Setup and Configuration

The *ESIP Testbed Amazon Web Services Best Practices* by Christine White is an excellent overview of AWS and great place to start. We found this document helpful in getting started and establishing consolidated billing.

- http://wiki.esipfed.org/index.php/ESIP_Testbed_Amazon_Web_Services_Best_Practices

We utilized the t1.micro instance running Ubuntu 12. The micro instance was chosen in an attempt to reduce costs as it has the lowest per hour rate.

Notes/Lessons Learned with our specific configuration

- AWS Elastic IP – our project utilized the Facebook and Google+ login APIs to authenticate users. We found that LODSpeakr (Linked Open Data hosting software), Facebook code, and Google+ code require a static IP address. However, because we were stopping the AWS instance when save costs when not in use, we ended up getting a new hostname and IP each time the instance was started. Initially, this required reconfiguration of code each time the instance was started. We subsequently obtained an AWS Elastic IP, which provided a static IP address across re-starts of the instance. Lesson learned from this is that starting/stopping the instance is a cost saving measure, but it may have unintended consequences on your application. Further, AWS Elastic IP incurs its own costs (current 1 cent per hour), which need to be factored into your project.

- We utilized the AWS micro instance as it had the lowest cost. We ran into an issue of not having enough memory to insert large amounts of RDF in our RDF database. This was solvable by creating scripts to loop through the RDF files and insert them one at a time rather than in bulk; however, this slowed down development. We had roughly 2GB of RDF data and we note that the micro instance may cause problems for other Testbed applications with larger data requirements.

- We also ran into storage issues with the AWS micro instance not being able to hold all of our data simultaneously. We addressed this by working in a piecemeal fashion and removing data when it was no longer needed. We note this because we had relatively minor data sizes (a few GB). Projects with larger data requirements will likely need a larger AWS instance, which comes with a substantial increase in price.

- We did find AWS ideal for collaborative development. Creating multiple accounts and editing the same project was much simpler than our previous experiences using university or government resources.

Our project required the use of Virtuoso, Apache Tomcat, and LODSpeakr (a project for hosting Linked Open Data). What follows next are installation instructions for each of these.

 # update the operating system and software

sudo apt-get update

 # install virtuoso and additional packages needed to configure/use

sudo apt-get install emacs
sudo apt-get install virtuoso-opensource
sudo apt-get install subversion

 # get scripts for virtuoso from our open source library – scripts for insert/deleting graphs

svn checkout http://linked-open-data-essi.googlecode.com/svn/utils/virtuoso-scripts

 # set up isql symlink

ln -s /usr/bin/isql-vt isql

 # set up temporary directory for virtuoso loading

sudo mkdir /tmp/vad
emacs /etc/virtuoso-opensource-6.1/virtuoso.ini
    add /tmp/vad to DirsAllowed

 # update virtuoso scripts

edit vload, replace "/opt/virtuoso-6.1.4/bin/isql" => "/usr/bin/isql"
edit vdelete, replace "/opt/virtuoso-6.1.4/bin/isql" => "/usr/bin/isql"
edit vrename, replace "/opt/virtuoso-6.1.4/bin/isql" => "/usr/bin/isql"
edit vdelete, replace "/opt/virtuoso-6.1.4/bin/isql" => "/usr/bin/isql"
ln -s /usr/bin/isql-vt /usr/bin/isql

 # install LODSPeaKr

mkdir /var/www/
cd /var/www/
sudo apt-get install apache2 curl php5-cli php5 php5-sqlite php5-curl git sqlite3
sudo a2enmod rewrite
edit /etc/apache2/sites-enabled/000-default
    in <Directory /var/www/> change option to AllowOverride All
sudo service apache2 restart

bash < <(curl -sL http://lodspeakr.org/install)
chown -R www-data lodspeakr/cache lodspeakr/meta

 # install Tomcat

sudo apt-get install tomcat7
sudo apt-get install tomcat7-admin

 # update .bashrc with appropriate environment variables

sudo vi ~/.bashrc
export JAVA_HOME=/usr/lib/jvm/default-java
export CATALINA_HOME=/usr/share/tomcat7
export CATALINA_BASE=/var/lib/tomcat7

 # update tomcat admin account

sudo vi /etc/tomcat7/tomcat-users.xml
# add roles "manager-gui" and "admin-gui" to account to allow
# access to manager web app and host-manager web app, respectively
http://aws.url:8080/manager/html
http://aws.url:8080/host-manager/html

 # startup tomcat

sudo /etc/init.d/tomcat7 start

 # default tomcat homepage

/var/lib/tomcat7/webapps/ROOT/index.html

 # we configured Tomcat to run with more memory than the default

create: /usr/share/tomcat7/bin/setenv.sh with
export JAVA_OPTS="-server -Xmx2G"

Script Management

/etc/init.d/virtuoso-opensource-6.1 - start, stop and restart script for Virtuoso

/etc/virtuoso-opensource-6.1/scripts/vload - loads any RDF file into a specified Virtuoso graph

/etc/virtuoso-opensource-6.1/scripts/vdelete - deletes and RDF graph in Virtuoso