Software Suite 0.3+ see **12.2.3 Defining a Path with Only an Egress**.

3. If your IDC is managing a network with one or more connections to domains running a version of the DCN Software Suite older than 0.2 or are running their own version of an IDC see **12.2.4 Defining a Path with Multiple Hops**.

*NOTE: In previous versions of OSCARS static-routes.xml maintained inter-domain paths as well as intra-domain path. It still maintains intra-domain paths but it does not maintain inter-domain paths. This means that you will need to add routes as indicated in this section when upgrading to 0.3 from a previous version. This new method does NOT require you to specify the exact path to every domain so this will hopefully be much easier than building your static-routes.xml table.*

### 12.2.1 Populating the <remoteLinkId> fields in `tedb-intra.xml` and `tedb-inter.xml`

The first step to adding inter-domain routes is to update your topology description with your neighboring domains' *remoteLinkIds*. You do not need to add them all at once, you can add them as you get them and the steps will be the same. First, to obtain the *remoteLinkIds* you must contact the IDC administrator of your neighboring domain (via phone, email, etc) and exchange this information. You will need to give them the fully-qualified link-id that represents the interface on your network that connects to their network. They will do the same for you and you will add the information they give you to the *remoteLinkId* field in `tedb-intra.xml` and `tedb-inter.xml` of the link you gave them. You will then run `updateTopology.sh` to update your database with this new information. The commands for this are as follows:

```
% cd dcn-software-suite-0.4/idc/tools/updatedbterce
% ./updateTopology.sh
http://127.0.0.1:8080/axis2/services/TERCE
```

### 12.2.2 Defining a Default Route

A default route is an inter-domain route that will be taken when a destination does not match any other entries in the routing tables. If the default route is the only route in the table then it will ALWAYS be used. This may be desirable when your IDC manages a network that only has a connection to one other network. Below is an example of how you may add a default route to the OSCARS routing tables:

```
% cd dcn-software-suite-0.4/idc/tools/utils
% ./idc-route add -default -loose -egress
urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=6:link=11.1.11.2
```

The above command states that if someone requests a circuit in another domain it will exit the network on `urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=6:link=11.1.11.2` if no other entry matches. The `-loose` option indicates that there may be other hops between the egress and the destination and that it will be the responsibility of other IDCs to fill-in the missing

values. If this is the only entry made to the routing tables ALL requests with a destination outside the domain will exit using the specified egress. Assuming this is the only entry, by looking at the example topology provided with the TERCE we see that this means requests will always be forwarded to the “red.pod.lan” domain. This can be seen by looking at the the *remoteLinkId* field in the definition of the link is tedb-intra.xml. Issue the above command and replace the value given to egress with a link-id on your network to create a default route.

### 12.2.3 Defining a Path with Only an Egress

You should define this type of path if you have multiple connections to other networks capable of accepting loose inter-domain paths (LIDP). A LIDP is one that does not contain the ingress and egress link of every domain in a path. You may define an LIDP containing only the egress of your network to use. The path passed to the next domain will include this egress but it will be the responsibility of the next domain to fill in any additional inter-domain hops between itself and the destination. All domains running version 0.3 or greater of the DCN Software Suite have this capability. Below is an example of a command used to create this type of route:

```
% cd dcn-software-suite-0.4/idc/tools/utils
% ./idc-route add -dest urn:ogf:network:domain=red.pod.lan -
egress
urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=6:link=11.
1.11.2
```

The above command states that any destination with a domain id of *red.pod.lan* will be routed to the next domain on the given egress link. **If the destination is not in a directly adjacent domain then you will want to specify the `-loose` option for this type of route.** Notice that the destination is a domain-id as opposed to a link-id. This means that any link-id that contains “domain= red.pod.lan” will match this entry. For example, *urn:ogf:network:domain=red.pod.lan:node=vlsr1:port=5:link=11.1.11.3* would match this entry if given as a destination. The destination field in the routing table may be a domain-id (as shown), node-id, port-id, or link-id. Destinations in requests will match the route for which there is the most specific definition. In other words, if one route matches at the node level and another matches at the domain level, then the node level match will be used. You may want to match at levels more specific than domain-d if you have multiple ways to get to a domain and it makes more geographic sense to take a particular egress when going to one node, port, or link than others. You may also make the routing decision based on the source in your domain. **Run `./idc-route -help` for more information.**

### 12.2.4 Defining a Path with Multiple Hops

You may define a route that indicates more than one inter-domain hop used to get to a destination. You should use this type of entry when talking to an IDC that requires a strict inter-domain path (SIDP). A SIDP is a path with the ingress and egress hop for each domain. In some



cases you may also define a loose inter-domain path with multiple hops, but that does not necessarily contain every hop in the path. Below is an example of a command used to define an SIDP:

```
% cd dcn-software-suite-0.4/idc/tools/utils
% ./idc-route add -dest urn:ogf:network:domain=green.pod.lan -
multi
Enter local egress:
urn:ogf:network:domain=blue.pod.lan:node=vlsr1:port=6:link=11.1.11.2
Enter hop URN or type 'end' to complete:
urn:ogf:network:domain=red.pod.lan:node=vlsr3:port=6:link=11.1.11.1
Enter hop URN or type 'end' to complete:
urn:ogf:network:domain=red.pod.lan:node=vlsr1:port=5:link=11.1.11.3
Enter hop URN or type 'end' to complete:
urn:ogf:network:domain=green.pod.lan:node=vlsr1:port=5:link=11.1.11.4
Enter hop URN or type 'end' to complete: end
Route Added
```

In the above example a SIDP is defined to *green.pod.lan*. The first element specified is the egress from the local domain (*blue.pod.lan*). The next link is the ingress to an intermediate domain (*red.pod.lan*). This link MUST be the same as that defined in the *remoteLinkId* field of your topology description (in *tedb-inta.xml*). The third link defined is the egress of *red.pod.lan* to *green.pod.lan*. The final hop is the ingress to *green.pod.lan*. Notice you do not need to define the egress in *green.pod.lan*. This is because the egress will be the destination in the user's request.

*This is the basic way in which you define a path with multiple hops. There are more advanced uses of this functionality, including LIDPs that include domain, node, or port identifiers (as opposed to link identifiers) but this is saved for a more advanced document on inter-domain routing in OSCARS.*

## References

D.Katz et al. "Traffic Engineering (TE) Extensions to OSPF Version 2", RFC3630, September 2003.

Kompella, K., Ed. and Y. Rekhter, Ed., "Routing Extensions in Support of Generalized Multi-Protocol Label Switching", RFC4202, October 2005.

K. Kompella, and Y. Rekhter, "OSPF Extensions in Support of Generalized Multi-Protocol Label Switching", RFC4203, Oct. 2005.

Braden, R. (Ed.), Zhang, L., Berson, S., Herzog, S. and S. Jamin, "Resource ReserVation Protocol -- Version 1 Functional Specification", RFC 2205, September 1997.

Awduche, D., Berger, L., Gan, D., Li, T., Srinivasan, V., and G. Swallow, "RSVP-TE: Extensions to RSVP for LSP Tunnels", RFC 3209, December 2001.

Berger, L., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Functional Description", RFC 3471, January 2003.

Berger, L., "Generalized Multi-Protocol Label Switching (GMPLS) Signaling Resource ReserVation Protocol-Traffic Engineering (RSVP-TE) Extensions", RFC 3473, January 2003.

Mannie, E., "Generalized Multi-Protocol Label Switching (GMPLS) Architecture", RFC 3945, October 2004.

Shiomoto, K., Papneja, R., Rabbat, R., "Use of Addresses in Generalized Multiprotocol Label Switching (GMPLS) Networks", RFC 4990, September 2007.

GMPLS: Architecture and Applications, Adrian Farrel, Igor Bryskin, Morgan Kaufmann (2006)

GMPLS Technologies: Broadband Backbone Networks and Systems (Optical Science and Engineering), Naoaki Yamanaka, Kohei Shiomoto, Eiji Oki, CRC Press (2006)

Optical-Network-Control-Architecture-Protocols, Greg Bernstein, Bala Rajagopalan, Debanjan Saha, Addison-Wesley Pub Co (2003).

The Internet and Its Protocols: A Comparative Approach, Adrian Farrel, Morgan Kaufman (2005)

DCN Software Suite, <https://wiki.internet2.edu/confluence/display/DCNSS/Home>

[SOAP] "SOAP Version 1.2 Part 1: Messaging Framework", W3C Recommendation.  
<http://www.w3.org/TR/soap12-part1/>

[WSDL] "Web Services Description Language (WSDL) 1.1", W3C Note. <http://www.w3.org/TR/wsdl>



## Notes

