caBIG_logo_w_tm

***Deployment and Administration Guide v. 1.0***

**caGRID**

**Identifier Framework**

|  |  |
| --- | --- |
| ***Contacts and Support*** | |
| Calixto Melean (Developer) | Calixto.Melean@osumc.edu |
| Scott Oster (Architect) | Scott.Oster@inventrio.com |

|  |  |  |  |
| --- | --- | --- | --- |
| ***Revision History*** | | | |
| **Version** | **Date** | **Author** | **Changes** |
| **0.1** | 2010-07-09 | **Calixto Melean** | **Initial Draft** |
| **1.0** | 2010-07-29 | **Calixto Melean** | **Release** |

Chapter 1 Deployment Planning 5

Chapter 2 Prefix Authority Deployment 6

Installation 6

Configuration 14

Server Name 14

Running on Port 80 15

Startup 16

Top Level Domain Creation 18

Chapter 3 Naming Authority Deployment 21

Web Application 21

Grid Service 22

Chapter 4 Security 25

Securing the Naming Authority Containers 25

Client Configuration 25

Creating the Naming Authority Administrator 26

# Deployment Planning

Prior to deployment, the identifier prefix must be established. This involves determining the URL end points for the naming authority and the prefix authority, as well as the PURL top level domain mapping to the naming authority.

It is particularly important, and critical, to choose an appropriate domain name for the prefix authority (PURL server) and the PURL [top level domain](http://purl.oclc.org/docs/faq.html#toc1.8), as these two components make up the identifier prefix, and this is not expected to ever change after deployment. Recall that identifiers are permanent URIs by definition. The live forever.

The naming authority URL can change at anytime since it is actually hidden or protected by the prefix authority. When such a change occurs, the mapping from PURL domain to naming authority is simply updated to specify the new end point.

The rest of this guide assumes the following domains:

|  |
| --- |
| Prefix Authority (PURL Server) End Point |
| http://identifiers-pa.nci.nih.gov |
| Naming Authority Identifier (PURL Domain) |
| production |
| Naming Authority End Point |
| http://identifiers-na.nci.nih.gov/namingauthority/NamingAuthorityService |

With the settings above, the identifiers prefix becomes:

http://identifiers-pa.nci.nih.gov/production/

And, examples of identifiers are:

http://identifiers-pa.nci.nih.gov/production/7e82e853-c972-4d63-a891-cbe0260316c2

http://identifiers-pa.nci.nih.gov/production/8t868f845-c972-4d63-a891-dfe945248423

etc, etc…

When an identifier such as *http://identifiers-pa.nci.nih.gov/production/030494848* is “followed” (resolved), the prefix authority (PURL) redirects the client to *http://identifiers-na.nci.nih.gov/namingauthority/NamingAuthorityService/030494848* for resolution services.

# Prefix Authority Deployment

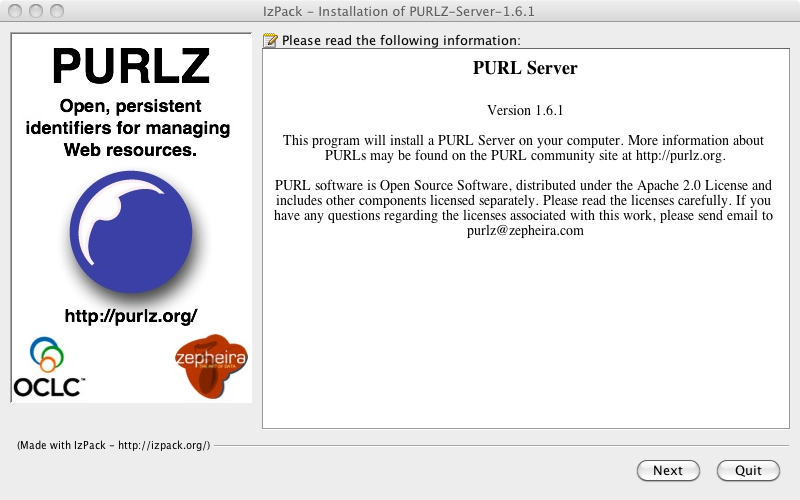
[PURLZ](http://purlz.org) is the official PURL server by OCLC. It provides a level of indirection that allows the underlying web addresses of resources to change over time without negatively affecting systems that depend on them. This capability provides continuity of references to network resources that may migrate from machine to machine.

caGrid’s identifiers framework leverages PURLZ as its prefix authority.

## Installation

Download PURLZ: <http://purlz.org/project/purl/downloads/PURLZ-Server-1.6.1.jar>

Double click the jar file or use “*java –jar PURLZ-Server-1.6.1.jar*” from a terminal window to start the installer.



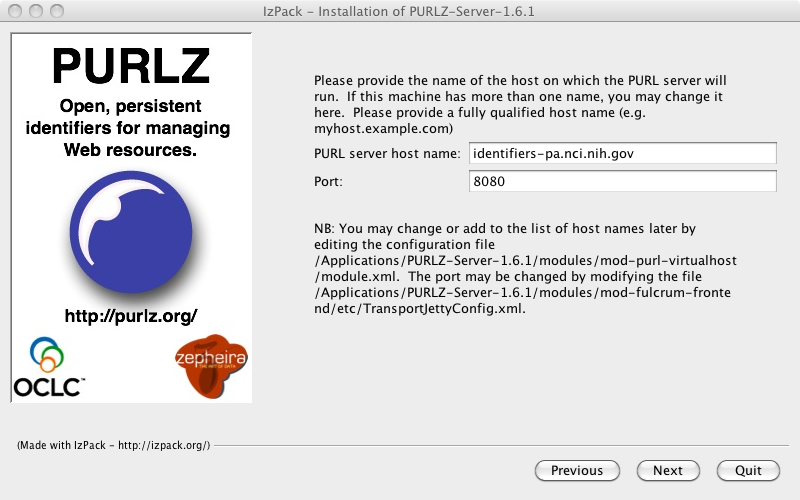
Click ***Next.***



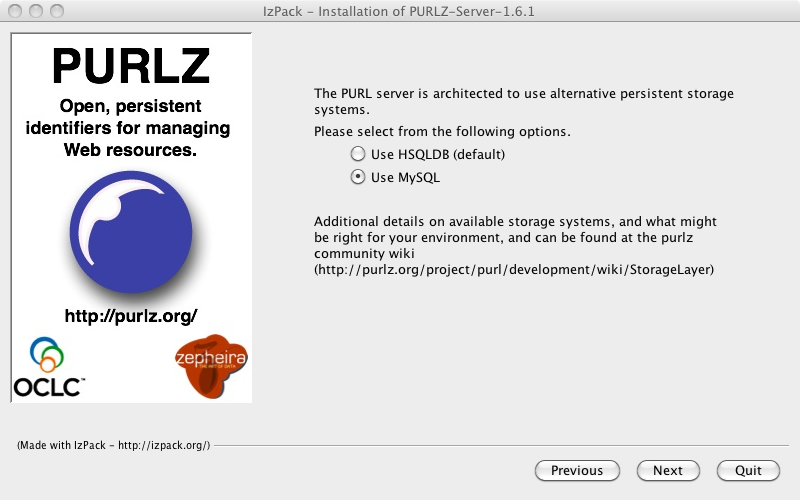
Accept the terms of the license agreement and click ***Next.***



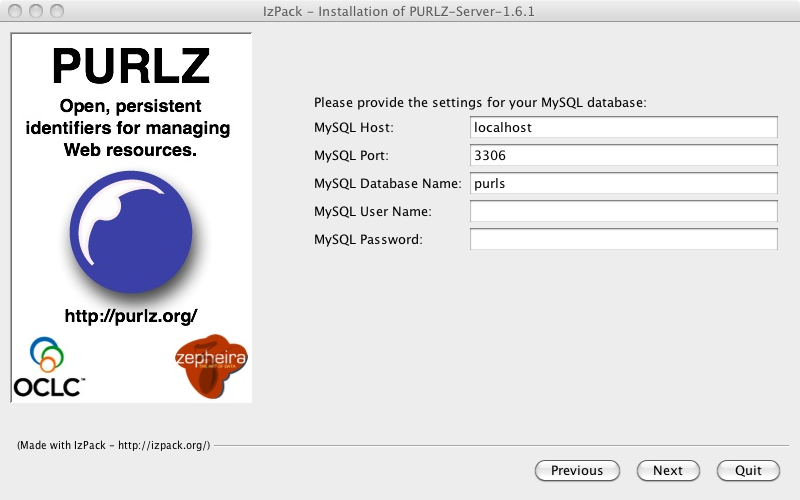
Specify an installation path and click ***Next.***



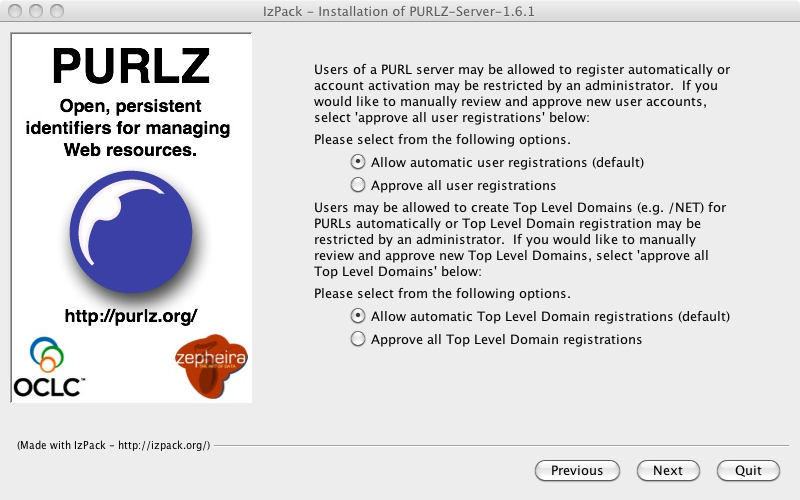
Enter the host name, port number, and click ***Next.***



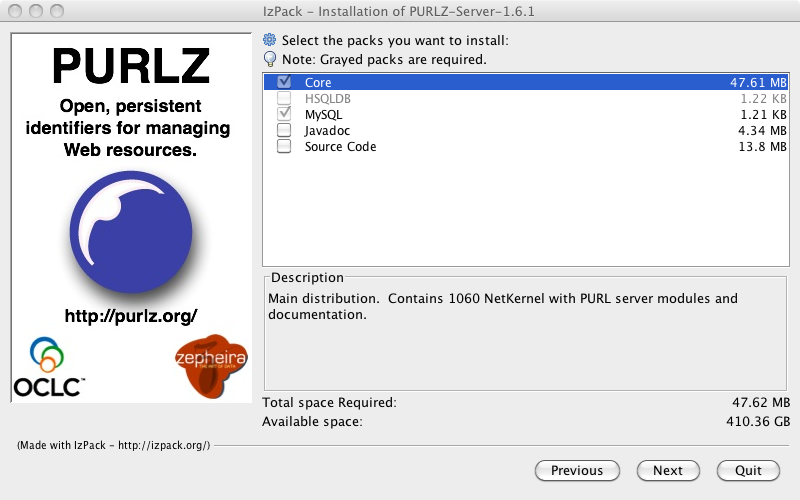
Choose “*Use MySQL*” and click ***Next.***



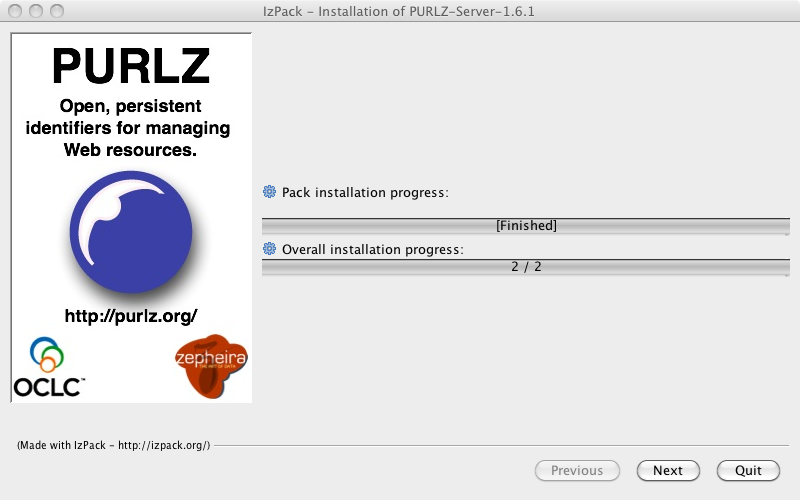
Enter *MySQL* connectivity parameters and click ***Next.***



For development and training environments, it’s probably ok to accept the defaults here. In controlled environments such as production, it is recommended that a PURL administrator be designated to approve user and top level domain registrations. Click ***Next***.



Accept defaults and click ***Next***.



The installer proceeds to complete the installation. Click ***Next*** twice and then ***Done***.

## Configuration

### Server Name

The host name ***identifiers-pa.nci.nih.gov*** has to be added to the server configuration before it can be used. Otherwise, the web interface and redirection services would only work when ***localhost*** is used in the URL.

Open */Applications/PURL-Server-1.6.1/modules/mod-purl-virtualhost/module.xml* and add our host name after *localhost* as follows (note the “.\*” after the host name):

*<export>*

*<!--*

*\*\*\*\*\*\*\*\*\*\*\**

*Export all of host address space - note could export multiple hosts here.*

*(Note have added localhost so you can test it)*

*\*\*\*\*\*\*\*\*\*\*\**

*-->*

*<uri>*

*<match>jetty://localhost.\*</match>*

*<match>****jetty://identifiers-pa.osu-citih.org.\*****</match>*

*<!-- Add any other jetty://<servername> matches that you want*

*to match. -->*

*<match>ffcpl:/etc/HTTPBridgeConfig.xml</match>*

*</uri>*

*</export>*

### Running on Port 80

The installation wizard above showed that port *8080* was entered along with the desired host name. In most setups, this wouldn’t be desired since the port number would then have to be part of the identifiers.

A problem to solve is that PURLZ seems to lack support for running on ports (i.e., 80) that are considered privileged by operating systems such as Linux. Even if the server is started by the *root* user, which is undesired, it exhibits other undocumented runtime issues.

PURLZ uses a jetty server internally, and there is jetty [documentation](http://docs.codehaus.org/display/JETTY/port80) pointing to a solution that allows the setting of a runtime user id after the port has been bound. This would potentially allow to start the server as *root* (enabling binding to port 80), and then jetty would switch to the specified runtime user id. The problem with this solution is that it requires rebuilding jetty’s source, which is again, undesirable.

Therefore, the recommended approach in this guide is to let PURLZ run on port 8080 (or other non-privileged port), and configure the operating system to redirect port 80 to the PURLZ port.

The following configuration has been tested to work on CentOS Linux.

1. Create file ***/etc/xinetd.d/http*** with the following contents

*service http*

*{*

*disable = no*

*flags = REUSE*

*socket\_type = stream*

*wait = no*

*user = root*

*redirect = 127.0.0.1 8080*

*log\_on\_failure += USERID*

*}*

1. Re-start *xinetd*

*$ /etc/init.d/xinetd restart*

### Startup

The server can be started in the foreground as follows:

$ cd /Applications/PURLZ-Server-1.6.1/bin

$ ./start.sh (or *startup.bat* if using MS Windows)

It may also be convenient to start the server as a daemon when the system starts. The following is a sample *init* script for CentOS Linux.

*#!/bin/sh*

*#*

*# Startup script for PURLZ*

*#*

*# chkconfig: - 85 15*

*# description: PURLZ server*

*# processname: purlz*

*# pidfile: /var/run/purlz.pid*

*# config:*

*##############################################################################*

*. /etc/init.d/functions*

*JAVA\_HOME=/usr/local/java*

*PURLZ\_HOME=/home/purlz/ext/purlz*

*PURLZ\_LOG=$PURLZ\_HOME/log/console.log*

*PURLZ\_USER=purlz*

*PID\_FILE=/var/run/purlz.pid*

*case "$1" in*

*start)*

*daemon --pidfile=$PID\_FILE --user=$PURLZ\_USER $PURLZ\_HOME/bin/start.sh &> $PURLZ\_HOME/log/console.log &*

*chown $PURLZ\_USER $PURLZ\_HOME/log/console.log*

*chgrp $PURLZ\_USER $PURLZ\_HOME/log/console.log*

*chmod 755 $PURLZ\_HOME/log/console.log*

*exit $?*

*;;*

*stop)*

*PID=`cat $PID\_FILE`*

*kill $PID*

*;;*

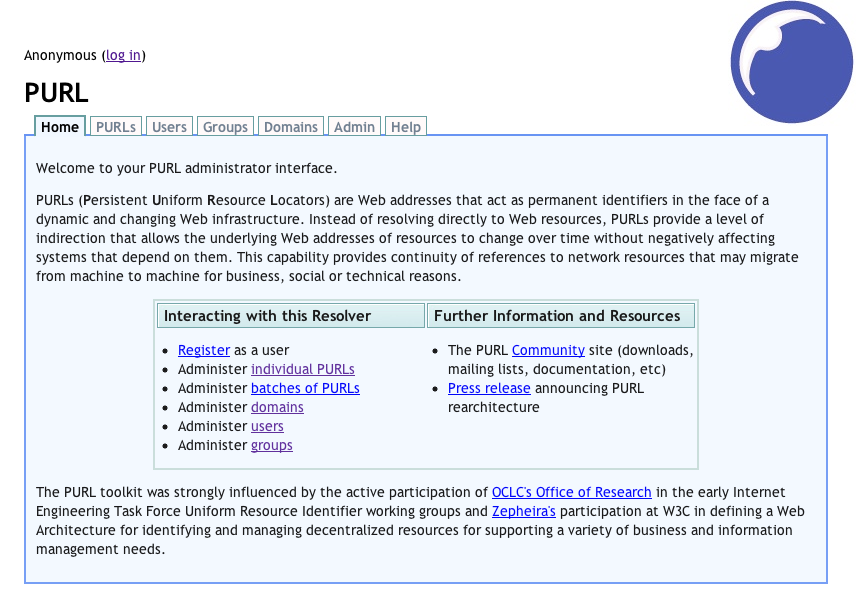
*\*)*

*echo "Usage purlz start/stop"*

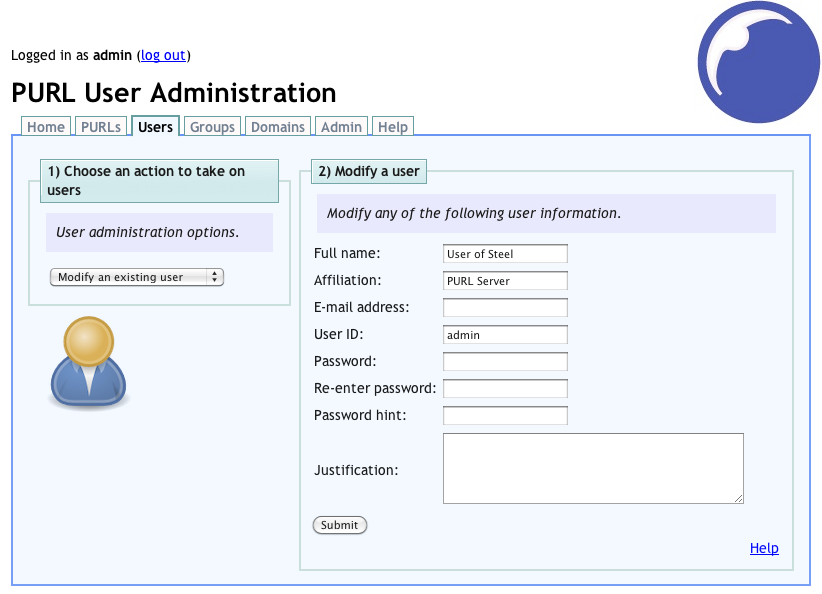
*exit 1;;*

*esac*

Once the server is started, verify it’s running by pointing your browser to *http://identifiers-pa.nci.nih.gov.* A page a like the one shown below should be displayed.



Login to the server as ‘*admin’* with password ‘*password’* and proceed to change the password.



### Top Level Domain Creation

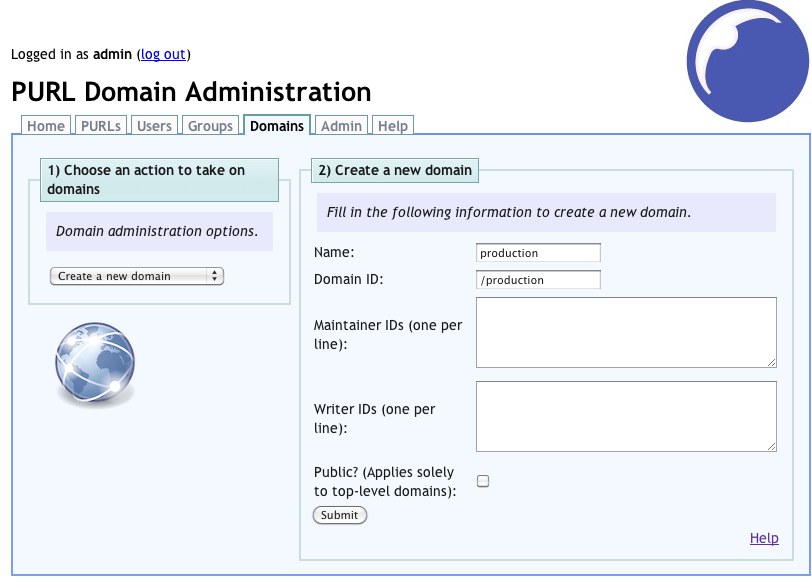
A PURL domain is needed to identify the target naming authority. The domain binds the identifier prefix to the naming authority. Therefore, a prefix authority (PURL server) can be used as an authority for multiple naming authorities by defining corresponding domains.

Following our deployment plan (Chapter 1), the goal is to create the following mapping:

***production*** => *http://identifiers-na.nci.nih.gov/namingauthority/NamingAuthorityService*

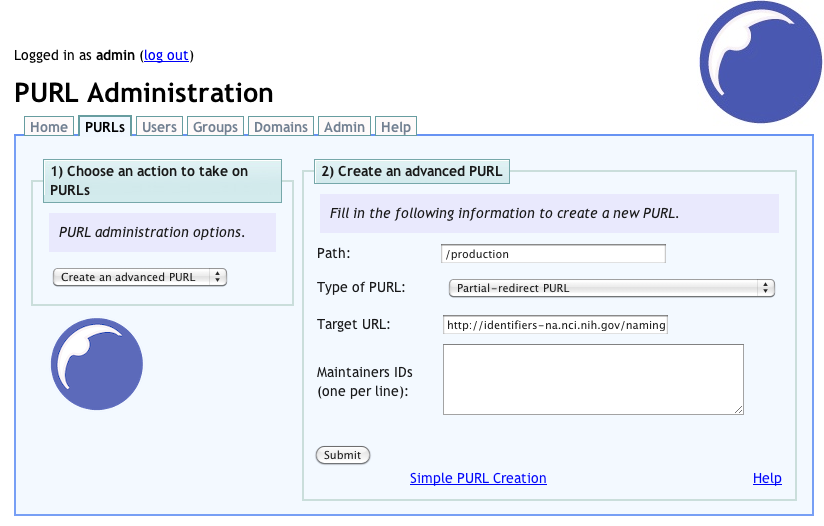
Where ***production*** is our [PURL domain](http://purl.oclc.org/docs/faq.html#toc1.8), and the mapping itself is a [partial-redirect PURL](http://purl.oclc.org/docs/faq.html#toc1.9). A domain has to be created before any PURL can placed in it.

Login as ‘*admin’* and click on *Domains* tab. Choose *Create a new domain* from the dropdown on the left, and enter the information as seen below. Click *Submit* to create the domain.



We can now create a PURL that would redirect resolution of our identifiers to their corresponding naming authority host.

Click the *PURLs* tab. Choose *Create an advanced PURL* from the dropdown on the left, and enter the information as seen below. Note that the full *Target URL* is *http://identifiers-na.nci.nih.gov/namingauthority/NamingAuthorityService*. Click *Submit* to create the PURL.



# Naming Authority Deployment

## Web Application

This section covers the installation of the naming authority core component, the web application, which enables resolution of identifiers via HTTP.

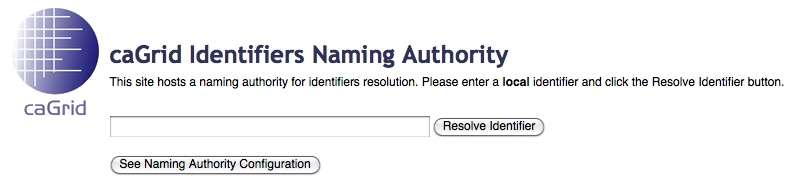
* [Install](http://cagrid.org/display/downloads/Home) caGrid 1.4 or later.
* Configure a Tomcat container to host our web application. Our deployment plan (Chapter 1) assumes this container runs on port 80.
* Customize *$CAGRID\_HOME/identifiers-namingauthority/WebContent/WEB-INF/na.properties* to match our deployment plan (*cagrid.na.prefix*). *cagrid.na.grid.url* is only needed if the optional Grid Service is being deployed.
  + **cagrid.na.prefix**=*http://identifiers-pa.nci.nih.gov/production/*
  + **cagrid.na.grid.url**=*http:*//*identifiers-na.nci.nih.gov:8080/wsrf/services/cagrid/IdentifiersNAService*
  + **cagrid.na.db.dialect**=*org.hibernate.dialect.MySQL5InnoDBDialect*
  + **cagrid.na.db.driver**=*com.mysql.jdbc.Driver*
  + **cagrid.na.db.name**=*nadb*
  + **cagrid.na.db.url**=*jdbc:mysql://localhost:3306/${cagrid.na.db.name}*
  + **cagrid.na.db.username**=*root*
  + **cagrid.na.db.password**=

**NOTE:** *cagrid.na.prefix* **MUST** include a trailing slash (/)

* Ensure the environment variable *CATALINA\_HOME* is set to point to the container previously configured.
* *cd $CAGRID\_HOME/projects/identifiers-namingauthority*
* *ant deployTomcat*
* Ensure *MySQL* is running and that the database configured above (*cagrid.na.db.name*) has been created
  + $> mysql –u root
  + mysql> create database nadb;
  + mysql> exit
* Start Tomcat
  + $CATALINA\_HOME/bin/startup.sh

(You may also want to configure Tomcat to start [automatically](http://tomcat.apache.org/tomcat-5.5-doc/setup.html)).

* Confirm the application is running by browsing *http://identifiers-na.nci.nih.gov/namingauthority*



## Grid Service

While the web application provides a *read* (HTTP resolution) interface to the naming authority, the grid service provides a *write* interface. Grid clients can leverage this interface to create and maintain identifiers.

We’ll assume here that the grid service is being deployed in its own Tomcat container running on port *8080* in host *identifiers-na.nci.nih.gov* (However, it is not necessary that the grid service be co-located with the Web Application. At minimum, at separate container is recommended).

* [Install](http://cagrid.org/display/downloads/Home) caGrid 1.4 or later.
* Configure a Globus-enabled Tomcat container (We’ll assume port 8080). This can be done with caGrid installer (previous step), or [manually](http://cagrid.org/display/knowledgebase/Manually+Configure+Tomcat+Container).
* Customize *$CAGRID\_HOME/identifiers-namingauthority-gridsvc/etc/na.properties* to match our deployment plan (*cagrid.na.prefix* and *cagrid.na.grid.url*).
  + **cagrid.na.prefix**=*http://identifiers-pa.nci.nih.gov/production/*
  + **cagrid.na.grid.url**=*http:*//*identifiers-na.nci.nih.gov:8080/wsrf/services/cagrid/IdentifiersNAService*
  + **cagrid.na.db.dialect**=*org.hibernate.dialect.MySQL5InnoDBDialect*
  + **cagrid.na.db.driver**=*com.mysql.jdbc.Driver*
  + **cagrid.na.db.name**=*nadb*
  + **cagrid.na.db.url**=*jdbc:mysql://localhost:3306/${cagrid.na.db.name}*
  + **cagrid.na.db.username**=*root*
  + **cagrid.na.db.password**=

**NOTE:** *cagrid.na.prefix* **MUST** include a trailing slash (/)

* Ensure the environment variable *CATALINA\_HOME* is set to point to the container previously configured.
* *cd $CAGRID\_HOME/projects/identifiers-namingauthority-gridsvc*
* *ant deployTomcat*
* Start Tomcat
  + $CATALINA\_HOME/bin/startup.sh

(You may also want to configure Tomcat to start [automatically](http://tomcat.apache.org/tomcat-5.5-doc/setup.html)).

* Confirm the service is running by browsing *http:*//*identifiers-na.nci.nih.gov:8080/wsrf/services/cagrid/IdentifiersNAService*



* For further verification, the *ant* target *runClient* can be run to try the creation of a simple identifier. Prior to this, ensure that the correct grid service URL is set in *run-tools.xml*



Now paste the created identifier URI into your web browser, resulting in the identifier metadata being displayed. This validates the correct operation of the prefix authority (PURL server) and the naming authority web application.

# Security

## Securing the Naming Authority Containers

The first step to execute in order to have a secure deployment is to secure the application containers (i.e. Tomcat) hosting the naming authority web application and the grid service. This document will not cover how to install certificates and configure Tomcat to use SSL. This is information is readily available from Apache [documentation](http://tomcat.apache.org/tomcat-5.5-doc/ssl-howto.html)[[1]](#footnote-2) and the web. The [caGrid installer](http://gforge.nci.nih.gov/frs/download.php/6860/caGrid-installer-1.3.0.1.zip)[[2]](#footnote-3) is also capable of installing and [configuring](https://cagrid.org/display/caGrid13/Install+caGrid+and+Configure+a+Secure+Container+Using+the+caGrid+1.3+Installer)[[3]](#footnote-4) a secure container by requesting host certificates from [Dorian](https://cagrid.org/display/dorian/Home)[[4]](#footnote-5).

Optionally[[5]](#footnote-6), in order to force the naming authority web application container to use SSL, uncomment the following block in *<PROJECT\_HOME>/WebContent/WEB-INF/web.xml*, and re-deploy to Tomcat.

|  |
| --- |
| <!-- Uncomment this to force the container to SSL  <security-constraint>  <web-resource-collection>  <web-resource-name>HTTPS Only Naming Authority</web-resource-name>  <url-pattern>/NamingAuthorityService/\*</url-pattern>  <http-method>GET</http-method>  <http-method>POST</http-method>  </web-resource-collection>  <user-data-constraint>  <transport-guarantee>CONFIDENTIAL</transport-guarantee>  </user-data-constraint>  </security-constraint>  --> |

## Client Configuration

The client toolkit (*identifiers-client*) uses [Apache HttpClient](http://hc.apache.org/httpcomponents-client/index.html)[[6]](#footnote-7), which fully leverages the [Java Secure Socket Extension (JSSE)](http://java.sun.com/javase/technologies/security/)[[7]](#footnote-8). Hence, the only requirement to enable SSL is to configure JSSE properly.

The certificate from the naming authority must be added to the *keystore* used by the JVM running the client. For example, the following command imports the naming authority certificate that was used to secure its web application container:

|  |
| --- |
| $ keytool -keystore /home/client/keystore –import –alias tomcat -file identifiers-na.nci.nih.gov-cert.pem |

The JVM that runs the client program using the resolution toolkit (*identifiers-client*) must be passed they *keystore* being used (e.g. –*Djavax.net.ssl.trustStore*=/*home/client/keystore*)

## Creating the Naming Authority Administrator

The previous sections simply cover how to secure the containers such that communications with the naming authority are encrypted.

The next level of security involves enabling the identifiers security framework. By default, everyone is allowed to create and modify identifiers. The security framework allows control over who can create, modify, or read identifiers in the system.

The *identifiers-namingauthority* project provides an *ant* target *addAdmin* to bootstrap the framework’s security by adding an initial administrator identity.



The above grid identity can now be used to perform further administration, such as:

* Turning public creation of identifiers
* Specifying identities authorized to create identifiers
* Specifying additional administrators
* etc..

1. http://tomcat.apache.org/tomcat-5.5-doc/ssl-howto.html [↑](#footnote-ref-2)
2. http://gforge.nci.nih.gov/frs/download.php/6860/caGrid-installer-1.3.0.1.zip [↑](#footnote-ref-3)
3. https://cagrid.org/display/caGrid13/Install+caGrid+and+Configure+a+Secure+Container+Using+the+caGrid+1.3+Installer [↑](#footnote-ref-4)
4. https://cagrid.org/display/dorian/Home [↑](#footnote-ref-5)
5. This is a “best practice”. It’s not absolutely required to deploy to a secure container. It’s simply a way to force the container to have SSL enabled. [↑](#footnote-ref-6)
6. http://hc.apache.org/httpcomponents-client/index.html [↑](#footnote-ref-7)
7. http://java.sun.com/javase/technologies/security [↑](#footnote-ref-8)