

Bridge Globusonline Into Condor and Glidein WMS

Xi Duan
Illinois Institute of Technology

1 Introduction

Globusonline[1] is a data moving tool over specified network from one endpoint to another, which is developed by University of Chicago and Argonne National Laboratory. A lot of its users are from HPC community due to the underneath network support are from open science grid partners. Users have access to those clusters can make use of the high-speed links to transfer large-size files, and it provides multiple user interfaces including web interface which is very easy to understand and use so that it doesn't require a computer technician to operate on it. Aside from the feature of high-speed, it is reliable tool to transfer file in terms of fault tolerance.

Condor[2] is a long-term-support and well-developed workload management system, it is developed by University of Wisconsin. It can organize resources, allocate, distribute and reclaim them for user tasks based on certain criteria customized by users. Condor is a famous and popular tool in the HPC community as well because it provides hundreds of features and functions to meet the requirement of users, moreover, most of its features are mature and stable.

Glidein WMS[3] is a glidein-based workflow management system distributed on grid resources developed by Fermilab. It is built on Condor, thus it has lots of common attributes with Condor. The purpose of Glidein WMS makes use of shared grid resources among different virtual organizations in Open Science Grid[4] (e.g. Fermilab, ANL, Universities etc). With Glidein WMS, resources are aggregated and redistributed in a better and balanced way. This tool is also widely used in the community.

Owing to the limitation of job input/output files in Condor that those files can only be transferred from/to submit node – the link between submit node and work node is challenged. To remove the limitation and improve the performance and efficiency, Condor team made file transfer plugin interface at version 7.4.0[5] and enabled output file transfer plugin interface at version 7.6.0. With those fix, work node does not only get/send file from/to submit node, but also from/to other resources, like http, ftp resources. The purpose of this work is to build and test the plugin for Condor to transfer file between Globusonline resources and Condor work nodes, due to they are both popular tool in the HPC community. Therefore, we hope that with this plugin, files that reside on Globusonline resources can find its way to Condor-manage clusters or the Glidein work pool.

2 Globusonline transfer plugin for Condor

The plugin is written in python, according to the standard of the interface provided by the Condor team. Also, the plugin utilizes globusconnect[6], which is a command line tool to access Globusonline nodes, to do the connection and transmission jobs. We made the same interfaces and functionalities shown in the next section.

2.1 Functionalities

In total, there are 3 flags which can be utilized, and 2 of them required and used by condor, the rest 1 is for test purpose:

- Query Mode. To activate this mode, start the plugin with flag `-classad`. This mode makes condor to invoke the plugin with the `-classad` option to ask for information. Here is an example of it:

```
[xduan@fermicloud049 ~]$ ./globusonline_plugin.py -classad
PluginVersion = "0.1"
PluginType = "FileTransfer"
SupportedMethods = "globusonline,file"
```

figure 1: classad flag example

- Action Mode. This mode is what the plugin really does the job. In this mode, 2 arguments need to be provided: one is the source (the file to be transmitted), and the other is the destination (where does the file go). The following picture is an example of it:

```
[xduan@fermicloud049 ~]$ ./globusonline_plugin.py globusonline://xduan:teststep:/tmp/xferfile /
home/xduan/xferfile
```

figure 2: action mode example

Note that the format of globusonline endpoint file should follow:

globusonline : // < GUsername > : < GOendpoint > : < GOfilepath >

Where GUsername is the login name of Globusonline, GOendpoint is the identifier of user endpoint that the user created on Globusonline, and GOfilepath is the object file/directory path on the endpoint for input/output.

The local file could be used as a common way or begin with a file identifier:

file : // < filepath >

- Test Mode. This mode is not required by condor, but developers can use this mode to debug or test. With the flag of `-test` followed by an globusonline endpoint file with the format mentioned above, one can activate this mode. Here is an example with result:

```

[xduan@fermicloud049 ~]$ ./globusonline_plugin.py -test globusonline://xduan:teststep:/tmp/xfer
file
=====
Running Tests
=====
TEST: Plugin information ...
PluginVersion = "0.1"
PluginType = "FileTransfer"
SupportedMethods = "globusonline,file"

TEST: Setting up the Globus Connect Service ...

TEST: Status before starting ...
{'Globus Online': 'disconnected'}

TEST: Starting Globus Connect ...

TEST: Status after starting ...
{'Globus Online': 'connected'}

TEST: Creating a local file

TEST: Transfer from local to GO

TEST: Transfer from GO to local

TEST: GO endpoint remove

TEST: Stopping Globus Connect ...

TEST: Status after stopping ...
{'Globus Online': 'disconnected'}

TEST: Remove endpoint ...
=====

```

figure 3: test flag example

- Log. A log file is generated and updated locally on the work node. This log includes all the debug or error information when this plugin is running. It is useful for debugging if something unusual happens. If the user specifies in the job submit file, log can also be transfer back to submit node.

2.2 Plugin Configuration for Condor

The configuration in condor is quite easy, just mention the following in the configuration file of executor work machines:

```
FILETRANSFER PLUGINS = < Path of the plugin >
```

One thing that is very important needs to be pointed out the plugin uses gsissh to login globusonline. In order to get gsissh work, the user should do the prerequisites software installation and configuration. The first thing is the user has to obtain a valid certificate and key which can be authorized, for example, fermilab KCA or DOEgrids certificate and key. Then VDT is required, because trusted CA and utilities to generate proxy is provided by VDT (usually we use voms-proxy-init or grid-proxy-init). The last thing is get the environment set, the following picture is an example:

```

export X509_USER_PROXY=/tmp/x509up_u0
export X509_CERT_DIR=/home/xduan/.globus/certificates/
export X509_USER_CERT=~/.globus/usercert.pem
export X509_USER_KEY=~/.globus/userkey.pem

```

figure 4: environmental configuration file example

Where X509_USER_PROXY indicates the proxy generated by VDT utility, X509_CERT_DIR is set to the path of trusted CA, X509_USER_CERT is the valid user certificate and X509_USER_KEY is the valid user key. One more thing is user has to set flag DELEGATE_FULL_JOB_GSI_CREDENTIALS to True to enable job credentials delegation.

Also, if condor components are installed on different nodes, the user also has to set user proxy, certificate mapfile, trusted CA directory and so on in order to get different components on different nodes can talk to each other. However, this part is out of the scope of this report, and can be found in condor documents, we don't introduce it here.

For more information and help, use the -help flag to retrieve help information.

3 Performance Test

This test aims to find out how the Globusonline plugin for Condor influences the transmission stage. For a common job in condor, job executable file and data are submitted from the submit node, which is good when the transfer files are not big so that it won't introduce a lot network traffic. However, when the amount and size of the files are large and big, it is not a good method to get it from submit node since the network card or link of submit node would become a bottleneck. In such case, we need to fetch file from some other sources instead of submit node, for example, http server, ftp server, etc. The plugin aims to enable Condor work node access the resources on Globusonline sites for better aggregated throughput. One can register and hold a server on globusonline and others could access the data on that server if authorized and authenticated. Once the connection is established, we can enjoy high-speed bandwidth from globusonline.

The test is conducted in the way of running the plugin alone. Because there are some setup (authorization, authentication, connection establishing etc) before data transfer, we have to watch the overhead of this setting stage, or if there are some performance degradation via the plugin (e.g. connection error might happen during transmission which has impact on performance) compared with via regular transfer tasks. Moreover, we expect the regular transfer tasks have better performance than the plugin tasks do due to the setup overhead. The test is held on a virtual machine as the client: it has double QEMU Virtual CPU with 2660 MHz processors, 2 Gigabytes memory and 100Mbit/s NIC connection speed, running Scientific Linux Fermi 5, kernel 2.6.18-238.19.1.el5. On the other end, the configuration of the server is: double Intel(R) Xeon(R) CPU X5355 processors, 2 Gigabytes memory and 100Mbit/s NIC ,running Scientific Linux Fermi 5, kernel 2.6.18-238.12.1.el5xen. The regular tasks are using "globus-url-copy" command due to the test server needs certificate to access, but it is actually not much different than "scp". For the plugin tasks, we run the plugin stand alone, which is written in python, fetch the target file from the server to local. The measurement is transfer time, from the starting of task to the end. The tasks are ran 10 times, and the result is the average value. Here, in the following picture, we get the performance of both set of tasks:

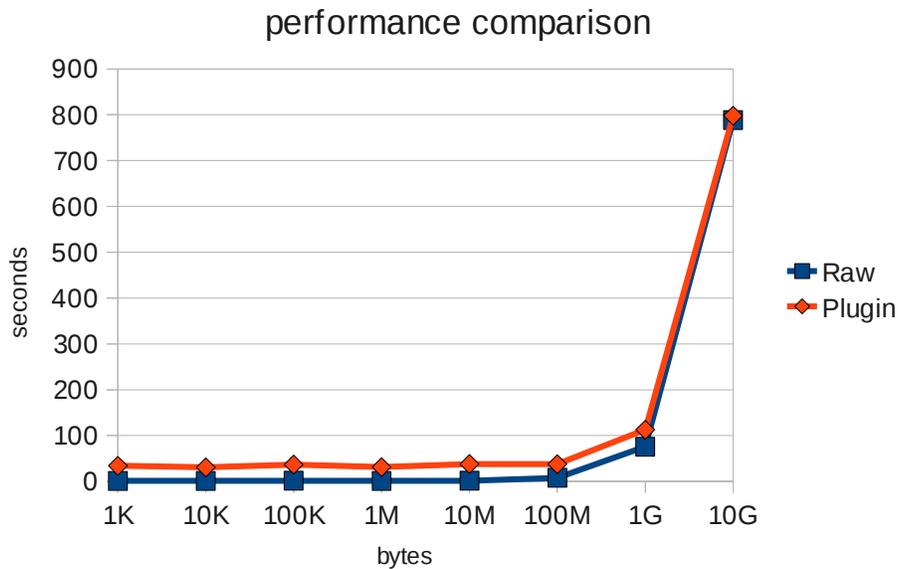


figure 5: performance comparison

The line with legend “Raw” refers to regular transfer task, and the line with legend “Plugin” stands for Globusonline plugin running alone. From this picture, we can observe there are performance gap in the less significant files, but the plugin tasks catch up in the large file test. We notice that the plugin always has 30 seconds overhead in setting up the connection. While transferring small file, it costs 1 second more or less for the regular transfer tasks, but that is 30 and more seconds for the plugin transfer tasks. However, it is not clear in this picture how the gap is in the early test. So here comes the next chart showing how these gaps shrink.

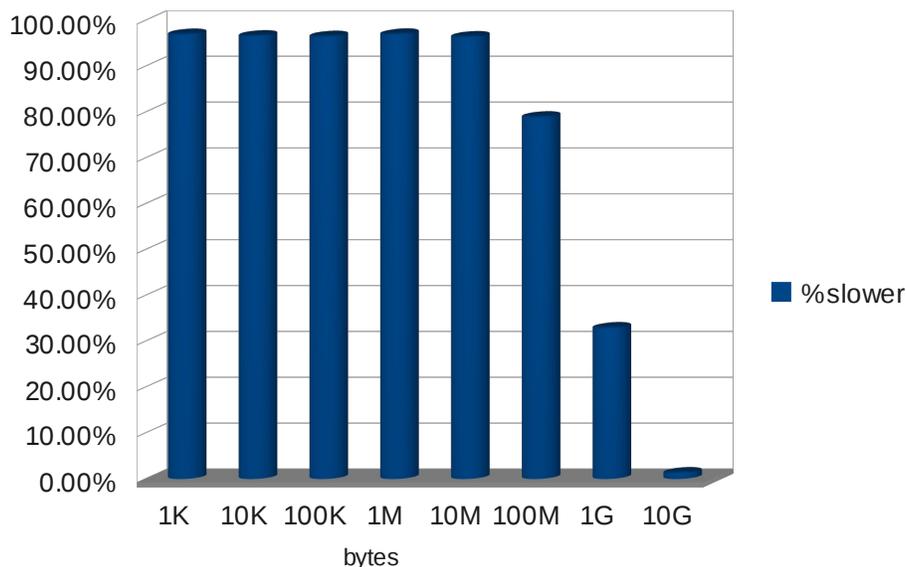


Figure 6: How much slower the plugin is?

In figure 6, the bars refer to the percentage that the difference of running time on the running time of plugin tasks, which is calculated by the formula of:

$$T = T_{plugin} - \frac{T_{regular}}{T_{plugin}}$$

Where T measures the result, T_{plugin} indicates the running time of plugin, and $T_{regular}$ is the running time of regular transfer.

From figure 6, we notice that the gap is huge – nearly 100%, but later on, they get peer performances. The reason is the constant setup overhead, when the file becomes larger, the transfer time becomes longer and longer, and the setup overhead becomes less and less significant. Moreover, plus the uncertainty of network traffic, the plugin transfer tasks get better chances to get close to the performance of regular transfer tasks, or even beat it.

4 Put it in Glidein WMS

4.1 Steps of configure and install Glidein WMS

Configuration and installation of Glidein WMS are complex due to it has much to do with certificate/key or proxy, and it has several components that need to be understood before the installation. Although a detailed tutorial is given at the Glidein WMS website, I would like to talk about the installation steps and list some pitfalls that I met.

Firstly, the components location problem. One who is new to those kind of tools including Condor would ask: should I use one node for all those components or should I use each node for each component? My suggestion is to use the latter way if you want a better understanding of those tools. Glidein WMS is much more complicated than Condor, in that sense, separated components are better. Before getting started, one should always finish installing the prerequisite software, and get the certificate/key or proxies ready. For certificate/key or proxies, one should prepare both host certificate/key pair for communication between components and a DOE certificate/key pair or KCA certificate/key pair for accessing Fermilab machines.

Next step is installing various components, as the guide mentioned, the sequence should be: WMS collector, factory, User collector, User submitter, Condor for frontend and frontend. The description of those steps is very straightforward and examples are provided, here I just provide something that needs to pay attention. The first one is to use host certificate/key pair instead of personal. This is quite helpful to understand how those components talk to each other, because in the configuration stage, each component's certificate/key is required to provide, different nodes have different host certificate/key, it's easy to know how many components and what they that one component talks to. Also, it is the reason I suggest to use one node for one component. The second thing is to put the installation folders of Condors, factory, and frontend in local folder, never put factory and frontend in a NFS, which will cause a bug that NFS is not supported for file lock. The last thing is to use personal certificate/key or proxy to submit jobs instead of host certificate/key or proxy due to the latter are usually unauthorized, failure will return.

The steps to enable the plugin on Glidein WMS is not complicated, aside from the configuration steps in Condor, we need to add plugin information in frontend's configure file. This can be found on the plugin webpage wiki[7].

5 Future Work

The plugin is implemented for the linux interface for Globusonline, for windows and mac os, we haven't done any work due to the Globusonline connect tool globusconnect commend line, it doesn't have enough feature or flag to implement the plugin. Once the Globusonline team release newer and completed globusconnect tool, we can start on those projects.

6 Conclusion

In this work, we build a transfer plugin to bridge Globusonline and Condor, and enable Condor to utilize the convenience provided by Globusonline. Although it requires extra time to setup the connection before file transmission, it frees submit node's network I/O, which makes sense when submit node's network I/O is limited and multiple identical large files need to be transferred. We did experiment to evaluate the performance of the plugin and happily found it is acceptable. Moreover, we also apply the plugin on Glidein WMS, enable the whole grid get access to the files on Globusonline endpoints.

7 Reference

- [1] Globusonline, <http://www.globusonline.org>, 2010-2011 Computation Institute, University of Chicago, Argonne National Laboratory
- [2] Condor, <http://www.cs.wisc.edu/condor/>
- [3] Glidein WMS, <http://www.uscms.org/SoftwareComputing/Grid/WMS/glideinWMS/doc/prd/index.html>
- [4] Open Science Grid, <http://www.opensciencegrid.org/>
- [5] Flexible Data Placement Mechanisms in Condor, <http://www.cs.wisc.edu/condor/CondorWeek2011/presentations/zmiller-cw2011-data-placement.pdf>
- [6] Globusconnect, https://www.globusonline.org/globus_connect/
- [7] go-condor-transfer-plugin, <https://cdcvs.fnal.gov/redmine/projects/go-condor-plugin>