Installation notes for ESGF Middleware

Bugs, fixes and everything in between!

Prashanth Dwarakanath

National Supercomputer Center, Linköping Sweden

Contents

Chapter 1

ESGF Pre-setup

1.1 Setting up a vm from scratch

1. Download Centos6.3 media and install it using VirtualBox.
2. Install prerequisites:
   yum install autoconf automake bison file flex gcc gcc-c++ gettext-devel libtool libuuid-devel libxml2 libxml2-devel libxslt libxslt-devel lsof make openssl-devel pam-devel pax readline-devel tk-devel wget zlib-devel *ExtUtils* perl-Archive-Tar perl-XML-Parser
3. Install man postgresl server and screen:
   yum install postgresql-server
   yum install man
   yum install screen
4. Initialize postgresql db
   service postgresql initdb
5. Start postgresql
   /etc/init.d/postgresql start
   chkconfig –levels 345 postgresql on
6. Set up postgres user account
   su - postgres
   psql
   create role dbsuper SUPERUSER LOGIN; alter user dbsuper with encrypted password 'thepassword';
   alter user postgres with encrypted password 'thepassword'; **(this should be same as password for dbsuper specified in esgf installation)**

In order to make further deployments/experiments easier, don't use this vm/vmdisk any further. Power down the vm, clone the disk and use that cloned disk instead. Clone the original disk for all your further deployments.

Chapter

2

ESGF Test-bed Setup

In this chapter, we'll briefly get acquainted with the various components in a full-fledged ESGF setup and learn about configuring them

2.1 Components

There are four major components in the full ESGF Node software stack. They are:-

1. idp
2. data
3. index
4. compute

Identity provider: idp

The idp node comprises of :-

- **OpenID identity provider web application**: It allows users to register and authenticate with the system, including single-sign-on functionality for browser-based access throughout the federation.
- **Globus SimpleCA and MyProxy**: MyProxy is used to create short term certificates from certs which are signed with the Globus SimpleCA.

Data node

The data node comprises of :-

- **ESGF Node Manager**: Web app to enable peer-to-peer interaction among nodes across the federation. Creates the ESGF registry which contains service endpoint lists.
- **ESGF Publisher and Postgres database**: Desktop app that allows users to publish data into a node. Metadata from files is extracted and THREDDS XML catalogs created. Postgres database used.
- **Thredds Data Server and ESGF Security filters**: TDS is the standard mechanism for serving data in various forms and protocols. Security built-in.
- **GridFTP server**: High performance gsi-enabled data transfer.

Index node

The index node comprises of :-

- **Apache Solr**: High performance web-app for storing and searching metadata.
- **ESGF Search back-end utils**: For harvesting external metadata repsos such as THREDDS XML catalogs produced by ESGF Publisher), and for searching metadata indexs deployed within the federation.
- **ESGF Web Portal**: A full blown website for users.

Compute Node

The compute node comprises of :-

- **Live Access Server**: Visualization software

2.2 PreSetup

The esgf-node installation script does not come with an uninstall script that returns the node to a completely pristine state. This can be inconvenient. Opensource Tripwire could be used to track all the changes made to the system by the installation and the changes can be reverted manually. You could put together your own 'complete uninstall' script, by looking at the tripwire output! For instructions on how to install, refer to Section 4.1

2.3 Running the script

All the commands in this section are to be run as root user. Exceptions will be mentioned clearly.

2.4 Setting up the all the roles

1. Fetch the latest esgf-bootstrap script from `http://rainbow.llnl.gov/dist/esgf-installer/esg-bootstrap`
2. Run the bootstrap script. It will check for updates if any, and proceed to install a binary called esg-node in /usr/local/bin.
3. Choose the role you wish to install (data/index/idp/compute) and run the esg-node binary with appropriate arguments.
   esg-node –type data idp index compute –install

Filling in the interactive questionnaire

The virtual machine that I used had the following fqdn testnode3.demonet.local. The following lines would have values based on that. Change according to your own setup.

Prompt Question	**Value**
Admin password	<thepassword>
Organization name	demonet
Short name	idp1
Descriptive long name	idpnode1
Namespace	local.demonet
Node peer group	esgf-test
Default peer	testnode3.demonet.local
Index peer	testnode3.demonet.local
Email id	<your email id>
Database connection string	postgresql://dbsuper@localhost:5432/esgcet
Low-priv db account	esgcet
db password for publisher user	<thepassword>
Password for dbsuper role	<thepassword>

Bug in sqlalchemy

This bug was fixed in later versions of the esgf installer. For details about the bug, refer to Section 3.1

Filling in the interactive questionnaire (contd)

Prompt Question	**Value**
Would you like to use the DN: (OU=ESGF.ORG, O=ESGF) ? [Y/n]	Y
Enter key password for my_esgf_node	<your-key-password>
Do you wish to generate a Certificate Signing Request at this time?	Y

2.5 Setting up the data node only

For filling in the interactive questionnaire, refer to Section 2.4.

Bug in install script for 'data-only' installation

A bug was discovered, when a 'data-only' setup of ESGF middleware was attempted. For details, refer to Section 3.2

2.6 Setting up self-signed certificates

To setup a completely self-contained esgf installation, we first need to setup local trust, i.e. self-signed certificates. Let's explore how certificates are managed in the esgf middleware. The certificate directory is /usr/local/tomcat/conf. The private key is hostkey.pem. The setup creates <fqdn.csr> certificate signing request file which is to be signed by the CA. In this case, we intend to self-sign it. After signing the certificate, the script recommends using the esg-node –installed-keypair <private key> <signed cert> command, which is what we hope to use, when ready.

Getting our hands dirty with the certs

We start by customizing globus simple ca. This is done by:-

- Export correct value for GLOBUS_LOCATION (/usr/local/globus, by default)
- Execute /usr/local/globus/setup/globus/setupsimpleca
- Set unique subject name for CA as **cn=Globus Simple CA, ou=ESGF.ORG, o=ESGF**
- Run the following command:

```
 /usr/local/globus/setup/globus_simple_ca_<certhash>_setup/setup-gsi
```

- Follow up with this command:

```
 cd /usr/local/tomcat/conf;
/usr/local/globus/bin/grid-ca-sign -in <csrfile from tomcat directory> -out \\
./hostcert.pem
```

- Execute:

```
 esg-node --install-keypair ./hostkey.pem ./hostcert.pem
```

- You'll be asked for the path to the ca cert. Provide it. (/etc/grid-security/certificates/<your-ca-hash.0>)

- It prompts you to replace current tomcat keystore with /usr/local/tomcat/conf/keystore-tomcat, if necessary. You don't need to do anything.
- Hit enter and after a while, tomcat will be setup with the new certificates and it'll prompt for a node restart. Execute 'esg-node restart' at this point.

2.7 Useful openssl commands

To verify correctness of generated certificates, you can use:

```
openssl x509 -noout -modulus -in certificate.crt | openssl md5
openssl rsa -noout -modulus -in privateKey.key | openssl md5
openssl req -noout -modulus -in CSR.csr | openssl md5
```

Chapter 3 Caveats

This chapter lists out the bugs/caveats encountered during various stages of setting up the ESGF data node services.

3.1 Bug in sqlalchemy

It was observed that the installation was failing after a python exception. This bug was still present, as of 16th January. This document will be updated once the bug is addressed. Error extract given below:

```
Installed /usr/local/cdat/lib/python2.6/site-packages/esgf_dashboard-0.0.1-py2.
6.egg
Processing dependencies for esgf-dashboard==0.0.1
Finished processing dependencies for esgf-dashboard==0.0.1
Traceback (most recent call last):
  File "/usr/local/cdat/bin/esgf_dashboard_initialize", line 5, in <module>
   pkg_resources.run_script('esgf-dashboard==0.0.1', 'esgf_dashboard_initialize')
  File "build/bdist.linux-x86_64/egg/pkg_resources.py", line 489, in run_script
  File "build/bdist.linux-x86_64/egg/pkg_resources.py", line 1207, in run_script
  File "/usr/local/cdat/lib/python2.6/site-packages/esgf_dashboard-0.0.1-py2.6
.egg/EGG-INFO/scripts/esgf_dashboard_initialize", line 8, in <module>
    from migrate.versioning.shell import main as versioning_main
  File "/usr/local/cdat/lib/python2.6/site-packages/sqlalchemy_migrate-0.6-py2.6
.egg/migrate/versioning/shell.py", line 11, in <module>
    from migrate.versioning import api, exceptions
  File "/usr/local/cdat/lib/python2.6/site-packages/sqlalchemy_migrate-0.6-py2.6.egg/
migrate/versioning/api.py", line 32, in <module>
    from migrate.versioning import (exceptions, repository, schema, version,
  File "/usr/local/cdat/lib/python2.6/site-packages/sqlalchemy_migrate-0.6-py2.6.egg/
```

```
migrate/versioning/schema.py", line 10, in <module>
    from sqlalchemy import exceptions as sa_exceptions
ImportError: cannot import name exceptions
ERROR: Could not create esgf dashboard database tables in esgcet
```

A google search on the import error revealed a possible bug. Upon modifying the offending import statement to 'from sqlalchemy import exc as sa_exceptions', the import error disappeared.

3.2 Bug in install script for 'data-only' installation

The installer logged the following:

```
Oops, Don't see ROOT web application

*******************************
Setting up Apache Tomcat...(v6.0.36) ROOT webapp
*******************************

Downloading ROOT application from http://198.128.245.140/dist/ROOT.tgz
 Hmmm... Could not find local file /usr/local/src/esgf/workbench/esg/ROOT.tgz
Fetching file from http://198.128.245.140/dist/ROOT.tgz -to-> ROOT.tgz
--2013-01-16 14:56:09--  http://198.128.245.140/dist/ROOT.tgz
Connecting to 198.128.245.140:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1213 (1.2K) [text/plain]
Saving to: 'ROOT.tgz'

100%[======================================>] 1,213       --.-K/s   in 0s

2013-01-16 14:56:09 (55.3 MB/s) - 'ROOT.tgz' saved [1213/1213]

[VERIFIED]
unpacking ROOT.tgz...
tar: /usr/local/tomcat/webapps: Cannot chdir: No such file or directory
tar: Error is not recoverable: exiting now
 ERROR: Could not extract /usr/local/src/esgf/workbench/esg/ROOT.tgz
```

A work-around was found, by manually calling the setup_tomcat and setup_java routines. The patch file which I submitted to the esgf-user list was as follows:

```
diff --git a/esg-node b/esg-node
index ecbf2ce..3865197 100755
--- a/esg-node
+++ b/esg-node
@@ -2153,6 +2153,9 @@ setup_root_app() {
     pushd ${workdir} >& /dev/null

     echo "Downloading ROOT application from ${root_app_dist_url}"
+    echo "Trying to circumvent installer bug by setting up tomcat and java myself"
+    setup_tomcat;
+    setup_java;
     checked_get ${root_app_dist_url}
     (( $? > 1 )) && echo " ERROR: Could not download ROOT app archive" && popd && check
     echo "unpacking ${root_app_dist_url##*/}..."
```

CHAPTER 4

Miscellaneous stuff

4.1 Tripwire Setup

1. Fetch the tripwire tarball (tripwire-2.4.2.2-src.tar.bz2) from http://sourceforge.net/projects/tripwire/.

2. Install tripwire:-

   a) uz tripwire-2.4.2.2-src.tar.bz2
   b) cd tripwire-2.4.22-src
   c) ./configure –prefix=/usr/local/tripwire
   d) make
   e) In the install/install.cfg file, set TWLATEPROMPTING=true and TWLOOSEDIRCHK=true
   f) make install
   g) /usr/local/tripwire/sbin/tripwire -m i //to initialize triwire db
   h) Change directory to /usr/local/tripwire/etc
   i) Create a file called twpoltest.sh and put the following contents in it:

```
#!/bin/bash

for i in ‘cat twpol.txt |tr -s ’ ’|grep ^’ ’|grep -v ’#’ \\
|grep ’/’|awk ’{print $1}’|grep -v ’!’|grep -v ’(’‘; do
if [ ! -e $i ]; then
echo "$i not found";
fi
done
```

j) Check for non-existent directories/files in policy file with twpoltest.sh and remove them from /usr/local/tripwire/etc/twpol.txt file.

k) /usr/local/tripwire/sbin/tripwire -m p -Z low /usr/local/tripwire/etc/twpol.txt //to update the policy file

l) /usr/local/tripwire/sbin/tripwire -m c -r /tmp/report //to generate report

m) To check the report, /usr/local/tripwire/sbin/twprint -m r -r /tmp/report //to print generated report