April 28, 2011

HCC's Use of Squid Caching

Derek Weitzel

Documentation

- Everything in this presentation, and even this presentation, is on the OSG twiki:

  https://twiki.grid.iu.edu/bin/view/Documentation/OsgHttpBasics

- Administrators wanting to install Squid can go to:

  https://twiki.grid.iu.edu/bin/view/ReleaseDocumentation/SquidInstallation

Application Need

- Open Mass Spectrometry Search Algorithm (OMSSA) from Nebraska Medical Center
- 22,000+ (short) Jobs per dataset, divided into 172 runs per dataset
- x45 Data sets = 961200 (short) jobs

Data Format

- 21MB for Per Dataset shared between datasets
- 83MB for Executables. Used in every job
- 172 Runs per Dataset

OSG Squid Setup

- User Documentation at: https://twiki.grid.iu.edu/bin/view/Documentation/OsgHttpBasics
- Every CMS & ATLAS site is required to have Squid (mandated by experiment)
- Environment variable: `OSG_SQUID_LOCATION`
- Applications use environment: `http_proxy`

Very Basic Usage

- Documented way
- Retries several times.
- Squid not always reliable

```
#!/bin/sh

website=http://google.com/

source $OSG_GRID/setup.sh
export OSG_SQUID_LOCATION=${OSG_SQUID_LOCATION:-UNAVAILABLE}
if [ "$OSG_SQUID_LOCATION" != UNAVAILABLE ]; then
  export http_proxy=$OSG_SQUID_LOCATION
fi

wget --retry-connrefused --waitretry=10 $website

# Check if the command worked
if [ $? -ne 0 ]
then
   unset http_proxy
   wget --retry-connrefused --waitretry=10 $website
   if [ $? -ne 0 ]
   then
      exit 1
   fi
fi
```

Squid (Python)

```
def main(argv):
        #set proxy if possible
        dataarchivefile, startat = _readparams(argv)
        osgsquid=os.getenv('OSG_SQUID_LOCATION')
        print "osgsquid is set to: "+str(osgsquid)
        hostname=socket.gethostname()
        print "hostname is set to: "+str(hostname)
        if osgsquid != None and string.find(osgsquid.upper(), 'UNAVAILABLE') < 0:
                os.putenv('http_proxy', osgsquid)
        elif string.find(hostname,purduematcstring) >=0:
                os.putenv('http_proxy', purduesquid)
        else:
                os.unsetenv('http_proxy')
        #download executables
        _downloadfile('wget '+basehref+commonexecsanddata_tar)

        #download comparisondata
        _downloadfile('wget '+basehref+dataarchivefile)
```

Squid Setup (Python)

- Check if the squid is available
- Set the Environment variable

```
if osgsquid != None and string.find(osgsquid.upper(), 'UNAVAILABLE') < 0:
        os.putenv('http_proxy', osgsquid)
elif string.find(hostname,purduematcstring) >=0:
        os.putenv('http_proxy', purduesquid)
else:
        os.unsetenv('http_proxy')
```

Squid Download (Python)

```
#download function
def _downloadfile(urloffile):
        succefulldownload = False
        print "using Proxy: "+str(os.getenv('http_proxy'))
        print "Max download trials: "+str(numwgetretries)
        for i in range(numwgetretries):
                print "Trial #: "+str(i)
                process=subprocess.Popen(urloffile, shell=True)
                process.wait()
                returncode=process.poll()
                if returncode==0:
                        succefulldownload=1
                        break
                sleepfor=(random.random())*180 # seconds
                print "Sleeping for %s seconds before retrying"%(sleepfor)
                time.sleep(sleepfor)
```

Squid (Python)

- Try the Squid multiple times
- Should sleep random time between iterations

```
print "using Proxy: "+str(os.getenv('http_proxy'))
print "Max download trials: "+str(numwgetretries)
for i in range(numwgetretries):
        print "Trial #: "+str(i)
        process=subprocess.Popen(urloffile, shell=True)
        process.wait()
        returncode=process.poll()
        if returncode==0:
                succefulldownload=1
                break
        sleepfor=(random.random())*180 # seconds
        print "Sleeping for %s seconds before retrying"%(sleepfor)
        time.sleep(sleepfor)
```

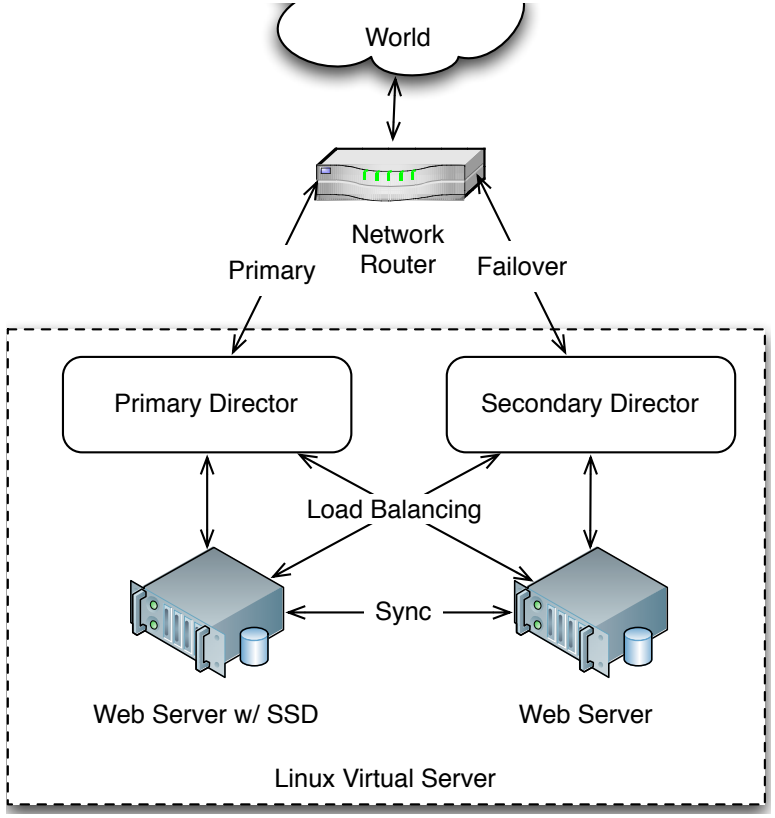
Squid doesn't work?

- Try directly a few times
- Sleep between tries.

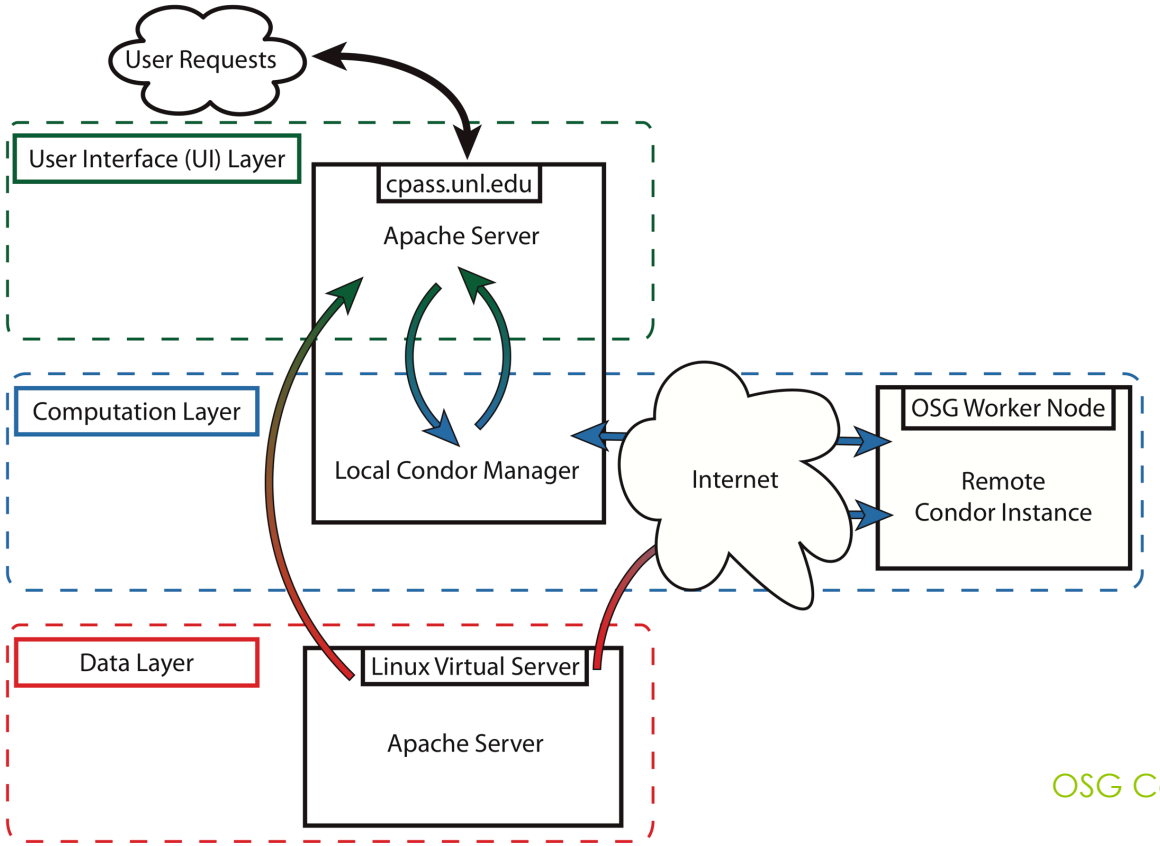
```
if not succefulldownload: #try unsetting the squid and going directly
        os.unsetenv('http_proxy')
        print "Not using using Proxy"
        print "Max download trials: "+str(numwgetretries)
        for i in range(numwgetretries):
                print "Trial #: "+str(i)
                process=subprocess.Popen(urloffile, shell=True)
                process.wait()
                returncode=process.poll()
                if returncode==0:
                        succefulldownload=1
                        break
                sleepfor=(random.random())*180 # seconds
                print "Sleeping for %s seconds before retrying"%(sleepfor)
                time.sleep(sleepfor)
```

Hardware Setup at UNL

- Failover with virtual IP
- Load balanced web servers
- SSD for serving small files
- 11 requests/sec - 0.6 GB/second - 58.2 MB/ request



CPASS use of HTTP/Squid



Additional Squid Details

Beware sites without Squid

- Sites (Pure OSG Sites):
  - Clemson
  - Cornell
  - UConn
  - OU OSCER
  - AGLT2
  - …
- Cause high load on your webserver
- Can hurt the site's network

Testing Squid

- Script on the OSG documentation page.
- Will test if the squid works
- And test if squid will cache your files.

```
#!/bin/sh

website=http://glidein.unl.edu/problems.txt

source $OSG_GRID/setup.sh
export OSG_SQUID_LOCATION=${OSG_SQUID_LOCATION:-UNAVAILABLE}
if [ "$OSG_SQUID_LOCATION" != UNAVAILABLE ]; then
  export http_proxy=$OSG_SQUID_LOCATION
fi

wget $website 2> /dev/null
wget -S $website 2> wget.err

grep "X-Cache: HIT" wget.err

if [ $? -ne 0 ]
then
   echo "Cache not working at $OSG_HOSTNAME"
else
   echo "Cache working"
fi
```

Securing Squid Usage

1. Pre-Compute checksums of files, append to global index
2. Checksum index
3. Pass checksum of index to the job securely (argument to job)
4. Verify checksum of index on worker node

Advanced HTTP Tools: Parrot

- Can mount directories as http resources
- Using can even 'ls' (after custom indexing)
- Uses squid (with failover):

```
setenv HTTP_PROXY "http://proxy.nd.edu:8080;http://
proxy.wisc.edu:1000;DIRECT"
```

Questions?