www.internet2.edu

DCN Software Suite v0.1: OSCARS Inter-Domain Controller (IDC) Installation Guide

Table of Contents

1 Overview

1.1 About this Document

This document 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, 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. The DRAGON software and documentation location is as described in section 1.3 of this document.

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	4.1.2	http://dev.mysql.com/downloads/mysql/4.1.html
Java Development Kit (JDK)	5.0	http://java.sun.com/javase/downloads/index_jdk5.jsp
Tomcat	5.5	http://tomcat.apache.org/download-55.cgi
Axis2	1.3	http://ws.apache.org/axis2/download/1_3/download.cgi
Rampart	1.3	http://www.apache.org/dyn/mirrors/mirrors.cgi/ws/rampart/1_3/rampart-1.3.zip
Ant	1.7	http://ant.apache.org/bindownload.cgi
Java Transaction API (JTA)[1]	1.0.1b	http://sourceforge.net/project/showfiles.php?group_id=40712&package_id=127784&release_id=529023

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-0.1.tar.gz
% tar –xvf dcn-software-suite-0.1.tar
```

This will create a directory called `dcn-software-suite-0.1`. The IDC software is located in the subdirectory `dcn-software-suite-0.1/idc`.

The DRAGON software is located in the subdirectory `dcn-software-suite-0.1/dragon`. DRAGON software installation instructions are located in `dcn-software-suite-0.1/dragon/docs/DRAGON-INSTALL-0.1.pdf`.

The remainder of this document will focus on the IDC installation.

2 Preparing your Environment

This section details how to install and configure perquisite software on the machine that will be running the IDC. There are seven steps in this process:

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

[1] The link given is to a library called Hibernate that contains JTA. This is currently the easiest way to obtain the library.

4. Install the Axis2 module in Tomcat and set the environment variable AXIS2_HOME
5. Install the Rampart module in Axis2
6. Install Ant and add it to your PATH environment variable
7. Install the Java Transaction API (JTA)

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/4.1.html

Version 4.1.2 is the most extensively tested. 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/refman/4.1/en/installing.html

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. Two common package managers are *up2date* and *yum*. You may install MySQL command 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 and the necessary utilities. 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_11`, 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**.

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

It is recommended that you download **JDK Version 5 Update 11**. 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_11". 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_11 /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 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**.

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 jdk1.5.0_11 /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.

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://you-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 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.

2.4.1 Download and Installation

Download Axis2 from the project’s web site at:

- http://ws.apache.org/axis2/download/1_3/download.cgi

You only need to download the **WAR (Web Archive) Distribution** of the software. **You MUST download version 1.3 for the IDC to function properly.**

Unpack and install the downloaded components with the following commands:

```
% unzip axis2-1.3-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
```

2.4.2 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).

2.4.3 Verifying a Successful Installation

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

- http://you-machine-name:8080/axis2

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.

2.5 Rampart

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

2.5.1 Download and Installation

Download the Rampart module from the project's web site at:

- http://www.apache.org/dyn/mirrors/mirrors.cgi/ws/rampart/1_3/rampart-1.3.zip

Unpack and install rampart with the following commands:

```
% unzip rampart.zip
% cp rampart-1.3/rampart-1.3.mar $CATALINA_HOME/
    axis2/WEB-INF/modules/
```

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

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

2.5.2 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.

2.6 Ant

Ant is a tool that is used to build the IDC code and deploy various configuration files. This section details how to install and configure Ant.

2.6.1 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.

2.6.2 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).

2.6.3 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
% 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).

2.7 Java Transaction API (JTA)

The Java Transaction API (JTA) is a library used for managing database connections in the IDC. It is a single JAR file but its license restricts distribution with the DCN Software Suite. The easiest way to obtain this library is to download another package called Hibernate and extract the appropriate JAR. The four steps for obtaining and installing JTA are:

1. Download and unpack the DCN Software Suite (see section **1.3 Downloading the IDC software**)
2. Download the Hibernate-Core library at:
    - http://sourceforge.net/project/showfiles.php?group_id=40712&package_id=127784&release_id=529023
3. Unpack the downloaded files with the following command:

```
% unzip hibernate-3.2.5.ga.zip
```

4. Copy the `jta.jar` file to the lib directory of downloaded IDC software. The command to do this is:

```
% cp hibernate-3.2/lib/jta.jar dcn-software-suite-0.1/idc/lib
```

3 Installing the Inter-Domain Controller (IDC) Software

This section details how to install the IDC Software. It assumes you have installed all the prerequisites as described in the previous section, **Preparing your Environment**. The three steps covered in this section are:

1. Deploying IDC Web Services (IDC-WS)
2. Setting up the MySQL database
3. Deploying the IDC Web User Interface (WBUI)

3.1 Deploying IDC Web Services (IDC-WS)

Deploying IDC Web Services (IDC-WS) is required. The IDC –WS interface is used by other domains' IDCs to contact your domain. It may also be used by end-users running custom clients. IDC-WS will run in Axis2 under the Tomcat web container. This section will walk you through the installation and verification of the IDC-WS software components.

3.1.1 Setting the DOMAIN_HOME environment variable

The first step is to set the DOMAIN_HOME environment variable. This variable is used to point to an empty directory where intermediate copies of configuration files will be stored during

installation. The following commands create an empty directory in your home directory and points DOMAIN_HOME toward that location:

```
% mkdir ~/domain_home_dir
% DOMAIN_HOME=~/domain_home_dir
% set DOMAIN_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).

3.1.2 Installing the Web Service

To install the IDC-WS, issue these commands

```
% cd dcn-software-suite-0.1/idc
% ant dragon
% ant deploydragon
```

You will know that deployment succeeded if you see "BUILD SUCCESSFUL" on your screen after each `ant` command finishes. The command `ant dragon` compiles and packages the IDC software. The command `ant deploydragon` copies the compiled software and default configuration files into Tomcat and Axis2. It also downloads and installs the TERCE, an additional web service used for handling topology-related interactions between the IDC and DRAGON.

3.1.3 Verifying the IDC Installation

Verify that the IDC-WS interface deployed correctly through the Axis2 administrator interface by:

1. Going to http://you-machine:8080/axis2/axis2-admin/
2. Logging in using the username and password (default user: **admin,** default password: **axis2)**
3. Clicking "Available Services" on the left-hand side of the screen

You should see an entry for "OSCARS" 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.

3.2 Setting-up the MySQL Database

You must create two databases in MySQL: *aaa* and *bss*. Scripts are provided for building each of these. This section details their creation

3.2.1 Creating the *aaa* Database

The *aaa* database contains authentication, authorization, and accounting information. This includes user accounts and attributes. To create the database, issue these commands:

```
% cd dcn-software-suite-0.1/idc/sql/aaa
% mysql –u your-username –p < ./createTables.sql
```

3.2.2 Creating the *bss* Database

The Bandwidth Scheduling Subcomponent (BSS) reserves and manages network resources. The *bss* contains the list of reservations and enough local topology information to track the resources in use by those reservations. To create the database, issue these commands:

```
% cd dcn-software-suite-0.1/idc/sql/bss
% mysql –u your-username –p < ./createTables.sql
```

3.3 Deploying the IDC Web User Interface (WBUI)

The IDC Web User Interface (WBUI) is a web page that end-users may access to request circuits. It may also be used by administrators as a simple way to monitor reservations and create new users. It is not required for inter-domain provisioning or clients that plan to use the web service interface, but it is required to administer users. This section details the installation and verification of the IDC WBUI.

3.3.1 Installing the WBUI

The WBUI is installed as a web application in your Tomcat web container. The commands to install the WBUI are:

```
% cd dcn-software-suite-0.1/idc
% ant deploywar
```

You will know that deployment succeeded if you see “BUILD SUCCESSFUL” on your screen when the command finishes.

3.3.2 Verifying the WBUI Installation

Verify that the deployment succeeded by visiting the following URL:

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

A login page should appear. You may not login until you have created at least one user account. User account creation is detailed in the next section of this document.

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 MysSQL *aaa* database and web service requests are authenticated

using X.509 certificates. Creating user accounts requires issuing certificates and setting-up user accounts via the WBUI.

4.1 Managing X.509 Certificates

X.509 certificates are passed in web service messages to authenticate users. Certificates need to be signed by a certificate authority (CA) that the server trusts. This section describes how to use an external CA or how to become your own CA.

4.1.1 Keystores

The OSCARS IDC keeps certificates in keystore files. These files can be accessed by running the `keytool` command. The important keystores are:

- *`sec-server.jks`* – stores root certificates used to authenticate user requests
- *`sec-client.jks`* – stores certificates used to send requests
- *`ssl-keystore.jks`* – stores CA certificates for authenticating servers contacted that run SSL

Various copies of each live on the system.

4.1.2 Using an External CA

You may have user certificates signed by external CAs. You may see which CA certificates are installed by running the following command:

```
% keytool –list –keystore $DOMAIN_HOME/server/sec-server.jks –V
```

Installing new CA certificates requires you to run `keytool` and rebuild the OSCARS IDC code (this is likely to change in the next release). Run the following command to import a new root certificate:

```
% keytool -import -keystore $DOMAIN_HOME/sec-server.jks -alias NewCA
-file path-to-cert/ca.cer
```

Don’t forget to rebuild the code after this step.

4.1.3 Generating Your Own CA

You may also 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

```
% 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
```

4.1.4 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 sec-client.jks -storepass
password -validity 3650
% keytool -certreq -alias andy -keystore sec-client.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 sec-client.jks -alias user1 -file user1-
X509.cer
```

4.2 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.

4.2.1 Initializing the Database

First, populate the *aaa* database with initial attributes and permissions. The following commands perform this task:

```
% cd dcn-software-suite-0.1/idc/sql/aaa
```

```
% mysql –u your-username –p aaa < populateTables.sql
```

4.2.2 Default User Account

Run the following commands to create a default user that can be used to create other users through the Web User Interface (WBUI):

```
% cd dcn-software-suite-0.1/idc/sql/aaa
% mysql –u your-password –p aaa < createDefaultUser.sql
```

The login and password for this account are:

- Login Name: oscars-admin
- Password: admin

Note: You should delete this account or change the password after you have created the new user.

4.2.3 Creating a New User Account

You can create new users from the WBUI. For instructions on installing the WBUI, see section **3.3 Deploying the IDC 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 the default user account (or another account with administrative privileges if you have already setup an account)
3. Click the **Add User** button
4. 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 copy an X.509 certificate subject of a user The subject of a certificate can be determined with `keytool –list`.
    - *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.
5. Click the **Add** button on the top of the screen

4.2.4 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.

4.2.5 Deleting Users

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

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[2]. 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[3]. You can find the example XML files in the following locations on your system:

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

[2] http://anonsvn.internet2.edu/svn/nmwg/trunk/nmwg/schema/

[3] http://events.internet2.edu/2007/DCN/

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
domain	node	Represents an administratively-similar set of devices.
node	port	Represents a network device. For this installation, it represents a VLSR
port	link	Represent a physical or virtual port on the network. Corresponds to a physical port number and/or DRAGON local-id for this installation.
link	-	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
domain ID	urn:ogf:network:domain=blue.net
node ID	urn:ogf:network:domain=blue.net:node=vlsr1
port ID	urn:ogf:network:domain=blue.net:node=vlsr1:port=3
link ID	urn:ogf:network:domain=blue.net: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
Id	blue-topology	An identifier for this element. It has no significance currently and may be any string.
<idcId>	https://idc.blue.net:8443/axis2/services/OSCARS	The URL to the IDC advertising this topology. In most cases, you will only need to replace 'idc.blue.net' 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
Id	urn:ogf.network:domain=blue.net	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
Id	urn:ogf.network:domain=blue.net: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
id	urn:ogf.network:domain=blue.net:node=vlsr1:port=3	This type of ID maps directly to the local-ID or physical port of an Ethernet switch.
Id	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
Id	urn:ogf.network:domain=dcn.internet2.edu:node=chic-vlsr:port= 10.100.80.185-106	This type of ID map is applied to ports on an ESLM card in a Ciena CoreDirector. The format is (GMPLS Link Address)-(Subnet Interface ID). See DRAGON documentation for more information about subnet interface IDs.
Id	urn:ogf.network:domain=dcn.internet2.edu:node=chic-vlsr:port= 10.100.80.185	This ID may only be used for internal ports containing TDM links.
<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>	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.

5.1.8 <link> Element

Element/Attribute	Example	Description
id	urn:ogf.network:domain=blue.net:node=vlsr1:port=3:link=11.2.1.2	This type of ID maps directly to the local-ID or physical port of an Ethernet switch.
<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 <switchingCapabilitySpecficInfo> 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 >	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.

5.2 Creating a Static List of Paths

The current version of the OSCARS IDC requires you to statically generate both Intra-Domain and Inter-Domain paths. Future releases will automatically calculate these paths. The paths are kept in an XML file. An example can be found at the following location on your system:

- `$CATALINA_HOME/shared/classes/terce.conf/static-routes.xml`

You may edit this file directly or make a copy. 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. 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.
- Inter-Domain paths only need to have the edge hops specified.

5.3 Configuring the TERCE

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 files are supported. To ensure the TERCE properties file can locate the static topology and route file you have generated, it must be edited thus:

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.Inter-Domain=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
```

5.4 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`.

5.4.1 Defining Your Local Domain

You must manually specify you local domain the database so the script run in the next section knows which links are local. This is done by logging-in to MySQL and running an INSERT command, thus:

```
% mysql –u your-username –p bss
% mysql> INSERT INTO domains VALUES(NULL, ‘blue.net’, ‘blue’,
‘https://your-server:8443/axis2/services/OSCARS’, ‘blue’, 1);
% exit;
```

Replace ‘blue.net’ with the local part of you domain-id, ‘blue’ with a string for your domain (can be anything), and add the URL to your IDC.

5.4.2 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-0.1/idc/tools/updatedbterce`

```
% cd i2-dc-suite-0.1/idc/tools/updatedbterce
```

3. Finally, run the `updateTopology.sh` script and point it to your TERCE service:

```
% ./updateTopology.sh http://127.0.0.1:8080/axis2/
    services/TERCE
```

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

6 Preparing for Circuit Creation

After creating user accounts and describing your topology, there are a few final configuration steps required before you can begin creating circuits. This section describes configuration of the `oscars.properties` file, Inter-Domain settings, and the automated scheduler.

6.1 Configuring `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.

6.1.1 MySQL Database Properties

The required properties related to the MySQL database are::

Property	Example Value	Description
hibernate.connection.username	oscars	The MySQL username the OSCARS IDC uses to access the database
hibernate.connection.password	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.*)

6.1.2 Authentication, Authorization, and Accounting (AAA) Properties

The required properties related to AAA are:

Property	Example Value	Description
aaa.salt	os	The value with which encrypted passwords are salted. A value of 'os' means a password could be generated with `crypt('password','os')`
aaa.userName	oscars	The username for accessing a secure cookie. Required if running the WBUI. It may be any valid string value.
aaa.sessionName	oscarssess	The session name for a secure cookie. Required if running the WBUI. It may be any valid string value.

6.1.3 Topology Exchange and Pathfinding Properties

The required properties related to topology exchange and pathfinding are:

Property	Example Value	Description
pathfinder.findPath	1	In general, always set to 1. If set to 0, the user must always provide the path.
pathfinder.pathMethod	terce	Always set to 'terce' for DRAGON installations. This property indicates the pathfinding component to use.
tedb.tedbMethod	terce	Always set to 'terce' for DRAGON installations. This property indicates the traffic engineering database type (i.e. where to get topology info).
terce.url	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.

6.1.4 Path Setup Properties

The required properties related to path setup are:

Property	Example Value	Description
pss.method	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.
pss.dragon.password	dragon	The telnet password used to access the dragon VLSR.
pss.dragon.ssh.portForward	1	Sends telnet traffic over an SSH tunnel if set to 1. If 0, telnet traffic is sent directly to VLSR.
pss.dragon.ssh.user	oscars	Required if pss.dragon.ssh.portForward= 1. The SSH user must be the same for all VLSR machines.
pss.dragon.ssh.key	/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.
pss.dragon.remotePort	2611	The telnet port on which the DRAGON CLI is running. Most DRAGON installations will use 2611.
pss.dragon.*nodeId*[4]	127.0.0.1	The address used by telnet to access the VLSR with *nodeId*. If pss.dragon.ssh.portForward= 1, you should set this to 127.0.0.1.

[4] 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.

pss.dragon.*nodeId*.ssh	192.168.2.4	Required if pss.dragon.ssh.portForward= 1. The SSH address to access a VLSR with the given *nodeId*.

6.1.5 Other Properties

Other required properties are:

Property	Example Value	Description
logging.rsvlogdir	/usr/local/tomcat/logs	Location to keep log information on individual reservations.
lookup.url	http://127.0.0.1:8080/axis2/services/TERCE	A URL to a perfSONAR Lookup Service
mail.webmaster	webmaster@blue.net	Email address of OSCARS administrator who will receive notifications from the server
mail.userAdd.subject	OSCARS user added	Subject of emails sent from the OSCARS IDC

6.2 Inter-Domain Configuration

This section details the three steps required to send Inter-Domain requests.

6.2.1 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:

- `$CATALINA_HOME/shared/classes/repo/sec-client.jks`.

These five steps will create a certificate for your domain:

1. Generate a certificate and certificate signing request:

```
% cd $CATALINA_HOME/shared/classes/repo/
% keytool -genkey -alias myidc -keystore sec-client.jks -storepass
password -validity 3650
% keytool -certreq -alias myidc -keystore sec-client.jks -file
~/myidc.csr
```

2. Send the certificate to a CA for signing. If you are running your own CA, see section **4.1.4 Signing User 'Certificate Signing Requests' (CSRs)**.

3. Install the signed certificate into your keystore:

```
% cd $CATALINA_HOME/shared/classes/repo/
% keytool -import -keystore sec-client.jks -alias myidc -file myidc-signed.cer
```

4. Modify $CATALINA_HOME/shared/classes/repo/axis2.xml to reference your certificate by changing the <user> element to the alias of your certificate in the keystore:

```
...
<parameter name="OutflowSecurity">
    <action>
        <items>Timestamp Signature</items>
        <user>myidc</user>
...
```

5. Send the subject of your certificate to your neighboring domains. Also, verify they have they CA certificate installed. You may view the subject of you certificate with the following command:

```
% keytool –list –keystore sec-client.jks –alias myidc -V
```

6.2.2 Making your IDC Aware of Other Domains

You must add an entry for each of your neighboring domains to the bss.domains table. This is currently a manual process done with the following commands:

```
% mysql –u your-username –p bss
% mysql> INSERT INTO domains VALUES(NULL, 'red.net', 'red,
'https://their-server:8443/axis2/services/OSCARS', 'red', 0);
% exit;
```

Replace 'red.net' with the local part of your neighbor's domain-id, 'red' with a string for your neighbor (can be anything), and add the URL of their IDC.

6.2.3 SSL Certificates

If a neighboring domain runs SSL, you will need to have at least the root of their SSL certificate installed. New SSL certificates can be installed using keytool in the following location:

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

The commands to import a new SSL certificate are:

```
% cd $CATALINA_HOME/shared/classes/repo/
```

```
% keytool -import -keystore ssl-keystore.jks -alias redssl -file
redssl.cer
```

The default password of this keystore is ‘oscars’.

6.3 Running the Scheduler

The scheduler is a script that automatically builds circuits for users that do not wish to use signaling and deletes expired reservation. Running the scheduler is required for the IDC to function properly. This section describes compiling and running the scheduler script.

6.3.1 Building the Scheduler

You may build the scheduler with these three steps:

1. Change to the scheduler directory:

```
% cd dcn-software-suite-0.1/idc/tools/scheduler
```

2. Compile the code and create a repo:

```
% ant
% ant setupRepo
```

3. Finally, copy your server’s keystores and axis2.xmlfile to the new repo:

```
% cp $CATALINA_HOME/shared/classes/rep/*.jks repo/
% cp $CATALINA_HOME/shared/classes/repo/axis2.aml repo/
```

6.3.2 Running the Scheduler

You may run the scheduler with the following command:

```
% cd dcn-software-suite-0.1/idc/tools/scheduler
% nohup ./scheduler.sh > ~/scheduler.log
```

The above command will run the scheduler in the background and log DRAGON output in scheduler.log.