# CentOS 7 Configuration for AGILE Pipelines

N. Parmiggiani[(1)], A. Bulgarelli[(1)]

[(1)]INAF/OAS – Bologna, Via P. Gobetti 101, 40129 Bologna

**Change history:**

| Version | Date | Notes |
|---|---|---|
| 1.0 | Feb 21th, 2018 | |
| 1.1 | Oct 9th, 2020 | Commands update and verification |

## Table of contents

# 1. Introduction

This document can be used to configure a new machine with CentOS 7 operating system. This is the base configuration for machines used for the AGILE software analysis system. To configure a Virtual Machine the user must start from Sect. 2, otherwise it is possible to start directly from the Sect. 3.

# 2. Install CentOS 7.x on Virtual Machine

You can install the VM using your local machine as host machine.

1. Install VirtualBox on your local machine.
    a. Install EPEL and virtualbox repositories:

```
root@host # yum install epel-release
root@host # wget http://download.virtualbox.org/virtualbox/rpm/rhel/virtualbox.repo
root@host # cp virtualbox.repo /etc/yum.repos.d
```

    b. Install VirtualBox:

```
root@host # yum groupinstall "Development Tools"
root@host # yum install dkms kernel-devel
root@host # yum install VirtualBox-5.1
```

2. Download CentOS-7.x minimal:

```
user@local $ wget
http://mi.mirror.garr.it/mirrors/CentOS/7/isos/x86_64/CentOS-7-x86_64-Minimal-1511.iso
```

3. Start VirtualBox.

4. Add a new virtual machine, choose 2 cores and 2GB of RAM for start, with a new VHD dynamic disk up to 2TB of space. I created a single root **/** partition with **no swap** to minimize hd usage.
5. Create /boot 10gb /swap 10gb / 280gb xfs

6. Mount the CentOS 7 minimal iso and install it.

Start the virtual machine and update the system packages:

```
root@vm # yum update
```

NOTE: This is now a generic CentOS7 minimal virtual machine, you can export if you like.

NOTE: When exporting, use the OVA format, and try to keep the VHD format that is compatible for cloud systems like Amazon AWS.

## 3. Configure the firewall (iptables)

We use the old good iptables, start and enable it on boot:

```
root@vm # yum install -y fail2ban
root@vm # systemctl enable fail2ban
root@vm # yum install -y iptables-services
root@vm # systemctl start iptables
root@vm # systemctl enable iptables
```

Fail2ban /etc/fail2ban/jail.local

Add  if needed

```
[mysqld-auth]
enabled = true
```

```
filter   = mysqld-auth
port     = [port]
logpath  = /var/log/mysqld.log
```

We can now stop firewalld service and disable it on boot:

```
root@vm # systemctl stop firewalld
root@vm # systemctl disable firewalld
```

Save the current iptables rules to open port 22 and 80:

```
/root/iptables.txt
        *filter
        :INPUT ACCEPT [0:0]
        :FORWARD ACCEPT [0:0]
        :OUTPUT ACCEPT []
        -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
        -A INPUT -p icmp -j ACCEPT
        -A INPUT -i lo -j ACCEPT
        -A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
        -A INPUT -p tcp -m state --state NEWS -m tcp --dport 80 -j ACCEPT
        -A INPUT -j REJECT --reject-with icmp-host-prohibited
        COMMIT
```

Load the above firewall configuration and save it:

```
root@vm # iptables-restore iptables.txt
root@vm # service iptables save
```

## 4. Install LAMP

Install EPEL repository and mysql: search here https://dev.mysql.com/downloads/repo/yum/
Check version you want

```
root@vm # yum localinstall https://dev.mysql.com/get/mysql57-community-release-el7-9.noarch.rpm
root@vm # yum -y install mysql-community-server
Check if mysql-community-devel is installed
OLD
root@vm # yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
root@vm # yum install https://dev.mysql.com/get/mysql57-community-release-el7-9.noarch.rpm
root@vm # yum install mysql mysql-devel mysql-server
```

Apache anti-DDoS configuration:

```
/etc/my.cnf
        #validate_password_policy=LOW
        explicit_defaults_for_timestamp=1
```

Start mysql server and setup mysql, say the default YES to all the questions and set the password:

```
root@vm # stemctl start mysqld
root@vm # systemctl enable mysqld
root@vm # grep "temporary password" /var/log/mysqld.log
root@vm # /usr/bin/mysql_secure_installation -u root -p
[password]
```

Install and start the webserver, configure it to use php and to start on boot:

```
root@vm # yum install -y httpd php php-pdo mod_evasive mod_security phpmyadmin php-mcrypt
root # chown -R apache:apache /var/www
root # usermod -G users apache
```

Apache anti-DDoS configuration:

```
/etc/httpd/conf.d/mod_evasive.conf
        DOSPaugieCount 10
        DOSEmailNotify [user email]
        DOSWhitelist 127.0.0.1
```

Php configuration:

```
/etc/php.ini
        upload_max_filesize = 20M
        allow_url_fopen = On
        allow_url_include = Off
        memory_limit = 256M
        expose_php = Off
```

Change the blowfish 32-character pass for cookie auth:

```
/usr/share/phpmyadmin/config.inc.php
        $cfg['blowfish_secret'] = '<blowfish_pass>';
```

To generate a a random 32-character password use:

```
root@vm # head /dev/urandom | tr -dc A-Za-z0-9 | head -c 32 ; echo
```

Set permissions for phpmyadmin:

```
/etc/httpd/conf.d/phpmyadmin.conf
        #
```

```
#  Web application to manage MySQL
#

Alias /phpmyadmin "/usr/share/phpMyAdmin"
<Directory "/usr/share/phpMyAdmin">
        DirectoryIndex index.php
        AllowOverride All
        Options FollowSymlinks
        Require all granted
</Directory>

# These directories do not require access over HTTP - taken from the original
# phpMyAdmin upstream tarball
#
<Directory /usr/share/phpMyAdmin/libraries/>
        Order Deny,Allow
        Deny from All
        Allow from None
</Directory>

<Directory /usr/share/phpMyAdmin/setup/lib/>
        Order Deny,Allow
        Deny from All
        Allow from None
</Directory>

<Directory /usr/share/phpMyAdmin/setup/frames/>
        Order Deny,Allow
        Deny from All
        Allow from None
</Directory>
```

Start the web server and enable on boot:

```
root@vm # systemctl start httpd
root@vm # systemctl enable httpd
```

NOTE: This is now a generic CentOS7 LAMP virtual machine, you can export it if you like.

## 5. Configure network

Now that we have iptables up and running we can give the virtual machine a public IP.

Stop and disable NetworkManager on boot (alternatively you can set the static interface with nmcli instead of the configuration below):

```
root@vm # systemctl stop NetworkManager
root@vm # systemctl disable NetworkManager
```

The ethernet configuration is:

```
/etc/sysconfig/network-scripts/ifcfg-enp0s3
        TYPE="Ethernet"
        BOOTPROTO=none
        DEFROUTE="yes"
        IPV4_FAILURE_FATAL="no"
        IPV6INIT="no"
        NAME="enp0s3"
        UUID=
        DEVICE=
        ONBOOT="yes"
        PREFIX=24
        PEERDNS=no
        PEERROUTES=yes
        IPADDR=[IP]
        NETMASK=[NETMASK]
        GATEWAY=[GATEWAY]
```

Set the DNS resolver configuration:

```
/etc/resolv.conf
nameserver [ip]
nameserver [ip]
search [domain]
```

Update hosts file with static naming for internal machines:

```
/etc/hosts
        127.0.0.1       localhost localhost.localdomain localhost6 localhost6.localdomain6
        ::1     localhost localhost.localdomain localhost6 localhost6.localdomain6
```

Set the hostname:

```
/etc/hostname
        [hostname]
```

Reboot the virtual machine or run:

```
root@vm # systemctl restart network
```

NOTE: From now on you can access the vm using ssh

## 6. Install system packages and users

Install common packages:

```
root@vm # yum install -y yum-utils arpwatch wget rsync unzip vim mlocate pciutils bind-utils mailx
ImageMagick ntp bzip2 emacs gedit
```

Install monitoring tools and profilers:

```
root@vm # yum install -y iotop lsof htop
```

Install monitoring tools and profilers:

```
root@vm # systemctl start ntpd
root@vm # systemctl enable ntpd
```

Install development tools:

```
root # yum install -y gcc gcc-c++ git gdb valgrind perf gitk
```

Install keychain script to avoid multiple ssh-agent spawning:

```
root@vm # wget http://www.funtoo.org/distfiles/keychain/keychain-2.8.2.tar.bz2
root@vm # tar xvjf keychain-2.8.2.tar.bz2
root@vm # cp keychain-2.8.2/keychain /usr/bin/
root@vm # rm -rf keychain-2.8.2*
```

Install the nfs client:

```
root@vm # yum install -y nfs-utils
root@vm # systemctl start rpcbind
root@vm # systemctl enable rpcbind
```

Install opencv

```
root@vm # yum install -y opencv-devel.x86_64
```

Install bc

```
root@vm # yum install -y bc
```

Install basic Xorg enviroment:

```
root # yum install -y xorg-x11-server-Xorg xorg-x11-server-utils xorg-x11-xinit xorg-x11-utils xclock
xorg-x11-server-Xvfb xorg-x11-drv* libXt-devel
```

```
Optional next 3 commands
root # Xorg -configure
root # Xorg -configure
root # mv /root/xorg.conf.new /etc/X11/xorg.conf
root # yum groupinstall -y Fonts
```

NOTE: If you have an ASPEED old card the Xorg-configure gives segfault. Ignore the segfault and copy the xorg.conf.new as normal. Then set "ast" video driver in the relative section instead of vesa. It works.

Use vim instead of vi:

```
/etc/profile
        alias vi=vim
```

Enable ssh authentication using public key:

```
/etc/ssh/sshd_config
        RSAAuthentication yes
        PubkeyAuthentication yes
        AuthorizedKeysFile .ssh/authorized_keys
```

Set selinux policy as permissive or disabled:

```
/etc/selinux/config
        SELINUX=permissive
```

Enable root password for single user mode:

```
/etc/sysconfig/init
        SINGLE=/sbin/sulogin
```

Change aggressive default swapiness (/proc/sys/vm/swappiness gives 60 by default):

```
/etc/sysctl.conf
        vm.swappiness = 10
```

Do not show version of postfix:

```
/etc/postfix/main.cf
        smtpd_banner = $myhostname ESMTP
        mydomain = [domain]
```

Remove the annoying terminal beep on tab completition:

```
/etc/inputrc
        set bell-style none
```

Create the **agile** group and change apache permissions (required for the agile archive):

```
root@vm # groupadd -g 500 agile
root@vm # usermod -g apache -G users,agile apache
```

Create the users **rt** and **agileobs**:

```
root@vm # groupadd -g 300 rt
root@vm # useradd -m -g 300 -u 300 -G video,audio,users,agile rt
root@vm # passwd rt
root@vm # groupadd -g 301 agileobs
root@vm # useradd -m -g 301 -u 301 -G video,audio,users,agile agileobs
root@vm # passwd agileobs
root@vm # useradd -m -g 301 -u 302 -G video,audio,users,agile agileusers
root@vm # useradd -m -g 301 -u 303 -G video,audio,users,agile asdcdaemon
root@vm # passwd agileusers
root@vm # passwd asdcdaemon
```

Setup the rt ssh private key :

```
root@vm # su - rt
rt@vm $ mkdir ~/.ssh
rt@vm $ chmod 700 .ssh
rt@vm $ scp rt@[machine name]:.ssh/rt* ~/.ssh
```

passphrase= []

To load rt identity to the ssh-agent identities on boot add the following to .bashrc, also enable core dumps:

```
/home/rt/.bashrc
        # Enable core dumps
        ulimit -c unlimited

        # Add rt identity to the ssh-agent
        /usr/bin/keychain --agents "ssh" $HOME/.ssh/rt &> /dev/null
        source $HOME/.keychain/$HOSTNAME-sh

        alias ll="ls -la"
```

NOTE: After each reboot (or ethvimrternet down/up) you need to connect through ssh to the machine and enter just once the rt private key adding it again to the ssh-agent.

Block ssh direct access to root

```
root@vm # vi /etc/ssh/sshd_config
Search PermitRootLogin and set to PermitRootLogin no
Save and restart service
```

```
root@vm # systemctl restart sshd
```

Time to reboot and check see if everything is ok:

```
root # reboot
```

# 7. Install task manager

## 7.1. Slurm

To use Slurm add this line in .bashrc

export EXECCOM="sbatch"

REQUIREMENTS:

Mysql mysql-devel, exit from anaconda environment

Tutorial to install slurm : https://www.slothparadise.com/how-to-install-slurm-on-centos-7-cluster/

You can get the new version to use with wget here:
https://www.schedmd.com/downloads.php

When an error occurs due to lack of perl dependency:

```
root@vm # yum install perl-ExtUtils-MakeMaker
root@vm #  rpmbuild -ta slurm-17.11.12.tar.bz2
root@vm # yum -y --nogpgcheck localinstall /root/rpmbuild/RPMS/x86_64/slurm-*rpm
```

For the following commands use * because file name are changed

```
root@vm #  mkdir /usr/sbin/slurm
root@vm #  chown  slurm:slurm /usr/sbin/slurm
root@vm #  mkdir /var/log/slurm
root@vm #  chown slurm:slurm /var/log/slurm
root@vm #  /var/spool/slurm/
root@vm # chown slurm:slurm /var/spool/slurm/
```

[redacted]

Create slurm.conf file inside /etc/slurm/

**SEE** chapter SLURM CONF for configuration file

# SLURM BASE CONFIGURATION

```
ControlMachine=[machine domain name]
ControlAddr=[control addr]
#BackupController=
#BackupAddr=
authtype=auth/munge
cachegroups=0
#CheckpointType=checkpoint/none
CryptoType=crypto/munge
#DisableRootJobs=NO
#EnforcePartLimits=NO
#Epilog=
#PrologSlurmctld=
#FirstJobId=1
JobCheckpointDir=/usr/sbin/checkpoint
#JobCredentialPrivateKey=
#JobCredentialPublicCertificate=
#JobFileAppend=0
#JobRequeue=1
#KillOnBadExit=0
#Licenses=foo*4,bar
MailProg=/usr/bin/mail
#MaxJobCount=5000
MpiDefault=none
#MpiParams=ports:#-#
#PluginDir=
#PlugStackConfig=
#PrivateData=jobs
ProctrackType=proctrack/pgid
#Prolog=
#PrologSlurmctld=
#PropagatePrioProcess=0
#PropagateResourceLimits=
#PropagateResourceLimitsExcept=
ReturnToService=1
#SallocDefaultCommand=
SlurmctldPidFile=/var/run/slurmctld.pid
```

```
SlurmctldPort=[port]
SlurmdPidFile=/var/run/slurmd.pid
SlurmdPort=[port]
SlurmdSpoolDir=/usr/sbin
SlurmUser=slurm
#SlurmdUser=root
#SrunEpilog=
#SrunProlog=
StateSaveLocation=/usr/sbin/slurm
SwitchType=switch/none
#TaskEpilog=
TaskPlugin=task/none
#TaskPluginParam=
#TaskProlog=
#TopologyPlugin=topology/tree
#TmpFs=/tmp
#TrackWCKey=no
#TreeWidth=
#UnkillableStepProgram=
#UnkillableStepTimeout=
#UsePAM=0



# TIMERS

#BatchStartTimeout=10
#CompleteWait=0
#EpilogMsgTime=2000
#GetEnvTimeout=2
#HealthCheckInterval=0
#HealthCheckProgram=
InactiveLimit=0
KillWait=30
#MessageTimeout=10
#ResvOverRun=0
MinJobAge=300
#OverTimeLimit=0
SlurmctldTimeout=300
SlurmdTimeout=300
#UnkillableStepProgram=
#UnkillableStepTimeout=60
Waittime=0
```

```
# SCHEDULING

#DefMemPerCPU=0
FastSchedule=1
#MaxMemPerCPU=0
#SchedulerRootFilter=1
#SchedulerTimeSlice=30
SchedulerType=sched/backfill
SchedulerPort=[]
SelectType=select/cons_res
SelectTypeParameters=CR_CPU


# JOB PRIORITY

#PriorityType=priority/basic
#PriorityDecayHalfLife=
#PriorityFavorSmall=
#PriorityMaxAge=
#PriorityUsageResetPeriod=
#PriorityWeightAge=
#PriorityWeightFairshare=
#PriorityWeightJobSize=
#PriorityWeightPartition=
#PriorityWeightQOS=



# LOGGING AND ACCOUNTING

#AccountingStorageEnforce=0
#AccountingStorageHost=
#AccountingStorageLoc=
#AccountingStoragePass=
#AccountingStoragePort=
AccountingStorageType=accounting_storage/none
#AccountingStorageUser=
ClusterName=cluster
#DebugFlags=
#JobCompHost=
#JobCompLoc=
#JobCompPass=
#JobCompPort=
#JobCompType=jobcomp/none
#JobCompUser=
#JobAcctGatherFrequency=30
```

JobAcctGatherType=jobacct_gather/none
#SlurmctldDebug=3
SlurmctldLogFile=/var/log/slurm/slurmctld.log
#SlurmdDebug=3
SlurmdLogFile=/var/log/slurm/slurmd.log

# POWER SAVE SUPPORT FOR IDLE NODES (optional)

#SuspendProgram=
#ResumeProgram=
#SuspendTimeout=
#ResumeTimeout=
#ResumeRate=
#SuspendExcNodes=
#SuspendExcParts=
#SuspendRate=
#SuspendTime=

#COMPUTE NODES

NodeName=[] NodeAddr=[] CPUs=[] RealMemory=[] Sockets=[]  CoresPerSocket=[] ThreadsPerCore=[] State=UNKNOWN
PartitionName=large Nodes=[] Default=YES MaxTime=28800 State=UP MaxCPUsPerNode=6
PartitionName=agilesor Nodes=[] Default=NO MaxTime=28800 State=UP MaxCPUsPerNode=1
PartitionName=fast Nodes=[] Default=NO MaxTime=28800 State=UP MaxCPUsPerNode=3

USEFULL COMMANDS

To change number of cpu per partition without reboot

```
root@vm # scontrol update Partition=[partition name] MaxCPUsPerNode=[number]
```

To restart node in dry state

```
root@vm # scontrol update nodename=[nodename] state=resume
```

## 8. Install environment modules

Install module:

```
root@vm # yum install environment-modules
root@vm # mkdir -p /opt/prod/modulefiles
```

Add our production path /opt/prod to the configuration:

```
/usr/share/Modules/init/.modulespath
        /opt/prod/modulefiles
```

## 9. Install AGILEPIPE (with spot6) and DeepVar

Here it is the list of the dependencies required for both AGILEPIPE and DeepVar:

- cfitsio-3.370
- root_v5.34.24
- ds9-7.3.2
- idl8.2sp2
- octave-devel
- heasoft
- libcurl-devel
- iniparser-devel
- libxml2-devel
- linphone rpms
- ruby
    - mysql
    - sqlite3
    - tslmail
- python2.7
    - numpy
    - matplotlib
    - scipy
    - astropy
    - pyfits
    - argparse
    - healpy
    - networkx
    - pyrr 0.6.2

### 9.1. Install ROOT, ds9, idl dependencies

Use this file to configure the license server

/opt/prod/idl8.6/idl8.6-install/license

```
root@vm # yum -y install libXext-devel libXpm-devel libXft-devel
root@vm # yum -y install libXScrnSaver
root@vm # yum -y install libXp
```

## 9.2. Install AGILEPIPE dependencies

Install gcndaemon dependencies:

```
root@vm # yum -y install libcurl-devel iniparser-devel libxml2-devel
```

The morfeoalarm uses the linphone library, install the rpms:

```
root@vm # scp -r rt@[machine name]:linphone-rpm .
root@vm # cd linphone-rpm
root@vm # yum install *.rpm
```

Install gcn2conf.rb/spot6 dependencies:

```
root@vm # yum install -y openssl-devel ruby ruby-devel rubygems sqlite-devel
root@vm # gem install sqlite3
```

AGILEPIPE uses python for Li&Ma analysis grb_gw.py and for ligo contour extraction. We installed python2.7 packages on system using pip.

You need the pypa repository:

```
/etc/yum.repos.d/pypa.repo
        [pypa-pypa]
        name=Copr repo for pypa owned by pypa
        baseurl=https://copr-be.cloud.fedoraproject.org/results/pypa/pypa/epel-7-$basearch/
        type=rpm-md
        skip_if_unavailable=True
        gpgcheck=1
        gpgkey=https://copr-be.cloud.fedoraproject.org/results/pypa/pypa/pubkey.gpg
        repo_gpgcheck=0
        enabled=1
        enabled_metadata=1
```

Install the python packages with pip:

```
root@vm # yum install lapack tk
root@vm # yum update [evalutate this command]
root@vm # yum install tkinter python python-devel
root@vm # yum install python2-pip
root@vm # pip install --upgrade pip
```

```
root@vm # yum install python-setuptools python-wheel
root@vm # pip install numpy matplotlib scipy astropy pyfits healpy networkx
root@vm # pip install
https://pypi.python.org/packages/5b/d9/d9c6fa4c425a9171aed645a077bd73743858ad14c0bfba7f5f2ad
06abc3f/pyrr-0.6.2.tar.gz#md5=4931f7568f19052bc200fb5caac9406a
```

## 9.3. Install AGILEPIPE

```
root@vm # mkdir -p /opt/prod/AGILEPIPE
root@vm # chown rt:rt /opt/prod/AGILEPIPE
root@vm # su - rt
rt@vm $ cd /opt/prod
rt@vm $ git clone https://github.com/ASTRO-EDU/AGILEPIPE.git
rt@vm $ cd AGILEPIPE
rt@vm $ git checkout centos7
rt@vm $ cd /opt/prod/AGILEPIPE/gcn
rt@vm $ make
rt@vm $ cd /opt/prod/AGILEPIPE/morfeoalarm
rt@vm $ make
rt@vm $ touch answers.log
rt@vm $ cd /opt/prod/AGILEPIPE/ligocontour
rt@vm $ make
```

## 9.4. Install AGILEPIPE-Web

```
root@vm # cd /var/www/html
root@vm # git clone https://github.com/ASTRO-EDU/AGILEPIPE-Web.git
```

Set the apache permissions (is important to deny access to analysis/commands!!):

```
/etc/httpd/conf.d/agileapp.conf
        <Directory  /var/www/html/analysis/ >
                Options FollowSymLinks
                Options +Indexes
                AllowOverride All
                allow from all
                ServerSignature Off
        </Directory>

        <Directory  /var/www/html/analysis/commands >
```

```
        allow from 127.0.0.1
        deny from all
    </Directory>

    <Directory  /var/www/html/analysis/log >
        deny from all
    </Directory>
```

We need to assign apache to the rt group:

```
root@vm # usermod -G apache,users,rt -g apache apache
root@vm # id apache
```

## 9.5. Install DeepVar dependencies

```
root@vm # yum install ruby ruby-devel rubygems
root@vm # gem install mysql tlsmail
root@vm # yum install octave octave-devel
```

## 9.6. Install DeepVar

```
root@vm # mkdir -p /opt/prod/DeepVar
root@vm # chown rt:rt /opt/prod/DeepVar
root@vm # ln -s /opt/prod/DeepVar $AGILE/DeepVar
root@vm # su - rt
rt@vm $ cd /opt/prod
rt@vm $ git clone https://github.com/ASTRO-EDU/DeepVar.git
```

8.6.1 Import spot6 detection to database

Setup database: copy an existing spot6 database

```
root@agilepipe # mysqldump -p -u root spot6 > spot6.sql
root@agilepipe # scp spot6.sql rt@[backup machine]:~
root@agilepipe # rm spot6.sql
root@vm # mysql -p -u root --execute="DROP DATABASE spot6; CREATE DATABASE spot6;"
root@vm # mysql -p -u root -D spot6 < /home/rt/spot6.sql
root@vm # rm /home/rt/spot6.sql
```

Setup configuration for database connection

```
rt@agilepipe $ cp DepeVar/import/conf.txt.default DeepVar/import/conf.txt
rt@agilepipe $ vim conf.txt
```

Start task from crontab -> remove comment to the following lines

```
### import spot6 alerts
*/5 * * * * pgrep import_last_spot6.rb > /dev/null || (. ~/.bashrc ; date >>
$LOG/import_detection_spot6.log ; cd $AGILE/DeepVar/import ; pwd >> $LOG/import_detection_spot6.log
; ruby ./import_last_spot6.rb >> $LOG/import_detection_spot6.log 2>&1)
```

Backup database

```
### backup spot6 database
0 0 * * * /usr/bin/sh /home/rt/spot6db_backup/backup.sh
0 0 * * * rsync -avz /home/rt/spot6db_backup/spot6.sql /data01/backup/mysql
```

## 9.7. Install AGILE-MCAL

Create the **module file:**

```
/opt/prod/modulefiles/agile-mcal
        #%Module

        module load root_v5.34.24
        module load cfitsio-3.370

        module-whatis "AGILE Mcal commit
        cf12ac6993d38d22262114df6baad3787b9e6a32"

        set path /opt/prod/mcalsw/
        setenv MCALSW           $path
        prepend-path LD_LIBRARY_PATH $path/lib
        prepend-path PATH               $path/bin:$path/scripts
```

Install the AGILE-MCAL in the /opt/prod/mcalsw:

```
rt $ cd
rt $ module load agile-mcal
rt $ git clone https://github.com/ASTRO-EDU/AGILE-MCAL
rt $ cd AGILE-MCAL
rt $ git checkout pipeline
```

```
rt $ make install
```

Install basemap for MCAL PIPE

### 9.8. https://github.com/matplotlib/basemap v1.0.7

### 9.9. INSTALL MCALPIPE

Log /ANALYSIS3/log/mcal_search_new_orbit.log
And slurm.out inside /ANALYSIS3/AGILE-MCAL orbit dir

### 9.10. Copy agile-mcal, agile-B23, agile-preB24, cfitsio, ds9, heasoft and idl

```
root@vm # cd /opt/prod
root@vm # rsync -av rt@[machine name]:/opt/prod/mcalsw .
root@vm # mkdir root
root@vm # cd root
root@vm # rsync -av rt@[machine name]:/opt/prod/root/root_v5.34.24 .
root@vm # cd /opt/prod
root@vm # rsync -av rt@[machine name]:/opt/prod/cfitsio .
root@vm # rsync -av rt@[machine name]:/opt/prod/ds9 .
root@vm # rsync -av rt@v:/opt/prod/agile-model .mm
root@vm # rsync -av rt@[machine name]:/opt/prod/idl8.2sp2 .
root@vm # rsync -av rt@[machine name]:/opt/prod/heasoft-6.17  .
root@vm # rsync -av
rt@[machinename]:/opt/prod/modulefiles/{agile*,cfitsio*,ds9*,heasoft*,idl*,root_v5.34.24}
/opt/prod/tmp/modulefiles
```

## 10. Install munin

Install munin:

```
root@vm # yum install -y munin munin-node
```

Set the munin password for [username]:

```
root@vm # htpasswd -c /etc/munin/munin-htpasswd [username]
```

Setup password for acces httpd

```
root@vm # vim /etc/httpd/conf.d/munin.conf
```

```
# This file can be used as a .htaccess file, or a part of your apache
# config file.
#
# For the .htaccess file option to work the munin www directory
# (/var/www/html/munin) must have "AllowOverride all" or something close
# to that set.
#
# As a config file enclose it in <directory> like so:
#
<directory /var/www/html/munin>

AuthUserFile /etc/munin/munin-htpasswd
AuthName "Munin"
AuthType Basic
require valid-user

# This next part requires mod_expires to be enabled.
#
# We could use <IfModule mod_expires> around here, but I want it to be
# as evident as possible that you either have to load mod_expires _or_
# you coment out/remove these lines.

# Set the default expiery time for files 5 minutes 10 seconds from
# their creation (modification) time.  There are probably new files by
# that time.

ExpiresActive On
ExpiresDefault M310

</directory>
ScriptAlias /munin-cgi/munin-cgi-graph /var/www/cgi-bin/munin-cgi-graph
<Location /munin-cgi/munin-cgi-graph>
 AuthUserFile /etc/munin/munin-htpasswd
 AuthName "Munin"
 AuthType Basic
 require valid-user
</Location>
```

Setup the paths:

```
/etc/munin/munin.conf
        dbdir   /var/lib/munin
        htmldir /var/www/html/munin
        logdir /var/log/munin
        rundir  /var/run/munin
```

Change permissions on www dir and start the service:

```
root@vm # chown munin:munin /var/www/html/munin
root@vm # systemctl start munin-node
root@vm # systemctl enable munin-node
```

## 10.1. Create the agile archive directory structure

The following paths are used by the rt pipeline and by agileobs:

```
root@vm # mkdir /data01
root@vm # cd /data01
root@vm # mkdir GTB_PROC3
root@vm # mkdir -p ASDC_PROC2/{DATA_2/{INDEX,LOG,AUX,COR},FM3.119_2/{INDEX,EVT}}
root@vm # mkdir -p
ASDC_PROC3/{DATA_ASDCSTDk/{INDEX,LOG},DATA_ASDCSTDf/{INDEX,LOG},DATA_ASDCe/{INDEX,LOG},
FM3.119_ASDC2,FM3.119_ASDCSTDf/{INDEX,EVT},FM3.119_ASDCSTDk/{INDEX,EVT},FM3.119_ASDCe/{I
NDEX,EVT,EVTROOT}}
root@vm # mkdir -p ANALYSIS3/{aitoff_rt,alerts,app,commands,log,spot6,monitoring,db}
root@vm # cd /
root@vm # ln -s /data01/ANALYSIS3 .
root@vm # ln -s /data01/ASDC_PROC2 .
root@vm # ln -s /data01/ASDC_PROC3 .
```

/data01/AGILE_PROC3 contains **symlinks** for the various scripts
/data01/GTB_PROC3 is the **rt/consolidated** archive from GTB
/data01/ASDC_PROC2 is the **rt** archive from ASDC
/data01/ASDC_PROC3 is the **consolidated** archive ASDC
/data01/ANALYSIS3 is the analysis **output** archive.

The rt-alert db is in

/var/rt

and you have to create the following link

```
root@vm # ln -s /var/rt/ /home/rt/db
```

## 10.2. Create the AGILE_PROC3 symlinks:

Create all the required symlinks within /AGILE_PROC3:

```
root@vm # mkdir /AGILE_PROC3
```

```
root@vm # cd /AGILE_PROC3
root@vm # ln -s /ASDC_PROC2/DATA_2 DATA_2
root@vm # ln -s /ASDC_PROC2/DATA_2 DATA_ASDC2
root@vm # ln -s /ASDC_PROC3/DATA_ASDCSTDf DATA_ASDCSTDf
root@vm # ln -s /ASDC_PROC3/DATA_ASDCSTDk DATA_ASDCSTDk
root@vm # ln -s /ASDC_PROC3/DATA_ASDCe DATA_ASDCe
root@vm # ln -s /ASDC_PROC2/FM3.119_2 FM3.119_2
root@vm # ln -s /ASDC_PROC2/FM3.119_2 FM3.119_ASDC2
root@vm # ln -s /ASDC_PROC3/FM3.119_ASDCSTDf FM3.119_ASDCSTDf
root@vm # ln -s /ASDC_PROC3/FM3.119_ASDCSTDk FM3.119_ASDCSTDk
root@vm # ln -s /ASDC_PROC3/FM3.119_ASDCe FM3.119_ASDCe
root@vm # ln -s /ASDC_PROC2/FT3ab_2 .
root@vm # ln -s /ASDC_PROC2/FT3ab_2 FT3ab_ASDC2
```

## 10.3. Copy the archive data

We can now copy the AGILE archives in the /data01 shared folder:

```
root@vm # rsync -av rt@[machine name]:/ASDC_PROC2/ /data01/ASDC_PROC2/
root@vm # rsync -av rt@[machine name]:/ASDC_PROC3/ /data01/ASDC_PROC3/
```

## 10.4. Set the archive permissions

Set the archive file and directory file permissions:

```
root@vm # chown -Rh rt:agile /data01
root@vm # find /data01 -type d -exec chmod 750 {} \;
root@vm # find /data01 -type f -exec chmod 640 {} \;
```

## 11. Disable root ssh login

vi /etc/ssh/sshd_config

## 12. Setup passwordless rsync

If passwordless connections are not yet established (only the first time) you need add the key to the required machines using ssh-copy-id:

```
rt $ ssh-keygen -R [machine name]
rt $ ssh-copy-id -i ~/.ssh/rt [user]@[machine name]
```

NOTE: Assuming you have the ssh-agent up and running and the key configured with keychain, if you still have connection issues is 99% of the time because of permissions on the /home /home/user or /home/user/.ssh folders and content. See /var/log/secure if gives errors on permissions.

## 13. Setup certificates

Using this guide https://coderwall.com/p/ez1x2w/send-mail-like-a-boss:

```
rt@vm $ mkdir ~/.certs
rt@vm $ certutil -N -d ~/.certs  ----  password
rt@vm $ echo -n | openssl s_client -connect smtp.gmail.com:465 | sed -ne '/-BEGIN
CERTIFICATE-/,/-END CERTIFICATE-/p' > ~/.certs/gmail.crt
rt@vm $ certutil -A -n "Google Internet Authority" -t "C,," -d ~/.certs -i ~/.certs/gmail.crt
```

To test use:

```
rt@vm $ mailx -v -s "Email subject" -S smtp-use-starttls -S ssl-verify=ignore -S smtp-auth=login -S
smtp=smtp://smtp.gmail.com:587 -S from="[user]@gmail.com" -S smtp-auth-user=[user]@gmail.com -S
smtp-auth-password=password -S nss-config-dir=~/.certs -S ssl-verify=ignore [target user]@gmail.com
<< EOF
hi,
this is the body
EOF
```

## 14. Setup .bashrc and crontab

Edit the bashrc to load the environment and env paths used by spot6:

```
/home/rt/.bashrc
        # .bashrc

        # Source global definitions
        if [ -f /etc/bashrc ]; then
                . /etc/bashrc
        fi
```

```
# Enable core dumps
ulimit -c unlimited

# Add rt identity to the ssh-agent
/usr/bin/keychain --agents "ssh" $HOME/.ssh/rt &> /dev/null
source $HOME/.keychain/$HOSTNAME-sh

# AGILEPIPE env
export MODULELOAD="agile-preB24_3-r5"
module load $MODULELOAD
module load agile-mcal


export EXECCOM="llsubmit"
export PATH_DATA="/AGILE_PROC3/"
export PATH_RES="/ANALYSIS3/"
export ARCHIVE="ASDC2"
#spot6
export SPOT6CARDDIR="spot6"
export SPOT6QUEUE="large"
export SPOT6GEN_HTMLANDPNG=1
```

Test each **pipeline command** uncommenting from the following crontab:

```
crontab -e
PIPE=/opt/prod/AGILEPIPE
LOG=/ANALYSIS3/log

##################################################################################
# AUTOMATIC PIPELINE
#
# Run spot6, create .conf from spot6 and rt-alert database, run the analysis,
# update monitoring charts, rotate the pipeline logs, and perform cleaning.

*/2 * * * * pgrep updatecommandtime.sh > /dev/null || (. ~/.bashrc ; date >>
$LOG/updatecommandtime.log ; $PIPE/updatecommandtime.sh >>
$LOG/updatecommandtime.log 2>&1)

### generate conf from SPOT6
*/2 * * * * pgrep realtime_run.rb > /dev/null || $(. ~/.bashrc ; date >> $LOG/realtime_run.log ;
$PIPE/spot6/realtime_run.rb >> $LOG/realtime_run.log 2>&1)
### import spot6 alerts
*/5 * * * * pgrep import_last_spot6.rb > /dev/null || (. ~/.bashrc ; date >>
$LOG/import_detection_spot6.log ; cd $AGILE/DeepVar/import ; pwd >>
```

```
$LOG/import_detection_spot6.log ; ./import_last_spot6.rb >> $LOG/import_detection_spot6.log
2>&1)

### generate conf from GCN
*/1 * * * * pgrep gcn2conf.rb > /dev/null || $(. ~/.bashrc ; date >> $LOG/gcn2conf.log ;
$PIPE/gcn/gcn2conf.rb >> $LOG/gcn2conf.log 2>&1)

### run analysis
*/1 * * * * pgrep submit_run.rb > /dev/null || $(. ~/.bashrc ; date >> $LOG/submit_run.log ;
$PIPE/submit_run.rb >> $LOG/submit_run.log 2>&1)

### where is my data?
*/5 * * * * pgrep whereismydata.sh > /dev/null || $(. ~/.bashrc ; date >>
$LOG/whereismydata.log ; $PIPE/whereismydata.sh >> $LOG/whereismydata.log 2>&1 ;
$PIPE/whereismydata.sh > /ANALYSIS3/monitoring/whereismydata 2>&1)

### update graphs
*/1 * * * * pgrep delaychart.sh > /dev/null || $(. ~/.bashrc ; $PIPE/delaychart.sh > /dev/null
2>&1)
* */1 * * * pgrep orbitchart.sh > /dev/null || $(. ~/.bashrc ; $PIPE/orbitchart.sh > /dev/null
2>&1)

### rotate logs every 2 days
0 0 */2 * * pgrep logrotate || $(. ~/.bashrc ; /usr/sbin/logrotate -s $LOG/logrotate.status -f
$PIPE/logrotate.conf > /dev/null 2>&1)

### clear temporary fits files older than 5 days at 01:00
0 1 * * * find /tmp -iname "file??????" -atime +5 -exec rm {} + 2> /dev/null


##################################################################################
# ONLY AGILEPIPE
#
# Start gcndaemon (agilepipe.conf), morfeoalarm, sync AGILE data,
# copy maps for mobile apps, and
# import a selection of spot6 alerts to the deep variability database.

#*/1 * * * * pgrep gcndaemon > /dev/null || (. ~/.bashrc ; date >> $LOG/gcndaemon.log ; nohup
$PIPE/gcn/gcndaemon $PIPE/agilepipe.conf >> $LOG/gcndaemon.log 2>&1 &)
#*/1 * * * * pgrep -fx "python -u morfeoalarm.py" > /dev/null || (. ~/.bashrc ; date >>
$LOG/morfeoalarm.log ; cd $PIPE/morfeoalarm ; nohup python -u morfeoalarm.py >>
$LOG/morfeoalarm.log 2>&1 &)
#*/1 * * * * pgrep syncdata.sh > /dev/null || (. ~/.bashrc ; date >> $LOG/syncdata.log ;
$PIPE/syncdata.sh >> $LOG/syncdata.log 2>&1)
```

```
### import spot6 alerts
*/5 * * * * pgrep import_last_spot6.rb > /dev/null || (. ~/.bashrc ; date >>
$LOG/import_detection_spot6.log ; cd $AGILE/DeepVar/import ; pwd >>
$LOG/import_detection_spot6.log ; ./import_last_spot6.rb >> $LOG/import_detection_spot6.log
2>&1)

##################################################################################
# ONLY AGILEPIPEBKP
#
# Start the gcndaemon (agilepipebkp.conf)

*/1 * * * * pgrep gcndaemon > /dev/null || (. ~/.bashrc ; date >> $LOG/gcndaemon.log ; nohup
$PIPE/gcn/gcndaemon $PIPE/agilepipebkp.conf >> $LOG/gcndaemon.log 2>&1 &)
```

## 15. Setup diskusage script

To monitor the disk usage and send an email on high disk usage install the following script under the root home directory (you can change the ADMIN and THRESHOLD fields:

```
/root/scripts/diskusagealert.sh
        #!/bin/bash

        ADMIN="[email]"
        THRESHOLD=90
        TMP_THRESHOLD=10000

        df -PkH | grep -vE '^Filesystem|tmpfs|cdrom|media' | awk '{ print $5 " " $6 }' | while read
output;
        do
                usep=$(echo $output | awk '{ print $1}' | cut -d'%' -f1 )
                partition=$(echo $output | awk '{print $2}' )
                if [ $usep -ge $THRESHOLD ] ; then
                echo "$(df -h)" |
                mail -s "Alert from $(hostname): disk usage $usep% on partition $partition" $ADMIN
                fi
        done

        tmp_mb=$(du -sm /tmp/ | cut -f1)
        if [ $tmp_mb -ge $TMP_THRESHOLD ] ; then
                echo "$(du -sh /tmp)" |
                mail -s "Alert from $(hostname): /tmp usage above $TMP_THRESHOLD MB" $ADMIN
        fi
```

NOTE: postfix is not configured to send mails outside the domain

```
chmod 770 diskusagealert.sh
```

```
crontab -e
        0 */2 * * * /root/scripts/diskusagealert.sh > /dev/null 2>&1
```

## 16. System packages security updates

To view the informations of the security updates:

```
root@vm # yum info-sec
```

To perform the update:

```
root@vm # yum update --security
```

## 17. Install Hermes

This tool is used to call, send sms, send email when a new alert from LIGO is received on the email account

Go to hermes directory

```
rt@vm # cd /opt/prod/AGILEPIPE/hermesalarm
```

Run for the first time the command to check mail

```
rt@vm # php -f hermes.php debug 0 0 0 0
```

The prompt will write a html link, copy it and paste it on the browser -> login with gmail user and allow the read permission. After that the browser will show you a code, copy it and paste it in the terminal prompt and then press enter.

The script write the triggerList.txt file whit all past GW alert, in the next run the script will compare each new alert with them to check if there is a new allert -> not delete the file

Put the following command in the crontab

### hermes check email GW circular and notice
*/5 * * * * pgrep hermes.php > /dev/null || (. ~/.bashrc ; date >> $LOG/hermes.log ; cd $AGILE/AGILEPIPE/hermesalarm/ ; pwd >> $LOG/hermes.log ; /usr/bin/php ./hermes.php prod 1 1 1 0 >> $LOG/hermes.log 2>&1)

## 18. password http page reserved

Login as root and vim /etc/httpd/conf/httpd.conf

```
<Directory "/var/www/html">
    #
    # Possible values for the Options directive are "None", "All",
    # or any combination of:
    #   Indexes Includes FollowSymLinks SymLinksifOwnerMatch ExecCGI MultiViews
    #
    # Note that "MultiViews" must be named *explicitly* --- "Options All"
    # doesn't give it to you.
    #
    # The Options directive is both complicated and important.  Please see
    # http://httpd.apache.org/docs/2.4/mod/core.html#options
    # for more information.
    #
    Options Indexes FollowSymLinks

    #
    # AllowOverride controls what directives may be placed in .htaccess files.
    # It can be "All", "None", or any combination of the keywords:
    #   Options FileInfo AuthConfig Limit
    #
    AllowOverride None

    #
    # Controls who can get stuff from this server.
    #
    #Require all granted
    AuthType Basic
    AuthName "Restricted Content"
    AuthUserFile /etc/httpd/.htpasswd
    Require valid-user
</Directory>
<Directory "/var/www/html/agilemcal">

    Options -Indexes
    #Satisfy Any
    Require all granted
</Directory>
<Directory "/var/www/html/mcaldata">

    Options -Indexes
    #Satisfy Any
    Require all granted
</Directory>

<Directory "/var/www/html/spot6">

    Options -Indexes
    #Satisfy Any
```

```
  Require all granted
</Directory>
<Directory "/var/www/html/AGILEApp">

  #Options -Indexes
  #Satisfy Any
  Require all granted
</Directory>
<Directory "/var/www/html/analysis">

  #Options -Indexes
  #Satisfy Any
  #AllowOverride None
  #Options None
  Require all granted

</Directory>
<Directory "/var/www/html/analysis/LIGO-VIRGO">

  AuthType Basic
  AuthName "Restricted Content"
  AuthUserFile /etc/httpd/.htpasswd
  Require valid-user
</Directory>
<Directory "/var/www/html/analysis/LIGO-VIRGO_RUN">

  AuthType Basic
  AuthName "Restricted Content"
  AuthUserFile /etc/httpd/.htpasswd
  Require valid-user
</Directory>
<Files "loadconf.php">
  Require all granted
  #Satisfy Any
</Files>
```

Set the password for user

```
root@vm # htpasswd -c /etc/httpd/.htpasswd [user name]
Prompt password
```

## 19. INSTALL AGILE-MCAL-PIPE

```
root@vm # cd /opt/prod/
root@vm # git clone https://github.com/ASTRO-EDU/AGILE-MCAL-PIPE.git
root@vm # chown -R rt:rt AGILE-MCAL-PIPE
root@vm # mkdir  /ANALYSIS3/AGILE-MCAL
rt@vm $ cd AGILE-MCAL-WEB/database/
rt@vm $ mysql -uroot -p --execute="DROP DATABASE mysql; CREATE DATABASE mysql;"
rt@vm $ mysql -uroot -p mcal < mcal_schema.sql
rt@vm $ cp import/config.rb.default import/config.rb
rt@vm $ vim import/config.rb
```

Update crontab

```
### MCAL PIPE
* * * * * pgrep -xf "ruby ./mcal_search_new_orbit.rb" > /dev/null || (. ~/.bashrc ; date >>
$LOG/mcal_search_new_orbit.log ; cd /opt/prod/AGILE-MCAL-PIPE/pipe ; pwd >> $LOG/mcal_search_new_orbit.log ;
ruby ./mcal_search_new_orbit.rb >> $LOG/mcal_search_new_orbit.log 2>&1)

##import mcal on database
* * * * * pgrep -xf "ruby ./import_new_mcal_orbit.rb" > /dev/null || (. ~/.bashrc ; date >>
$LOG/import_new_mcal_orbit.log ; cd /opt/prod/AGILE-MCAL-PIPE/database/import ; pwd >>
$LOG/import_new_mcal_orbit.log ; ruby ./import_new_mcal_orbit.rb >> $LOG/import_new_mcal_orbit.log 2>&1)
```

Configure website to view result

```
root@vm # ln -s /opt/prod/AGILE-MCAL-PIPE/website/  /var/www/html/agilemcal/
rt@vm $ cd AGILE-MCAL-WEB/website/
rt@vm $ cp config.php.default config.php
rt@vm $ vim config.php
```

The error log are in slurm.out in orbit directory in  /ANALYSIS3/AGILE-MCAL
Or in /ANALYSIS3/log

## 20. INSTALL AGILE-GRID-SHORT

```
root@vm # cd /opt/prod/
root@vm # git clone https://github.com/ASTRO-EDU/AGILE-GRID-PIPE-SHORT.git
root@vm # chown -R rt:rt AGILE-GRID-PIPE-SHORT
root@vm # mkidr /ANALYSIS3/AGILE-GRID-SHORT
rt@vm $ mysql -uroot -p --execute="DROP DATABASE grid_short; CREATE DATABASE grid_short;"
rt@vm $ cd AGILE-GRID-PIPE-SHORT/database/
rt@vm $ mysql -uroot -p grid_short < grid_short.sql
```

```
rt@vm $ cp import/config.rb.default import/config.rb
rt@vm $ vim import/config.rb
rt@vm $ cd database
rt@vm $ ruby import_ring.rb healpix_2.txt
```

Configure website to view result

```
root@vm # ln -s /opt/prod/AGILE-GRID-PIPE-SHORT/website/  /var/www/html/agilegridshort/
rt@vm $ cd AGILE-GRID-PIPE-SHORT/website/
rt@vm $ cp config.php.default config.php
rt@vm $ vim config.php
```

Activate crontab

```
### GRID SHORT PIPE
#* * * * * pgrep -xf "ruby ./run_grid_short_pipe.rb" > /dev/null || (. ~/.bashrc ; date >> $LOG/grid_short_analysis.log ;
cd /opt/prod/AGILE-GRID-PIPE-SHORT/pipe ; pwd >> $LOG/grid_short_analysis.log ; ruby ./run_grid_short_pipe.rb >>
$LOG/grid_short_analysis.log 2>&1)
```

The error log are in slurm.out in orbit directory in  /ANALYSIS3/AGILE-GRID-SHORT
Or in /ANALYSIS3/log

## 21. Install  OpenCV

http://docs.opencv.org/2.4/doc/tutorials/introduction/linux_install/linux_install.html#linux-installation

Centos http://techieroop.com/install-opencv-in-centos/

Add to the installation command  -D WITH_CUDA=OFF

## 22. INSTALL SINGULARITY

Easy install following docs:
https://www.sylabs.io/guides/2.6/user-guide/installation.html#installation
Take care to have squashfs-tools in addition to those indicated.

In short (on centos):
yum update  (DANGER, be carefull) non lanciarlo
yum groupinstall 'Development Tools' --setopt=group_package_types=mandatory,default,optional
yum install libarchive-devel squashfs-tools

```
git clone https://github.com/sylabs/singularity.git
cd singularity
git fetch --all
git checkout 2.6.0
./autogen.sh
./configure --prefix=/usr/local
make install
```

To redirect PORT for apache inside container use this command

```
root # iptables -tnat -A PREROUTING -p tcp -m tcp --dport 80 -j REDIRECT --to-ports 28080
```

## 23. Setup NFS

### 23.1. Client and server configuration step 1

On both the **client** and **server** edit the following:

```
/etc/sysconfig/nfs
        RQUOTAD_PORT=[]
        LOCKD_TCPPORT=[]
        LOCKD_UDPPORT=[]
        MOUNTD_PORT=[]
        STATD_PORT=[]
        STATD_OUTGOING_PORT=[]
```

On both the **client** and **server** edit the following:

```
/etc/idmapd.conf
        [General]
        Domain = local
        [Translation]
        Method = nsswitch
```

To be able to setup a **NFSv4** export you need to setup a virtual root directory on the **server**. You need to specify **fsid=0** for the virtual root directory then the subdirectories:

```
/etc/exports
        /[directory base]/        [IP]/24(insecure,rw,sync,no_root_squash,fsid=0)
        /[directory base]/[directory]/  [IP]/24(insecure,rw,sync,no_root_squash)
```

Reload the exported files with:

```
root # exportfs -ax
```

NOTE: You cannot export both a directory and one of its subdirectory. It works only for the virtual root directory.

On **server** you need to bind the directories you prefer to the export mountpoints:

```
/etc/fstab
        [directory base]
        /[dircetory]            /[directory base]/[directory]       none    bind     0 0
```

After that, do the following command

```
root # mount /[directory base]/[directory]/
```

## 23.2. Client mount

Create the entry point directories:

```
root # mkdir -p /disks/[machine name]/[directory]
```

Two options:

- Automount
- Manual mount

### 23.2.1. Automount

On **client** you need to mount the exported directories **without the virtual root directory**:

```
/etc/fstab
        [machine]:/[bas directory]/[directory] /disks/[machine]/[directory] nfs
        rw,hard,intr,_netdev 0 0
```

### 23.2.2. Manual mount

Use the following command line, e.g.

```
root # mount [machine]:/[directory base]/[directory] /disks/[machine ]/[directory]
```

## 30.1 Client and server configuration step 2

On **server** and **client** restart the services the CentOS 6 way:

```
root # /etc/init.d/rpcbind restart
root # /etc/init.d/nfs restart
root # chkconfig rpcbind on
root # chkconfig nfs on
```

Or on **server** and **client** restart the services the CentOS 7 way:

```
root # systemctl restart rpcbind
root # systemctl restart nfs-server
root # systemctl restart rpcidmapd
root # systemctl enable rpcbind
root # systemctl enable nfs-server
root # systemctl enable rpcidmapd
```

We need to update the iptables on **server**, starting from the current configuration (port 22 opened presumably):

```
root # iptables-save > /root/iptables.txt
```

Change the rules to open the right ports for nfs only on the given subnet,
Update the current iptables:

```
root # iptables-restore /root/iptables.txt
root # service iptables save
root # iptables -L
```

From the **client** you can see a list of mounts available from a given **server** with:

```
root # showmount -e [remote machine]
```

**If automount execute the following commands**

From the client you can mount the mount points you have previously set on /etc/fstab:

```
root # mount /[directory]
```