Experiment Configuration for GridFTP-SDN

Zhe Zhang

November 5, 2015

Abstract

This document illustrates the detailed work flow in steps to re-produce the GridFTP SDN experiment. This includes the packages that need to be installed, configuration files need to be modified and all the tricky details about the experiment that need to be taken care of The goal of this documentation is to make future experiments repetition much easier.

1 HTCondor

1.1 Installing HTCondor

There are two main branches in HTcondor repositories that I am working on. `network_namespaces` is corresponding to the "Lark" project and `network_negotiator` is corresponding to the more recent network assisted HTCondor matchmaking.

To checkout the `network_namespaces` branch do:

```
git checkout network_namespaces
```

The latest version that integrates with network namespace support is 8.3.2 (This has the default route bug fixed. Previous version of Lark codes cannot add default route to the inner private network namespace veth device, thus making it possible for jobs running in that private network namespace to communicate with outside network). Instead of using the old `fedpkg-condor-hcc` tool to build the RPM, the new work flow to build the RPM is in the following:

```
cd $HOME/htcondor
rpm -i build/packaging/srpm/condor-8.3.2-0.1.5fd9bb7.git.el6.src.rpm
rpmbuild -ba  /rpmbuild/SPECS/condor.spec
```

Of course the source RPM should be replaced with the one you build. Useful commands for checking installed RPM and install/uninstall RPM:

```
sudo rpm -qa | grep condor
sudo rpm -ev --nodeps $ALL_THE_CONDOR_RELATED_RPMS
sudo yum install $ALL_THE_CONDOR_RELATED_RPMS
```

1.2 Configuration

After HTCondor with network namespace support is successfully installed, we need to configure HTCondor to make it work correctly.

For central matchmaker: remember to disable (comment out) the `condor_config.local` and the example configuration files can be found at the `application-aware-sdn-module` github repository under directory `experiment/`. It also includes some of the configuration for `network_negotiator` work.

For worker nodes: corresponding configuration files can also be found at the directory mentioned above.

1.3 Submission Script

The experiment has two HTCondor jobs, each of them belongs to a different user. (e.g. `zzhang` vs `larkuser1`). To submit condor job script under another user do: `sudo su- $USERNAME`, which will take you to the home directory of the user.

The submission script is `ftp_upload_submit_1` (also in github repo). The only thing needs to be changed

is the FTP server IP address in the argument list. These two users upload files to different FTP servers.

After two jobs are submitted, we can use `condor_release` to release the hold for the two jobs.

Useful command: `fallocate -l 10G gentoo_root.img` quickly generate file with arbitrary size.

1.4 Debugging

There could be problems running these HTCondor jobs.

- Make sure that the indicated lark plugin path is correct and can be loaded.
- Check HTCondor log to make sure DHCP offer is received, if not use `tcpdump` to debug: `tcpdump -vnes0 -i eth0 port 67 or port 68` It is possible that DHCP server does not have dynamic addresses.
- When initially there is no DHCP lease cache available, the current SDN controller code seems to block the private network namespace from receiving the DHCP packet. (This needs to be further examined and debugged.) The work around is to first run HTCondor lark job without SDN enabled to get a DHCP offer lease cache, then run it again with SDN enabled, HTCondor can use the DHCP offer in the cache. Directory is `/var/lock/condor/lark/dhcp_leases/`
- There is a MACRO to disable network namespace support for file transfer: `ENABLE_NETWORK_NAMESPACE_FOR_FILETRANSF` by default it is `TRUE`. Currently the network namespace integration with file transfer is not good enough, we need to set this MACRO to be `FALSE` explicitly to make it work.

2 GridFTP

Currently, `hcc-lark02.unl.edu` is used as the GridFTP server together with HDFS installed as the underlying file system.

2.1 Installed Packages

- GridFTP server: `sudo yum install globus-gridftp-server globus-gridftp-server-devel`
- Hadoop-HDFS: `sudo yum install hadoop-hdfs`
- HDFS datanode and namenode: `sudo yum install hadoop-hdfs-namenode & sudo yum install hadoop-hdfs-dat`
- globus-xio-devel: `sudo yum install globus-xio-devel`
- Compile xio_callout and install it
- Compile gridftp_hdfs and install it
- Install library libtool: `sudo yum install libtool`
- When compile gridftp_hdfs, go to the home directory gridftp_hdfs, not the one inside xio_callout
- Install `java-openjdk-devel`
- Install `openssl openssl-devel`
- Set up configuration files like: `core-site.xml` and `hdfs-site.xml`, the parameters of these two files can be found in hadoop website, and need to change the indicated directory's user and group to be `hdfs:hadoop`, otherwise datanode and namenode cannot access them. Need to format namenode. (When format, need to run the command `hadoop namenode -format` under hdfs user) `sudo -H -u hdfs hadoop namenode -format` useful link: `http://hadoop.apache.org/docs/r1.0.4/cluster_setup.html`
- need to create `hadoop-env.sh` in directory `/etc/hadoop/conf` and within `export JAVA_HOME = dir/to/jdk`
- install `globus-gridftp-server-progs`

2.2 Server Side Execution

First, run command `source /usr/share/gridftp-hdfs/gridftp-hdfs-environment` need to modify `JAVA_HOME` `HADOOP_HOME` environment variable, also make sure to include all the necessary jar files inside of `$HADOOP_HOME`.

Command to run GridFTP server (don't run it with sudo which would clear the environment variables set up via the above command): `/bin/sh -c 'LD_LIBRARY_PATH=/usr/lib/jvm/jre-1.7.0/lib/amd64/server/:/usr/lib/jvm/jre`
`globus-gridftp-server -c /etc/gridftp-hdfs/gridftp.conf -C /etc/gridftp.d -log-level ALL -control-inter`
`129.93.241.12 -p 5000 -password-file /home/bockelman/zzhang/pwfile'`

Command to generate password for user and write to password file: `globus-gridftp-password` The experiment for WAN GridFTP file transfer is from the same user , but to access files in different directories. (Even though it should be different users)

Overall need to make sure `JAVA_HOME`, `HADOOP_HOME` and all the jar files are correctly set up and included. It is possible the version of `HDFS` or `Hadoop` installed is not compatible with `GridFTP-HDFS` installed. Thus we need to make sure to update the configuration in `GridFTP-HDFS` for the right version of `Hadoop`.

2.3 Client Side Execution

At client side, use command `globus-url-copy` to do file transfer. (Client downloads file from GridFTP server) `sudo apt-get install globus-gass-copy-progs`.

Command to download file from GridFTP server:

```
globus-url-copy -v ftp://zzhang:passwd@hcc-lark02.unl.edu:5000/tmp/group file:///etc/group
```

2.4 Useful Commands

Copy from local file system to HDFS `hadoop fs -put /tmp/test.img /test3/test.img`

2.5 Useful Links

For GridFTP-HDFS:
`https://twiki.grid.iu.edu/bin/view/Documentation/Release3/InstallHadoopSE`
`https://twiki.grid.iu.edu/bin/view/Documentation/Release3/InstallHadoop200SE`

For GridFTP administration:
`http://toolkit.globus.org/toolkit/docs/latest-stable/gridftp/admin/index.html#gridftp-config-overview`

3 OpenFlow SDN Controller

Detailed README doc can be found at github repo `application-aware-sdn-module`. The command to execute the implemented SDN OpenFlow controller is:

```
./pox.py log.level --DEBUG application_aware_switch proactive_sdn_module
```

The command that can be used to show the mapping between numerical port number and Ethernet device name: `ovs-ofctl show <br>`, to show the datapath number: `ovs-dpctl show <br>`.

To apply Open vSwitch QoS rate limiting with different QoS queues, use the implemented script `ovs_qos_setup.py` and correspondingly `ovs_qos_cleanup.py` is used to clear all the QoS queues. Sometime, Open vSwitch returned with errors when cleaning up because of some existing references, run this command to fix before execute the implemented `ovs_qos_cleanup.py` script: `ovs-vsctl clear qos eth0 queues`, `eth0` should be replaced with the actual port that has QoS queues configured.

`sdn_controller.cfg` is the configuration file to indicate network policy (application_oriented vs project_oriented) and QoS bandwidth rate limiting numbers, etc.

4 FTP Server-vsftpd

To make it simple, we enable the anonymous user access for vsftpd server. The example configuration file is `vsftpd.conf` in the github repo. On FTP server: do `sudo yum install vsftpd` or `sudo apt-get install`

vsftpd, on client do sudo yum install ftp (then client can use curl to upload and download). On Ubuntu, the file directory for FTP server is: /srv/ftp, on CentOS it is /var/ftp.

Disable iptables first to make sure firewall does not block traffic.

To make sure anonymous user can upload file, remember to set selinux to be permissive. If you modify the configuration file at /etc/selinux/config, you have to restart to make it take effect, otherwise you can type "setenforce 0" to make it take effect immediately. Another possible solution is use: setsebool -P ftp_home_dir 1 This commands makes selinux to allow ftp and at the same time Selinux is not disabled. It's not recommended to disable Selinux.

Also need to make sure /var/ftp to be owned by root:root, (of course this is by default), if use command it is like: chown -R root.root /var/ftp/, after that you also need to make sure /var/ftp/pub is set to be 777 by using chmod.

The best practice is to create a pub directory under /var/ftp or /srv/ftp, change its permission to 777. Then the corresponding command to upload to that directory is: curl -4 -T 5GB_1.zip ftp://128.104.222.79/pub/ --user anonymous: Notice that the pub directory is used immediately after the server IP address. Basically / means the root directory for vsftpd server, which is /var/ftp or /srv/ftp depending the Linux distributions used.

5 Ganglia

The example Ganglia configurations files are: gmond.conf (for client) and gmetad.conf for server. Both of these are included in the github repo. And there is a very clear tutorial to follow for setting up Ganglia: https://www.digitalocean.com/community/tutorials/introduction-to-ganglia-on-ubuntu-14-04

Since we are using CloudLab for experiment, we can request for a dedicated Ganglia server node and install Ganglia client on the two GridFTP client nodes. The GridFTP related file transfer rates can be shown at the Ganglia server.

6 Generating Experiment Plots

After the experiments, CSV files can be collected from Ganglia and we can use plot_generate.py to draw the figures for these experiment. The actual scripts and corresponding CSV files the scripts operate on are included in the github repo for reference.