JAVA based portable GRID storage access tools installation (no root access required)

This page describes the procedure to set up the tools for working directly on the srm storage. You may want to consider using the staging and download services provided by Astron if you just want to retrieve data from the archive: http://www.lofar.org/wiki/doku.php?id=public:lta_howto.

For the impatient: skip text and jump to the installation walk-through. (Assumes you have a grid certificate available and have been placed in the VOMS /lofar/user group.)

For further reading and a non-portable installation you might be interested in SRM software installation.

Installation of grid middleware used to be quite challenging although their is now often reasonable support via linux installation packages (see e.g. the page on SRM software installation. It is however possible to deploy the required client tools without the need for root access and reasonably independent of the OS, The setup below has been tested on Kubuntu 12.10 and Mac OS X 10.6.8. Since all programs are JAVA based, it is very likely that these packages can be made to work on other systems as well. However, the tools would require customized scripts as provided in the (s)bin directories.

General requirements

Given the title of this page, it is obviously necessary to start with a system with a JAVA VM installed (1.6 or higher). NB Since there is a known issue for the dCache/FNAL SRM client tools on openjdk-7, it may be required to install an alternative JVM than the system default and point `JAVA_HOME` to the alternative location.

The key elements in any minimal installation are:

- A valid (personal) grid certificate.
- Functionality to generate a proxy with VOMS attributes
- Functionality to initiate data transfers using SRM and the gsiftp protocol

Below, we will take a detailed look at each individual element.

Note that you only have to install your grid certificate yourself and can then simply use our prepackaged tarball for the rest.

Grid certificate

The procedure for obtaining a personal grid certificate is described here. The default locations for the private key and the signed certificate are respectively `$HOME/.globus/userkey.pem` and `$HOME/.globus/usercert.pem`. Alternative locations may be used but should then be configured by setting appropriate values for the `X509_USER_KEY` and `X509_USER_CERT` environment variables.

NB It is not required to store the personal certificate on the system that has the SRM clients installed. These clients only require a valid 'proxy' generated from the certificate and it is perfectly possible to generate the proxy elsewhere and copy it to the SRM client system using e.g. `scp` or 'MyProxy' (to be further documented).

Note on security: By whatever means the certificate and/or proxy are generated, it is critical that the private key (stored as `userkey.pem` or in any other form) and the generated proxy (see below) are kept secure. These files should have set read/write access for only the owner (e.g. `chmod 600 userkey.pem`) and it is good practice to be very restrictive regarding the creation of copies, in particular on shared systems. In addition, the private key should always be secured by setting a (good) encryption password when it is generated.

Proxy generation

The `voms-proxy-init` tool from the VOMS client tools can be used to generate a proxy with VOMS attributes from the personal certificate.

Note that the previouly provided `proxy-init.sh` script from jLite is now discouraged because its encryption strength of 512 bit is not considered sufficient any more. SRM sites require a minimum strength of 1024 bit.

To allow usage of the LOFAR VO (Virtual Observatory), there are three additional steps to take:

- Create a `vomses` file to allow `voms-proxy-init` to contact the relevant VOMS server
- Create a file that contains the certificate chain for authentication of the VOMS server
- Install the set of trusted grid Certificate Authority (CA) certificates (these are also required by the SRM tools).

The 'vomses' file

To allow generation of proxies with LOFAR voms attributes, a 'vomses' file must exist that contains at least the following line:

```
"lofar" "voms.grid.sara.nl" "30019"
"/O=dutchgrid/O=hosts/OU=sara.nl/CN=voms.grid.sara.nl" "lofar"
```

Place the file in a location where the proxy generator (voms-proxy-init) can find it (e.g. ~/.glite/vomses or export custom location to $X509_VOMSES variable). Alternatively, you may point voms-proxy-init to any custom location by the '-vomses' option.

VOMS server certificate chain

To allow the proxy generator to check the authenticity of the voms server, the chain of certificates for the the VOMS server must be provided in a file named '<voms_server_address>.lsc'. This file must be placed in a subdirectory with the name of the virtual organization which is placed in the VOMS directory (see below). For the lofar VO, this file is `lofar/voms.grid.sara.nl.lsc` which should contain the following lines:

```
/O=dutchgrid/O=hosts/OU=sara.nl/CN=voms.grid.sara.nl
/C=NL/O=NIKHEF/CN=NIKHEF medium-security certification auth
```

The VOMS directory location can be configured by setting the `$X509_VOMS_DIR` environment value.

Trusted grid CA certificates

If your system has been installed with grid software already, it is likely that the trusted CA certificates can be found in the default directory `/etc/grid-security/certificates`. If this is not the case, or you receive error messages related to not being able to determine the authenticity of a certificate, you can retrieve the latest certificates, e.g. from EUGridPMA, and place them in a directory of your choosing (set environment variable X509_CERT_DIR accordingly). If you use our prepackaged tarball, you can use the provided script 'update_certificates.sh' (for bash; use the .csh version for c-shell) to deploy the latest certificates (Note: The script requires the lynx text browser to be installed on your system).

SRM client tools

There are (at least) two JAVA based packages that provide the required SRM client functionality, including support for the gsiftp (a.k.a. 'gridftp') protocol. One has been developed by Fermilab and is hosted by dCache. Another has been developed at Berkeley.

NB If the client srm copy call returns timeout messages, the most likely cause is that a firewall is blocking outward connections. The following ports are typically needed by the srm client tools: 8443/8444, 2811, and any port in the gridftp port range (typically in the range 20000 - 25000). Note that these ports are configured on the server side so this list may not be complete for all situations. If at all possible, it is advisable to configure the firewall to allow all outward connections. The next best option is to allow all outward connections to the domains that provide LOFAR LTA services (currently grid.sara.nl, fz-juelich.de, and target.rug.nl)

NB2 If you want to enable 'active' transfers, firewalls should allow incoming access to the ports configured as the globus port range (look in client documentation for more details). This type of transfer can improve performance of a single transfer as it will use multiple parallel connections for retrieving a file. For the FNAL/dCache client, 'active' transfers are initiated if the `-server_mode=passive` setting is omitted. For the Berkely client, parallel transfers will be initiated when the `-parallelism` parameter is set to a value larger than 1. Since in most cases LOFAR datasets consist of a large number of files, a similar performance improvement can be achieved by splitting the set of files over multiple srm copy processes.

FNAL/dCache client tools

We provide a slightly modified package in two different versions: srm client tools (Java 7) (or alternatively srm client tools (Java 6)). (Note that both are also part of the prepackaged tarball together with an init script to set up the environment.) These contain updated scripts that allow

installation in any directory. The client can be unpacked anywhere after which the $SRM_PATH environment variable should be set to the root directory of the unpacked tarball. Additionally, the $SRM_PATH/bin directory should be added to $PATH and trusted CA certificates must be installed. Finally, a valid proxy must be available in the default grid location (/tmp/x509up_u<UID>) or in the location configured as $X509_USER_PROXY. You should now be able to retrieve a file from the LOFAR LTA:

```
srmcp -server_mode=passive
srm://srm.grid.sara.nl/pnfs/grid.sara.nl/data/lofar/ops/fifotest/file1M
file:///file1M
```

If your firewall allows incoming connections to non-standard ports, you can try this command without the server_mode option which will enable utilization of multiple streams to increase performance.

If you have the gridftp client software installed (requires installation with root privileges) and in your path, it provides superior performance as compared to the native JAVA gridftp client that is provided by srmcp. In order to utilize this, download lta-url-copy.sh.gz, unzip it and use the command:

```
srmcp -use_urlcopy_script=true -urlcopy=./lta-url-copy.sh -
server_mode=passive
srm://srm.grid.sara.nl/pnfs/grid.sara.nl/data/lofar/ops/fifotest/file1M
file:///`pwd`/file1M
```

Note: Usually, the url-copy script requires an absolute destination path. Make sure to use our modified script (named /lta-url-copy.sh/, now also included in the tarball below) to retrieve data with the /srm.txt/ file.

Prepackaged tarball

We provide the Lofar Grid Clients tarball that comes prepackaged with the above (except your personal grid certificate of course) to allow easy access to the LOFAR VO. Extract the tarball to a directory of your liking and source 'init.sh' if you use Java 7 (or newer) or source 'init_java6.sh' if you still use Java 6. This sets up the environment for you. You can `voms-proxy-init` for generating a proxy and `voms-proxy-info` for inspecting the generated proxy.

Walkthrough

This walkthrough guides you to setting up the prepackaged tarball (containing the FNAL/dCache SRM client tools) on your system. It assumes a bash-like shell.

- Store your private key in `$HOME/.globus/userkey.pem`
- Change private key permissions to owner-read-only:

  ```
  chmod 400 $HOME/.globus/userkey.pem
  ```

- Store your signed certificate in `$HOME/.globus/usercert.pem`

- Download the Lofar Grid Clients
- Untar package in directory of your choosing:

```
tar -xvzf lofar_grid_clients.tar.gz
```

- Determine your java version:

```
java -version
```

- Source init.sh (Java 7) or init_java6 (Java 6) in lofar_grid/, e.g. :

```
. lofar_grid/init.sh
```

- Optional: Set proxy environment variable to custom location:

```
export X509_USER_PROXY=<proxy_location>
```

- Generate a proxy:

```
voms-proxy-init -voms lofar:/lofar/user
```

- Test data retrieval:

```
srmcp -server_mode=passive
srm://srm.grid.sara.nl/pnfs/grid.sara.nl/data/lofar/ops/fifotest/file1M
file:///file1M
```

- Done!
  NB If you modified any default location by the `export` command, you have to put it in a shell start-up script like '.bashrc' to make your changes permanent, of course (with full paths where appropriate).
- If you get any errors related to CA certificates, retry after running one of the provided scripts to update your certificates, e.g.

```
. update_certificates_eugridpma.sh
```

Note: For (t)csh, use *.csh init scripts and 'setenv <key> <value>' instead of 'export <key>=<value>'.

Troubleshoot

- OpenJDK 7 seems not to be capable of dealing with the certificate, make sure to run Java provided by Oracle
- Error: *Unexpected reply: 426 Connection refused*
  - Make sure that you call srmcp with option *-server_mode=passive*
- Error: *org.glite.voms.contact.VOMSException: AC validation failed!*
  - Have you registered at the Lofar VO? –> https://voms.grid.sara.nl:8443/voms/lofar
  - You need to have the certificate installed in your browser for this (http://ca.dutchgrid.nl/info/browser)
- Maybe your private key uses an unsupported algorithm. You might want to try the following

command:

```
openssl rsa -des3 -in .globus/userkey.pem -out .globus/userkey.pem
```

- Once in a while, you probably have to update your CA certificates (see trusted CA certificates)
  - If this does not solve the issue, you should carefully check if there are multiple sets of certificates installed and which one is used by srm tools (check environment). If the certificate set in use was updated but is still broken, check for broken symlinks in the certificate directory (especially when using a Mac):

Bob mentioned this at some point:

```
  [The broken symlinks] are not symlinks anymore but just copies of the
original
  files instead (though I don't see why this matters...). All the files,
  except for the .r0 ones, in the certificate directory that have a filename
  containing some hash value, should actually be symlinks to files having a
  regular name.
```

Apparently, this is caused by some hack-happy developer at Apple. Hanno:

```
  When I tar a directory on my Mac it
  includes '._<somefilename>' entries that apparently are used by Mac OS
  to set file properties if this tarball is deployed on a Mac again. This
  appears to be particularly the case for symlinks and it looks like these
  mess up the certificate handling of the SRM clients on the lexar (maybe
  all linux systems?).
```

You can check for outdated crls like this (copy-paste from Bob):

```
  Regarding the CRLs / .r0 files: these all have a nextUpdate value, which
  basically is their expiration date. This date seems to vary a lot for all
  different certificate authorities: some are valid for only a couple of
  days, some for weeks, months or even a year. So how often you need to run
  the fetch-crl script mostly depends on the CA of both your own certificate
  and the certificate of the server(s) you are connecting to. For instance,
  my personal certificate is signed by TERENA eScience Personal CA; its CRL
  is only valid for a couple of days, after which all these tools start to
  complain about an expired CRL.
  You can check this yourself by finding the right .r0 file for your CA:
they
  also have these hash value filenames, so you first have to find which
  <hash_value>.0 points to <your_CA>.pem. Then you can use the following
  command to find when the CRL was last updated and has to be updated next:
  openssl crl -noout -lastupdate -nextupdate -in 169d7f9c.r0
```