Introduction

FroNTier

FroNTier is a distributed database caching system. Its protocol is HTTP-based and uses a RESTful architecture. FroNTier relies on a standard web caching tool, squid, to cache data accessed by clients.
In WLCG, ATLAS and CMS use FroNTier to allow jobs to access conditions data located at CERN. As hundreds of thousands of clients may need these data at any given time, local caching via squid makes access via FroNTier to scale up. Each ATLAS and CMS site is required to have a squid installation. If a site has at least some worker nodes with just IPv6 connectivity, having squid servers in dual-stack becomes a necessity.
The squid server needed for FroNTier is a patched version of squid-3, which supports IPv6, while this was not the case for squid-2. The difference with respect to the standard version is that it is preconfigured to work with the FroNTier caching system, some bug fixes and some additional features that make it easier to manage and monitor. It can be used by any other application, including CVMFS. It is conveniently distributed in OSG and in UMD (for EGI sites).

CVMFS

CVMFS is a software distribution service, implemented as a POSIX read-only file system in user space using a FUSE module. Files and directories are hosted on standard web servers. It uses only outgoing HTTP connections and uses aggressive caching to reduce network traffic and latency. CVMFS is used by all LHC experiments and all WLCG sites must have a squid server for the local caching, which may be shared with FroNTier (aside from scalability considerations). As for FroNTier, a dual stack squid server is needed if the site has at least some IPv6-only WNs (or prefers IPv6 for any reason). The squid server will contact a Stratum 1 server to retrieve data. The CERN Stratum 1 is in dual stack, therefore a dual stack squid server will communicate via IPv6 by default.

VM creation (optional)

If you need to create a VM yourself, and you have an Openstack account at CERN, you can do it by following these steps.

1. For the Manchester tutorial, install a CentOS7 VM as instructed
2. If you want to use the CERN virtualised infrastructure, create a RHEL7 VM supporting IPv6, for example using openstack.cern.ch (see https://clouddocs.web.cern.ch/clouddocs/networking/ipv6.html for more information).
3. All commands are to be run from a normal user account with sudo privileges.

Software dependencies and tools

Install the following packages (or download and compile from the source).

1.
```
sudo yum install telnet
sudo yum install net-tools
sudo yum install policycoreutils-python
sudo yum install 'perl(DBI)'
sudo yum install wget
sudo yum install gcc
sudo yum install gcc-c++
sudo yum install openssl-devel
sudo yum install expat-devel
```

4. Download and compile pacparser:
```
wget
https://github.com/pacparser/pacparser/releases/download/1.3.7/pacparser-1.3.7.tar.gz
tar zxvf pacparser-1.3.7.tar.gz
cd pacparser-1.3.7
make -C src
sudo make -C src install
cd
```

Frontier client

1. Download and unpack the Frontier client source:
```
mkdir frontier
cd frontier
wget
http://frontier.cern.ch/dist/frontier_client__2.8.20__src.tar.gz
tar zxvf frontier_client__2.8.20__src.tar.gz
cd frontier_client__2.8.20__src
make
export LD_LIBRARY_PATH=$PWD:$LD_LIBRARY_PATH
export PATH=$PWD:$PATH
cd
```

Squid

1. Install the latest version of the Frontier squid, which works also with CVMFS (see https://twiki.cern.ch/twiki/bin/view/Frontier/InstallSquid more more information):
```
sudo rpm -Uvh
http://frontier.cern.ch/dist/rpms/RPMS/x86_64/frontier-squid-3.5.24-3.1.x86_64.rpm
sudo yum install frontier-squid
sudo chkconfig frontier-squid on
```
2. Edit `/etc/squid/customize.sh` and change `cache_dir` to something that fits in the storage of your VM (e.g. 1 GB).
3. In the same file, add a value to `acl NET_LOCAL src` that will include the IPv6 subnet of the node (e.g. if the IP is `2a01:56c0:4033:1006::c224:842` you can add `2a01:56c0:4033::/48`)
4. Fix some possible issues with SELinux:
```
sudo semanage port -a -t http_cache_port_t -p udp 3401
sudo restorecon -R /var/cache
```
5. Add a firewall rule for IPv6:
```
sudo ip6tables -A INPUT -i eth0 -p tcp --dport 3128 -j ACCEPT
```

6. Start squid:
   ```
   sudo service frontier-squid start
   ```
7. (optional) If you make a change to customize.sh, you can start using the new configuration by doing
   ```
   sudo service frontier-squid reload
   ```
8. Check that squid is listening:
   ```
   telnet localhost 3128
   telnet <hostname> 3128
   ```
9. Test the installation with fnget.py:
   ```
   fnget.py
   --url=http://cmsfrontier.cern.ch:8000/FrontierProd/Frontier
   --sql="select 1 from dual"
   ```
10. Repeat the above (twice) after having set
    ```
    export http_proxy=http://<hostname>:3128
    ```
    and check that squid cached the response by looking at `/var/log/squid/access.log` where you should find an entry containing "TCP_MEM_HIT".
11. Set the environment for the Frontier client:
    ```
    export
    FRONTIER_SERVER1="(failovertoserver=no)(preferipfamily=6)(serverur
    l=http://cmsfrontier.cern.ch:8000/FrontierInt)(proxyurl=http://<ho
    stname>:3128)"
    ```
12. Run the Frontier client:
    ```
    cat > test_query.sql <<EOF
    SELECT COLUMN_NAME FROM CMS_COND_FRONTIER.POOL_OR_MAPPING_COLUMNS
    WHERE VERSION='automatic_default_for_EcalPedestals' ORDER BY
    POSITION
    EOF
    export FRONTIER_LOG_LEVEL=debug
    fn-req -f test_query.sql
    ```
13. Check that the client is indeed using IPv6 by looking for messages similar to
    ```
    debug [fn-htclient.c:276]: connecting to proxy
    vm1.wlcg8.esc.qmul.ac.uk
    debug [fn-urlparse.c:160]: getting addr info for
    vm1.wlcg8.esc.qmul.ac.uk, prefer ipv6
    debug [fn-urlparse.c:215]: ipv6test.cern.ch: found ipv6 addr
    <2a01:56c0:4033:1007::c224:84a>
    debug [fn-urlparse.c:228]: ipv6test.cern.ch: found addr
    <194.36.8.74>
    debug [fn-socket.c:51]: new socket s=3
    debug [fn-socket.c:110]: connect s=3 addr
    2a01:56c0:4033:1007::c224:84a waiting for response
    debug [fn-socket.c:149]: connected, s=3 .
    ```

CVMFS

1. Install CVMFS:
   ```
   sudo yum install
   https://ecsft.cern.ch/dist/cvmfs/cvmfs-release/cvmfs-release-lates
   ```

```
t.noarch.rpm
sudo yum install cvmfs cvmfs-config-default
```
2. Run the basic setup:
   ```
   sudo cvmfs_config setup
   ```
3. Create `/etc/cvmfs/default.local` with this content:
   ```
   CVMFS_REPOSITORIES=atlas.cern.ch
   CVMFS_SERVER_URL="http://cvmfs-stratum-one.cern.ch/cvmfs/@fqrn@"
   CVMFS_HTTP_PROXY="http://<hostname>:3128"
   CVMFS_IPFAMILY_PREFER=6
   ```
4. Get out of the way existing configuration files (it should not be needed but…)
   ```
   cd /etc/cvmfs
   sudo mv domain.d domain.d.BACKUP
   ```
5. Check if autofs mounts the specified repositories:
   ```
   sudo cvmfs_config probe
   ```
6. Restart autofs:
   ```
   sudo service autofs restart
   ```
   and do it again if you make a change in the CVMFS configuration
7. Try to access a repository:
   ```
   ls /cvmfs/atlas.cern.ch/repo
   ```
8. If this fails, check the configuration:
   ```
   sudo cvmfs_config chksetup
   ```
9. If you think that SELinux can be a source of problems, do
   ```
   sudo /usr/sbin/setenforce 0
   ```
10. After having fixed a problem, remember to do
    ```
    sudo service autofs restart
    ```
11. Check in `/var/log/squid/access.log` that the entries corresponding to the connections with the stratum one server are preceded by the IPv6 address of the squid node.

References

1. Frontier squid installation (https://twiki.cern.ch/twiki/bin/view/Frontier/InstallSquid)
2. CVMFS client installation (https://cernvm.cern.ch/portal/filesystem/quickstart)
3. CVMFS documentation (https://cvmfs.readthedocs.io/en/stable/)
4. Frontier squid test (https://twiki.cern.ch/twiki/bin/viewauth/CMS/SquidForCMS#Test_the_configuration)
5. Frontier client (http://frontier.cern.ch/dist/FrontierClientUsage.html)

Acknowledgements

Many thanks to Dave Dykstra for his invaluable input.

Appendix

For convenience, you can add these lines to `$HOME/.bash_profile`, setting all needed environment variables (replace <N> and <M> with the values that apply to your machine):

```
PATH=$PATH:$HOME/.local/bin:$HOME/bin:$HOME/frontier/frontier_client__2.8.20__src
```

```
export PATH
export LD_LIBRARY_PATH=$HOME/frontier/frontier_client__2.8.20__src:$LD_LIBRARY_PATH
export http_proxy=http://vm<N>.wlcg<M>.esc.qmul.ac.uk:3128
export FRONTIER_LOG_LEVEL=debug
export
FRONTIER_SERVER1="(failovertoserver=no)(preferipfamily=6)(serverurl=http://cmsfrontier.cern.ch:80
00/FrontierInt)(proxyurl=http://vm<N>.wlcg<M>.esc.qmul.ac.uk:3128)"
```