System Administrators (Debian)

Select System Administration topic from the side menu for the following subjects:

Installation

- How to install HUBzero via Debian packages

Upgrading

- How to upgrade from older versions of HUBzero to this release
- How to upgrade to newer versions of Debian/Linux with this HUBzero release

Installation

What is HUBzero?

HUBzero is a platform used to create dynamic web sites for scientific research and educational activities. With HUBzero, you can easily publish your research software and related educational materials on the web. Powerful middleware serves up interactive simulation and modeling tools via your web browser. These tools can connect you with rendering farms and powerful Grid computing resources.

Minimum System Requirements

HUBzero installations require one or more dedicated hosts running Debian GNU/Linux version 7 (wheezy) or version 8 (jessie).

A typical starter HUBzero installation might consist of a single physical server with dual 64-bit quad-core CPUs, 24 Gigabytes of RAM and a terabyte of disk.

Production systems should try to not limit hardware resources, HUBzero is designed to run on systems with many CPU cores and lots of RAM. If you are looking for a system to run a small site with limited physical or virtual resources this is probably not the system for you. However, for demonstration or development purposes we often create VM images with less than a gigabyte of RAM and 5 gigabytes of disk. While fully functional, these virtual machines would only be suitable for a single user doing development or testing.

Target Audience

This document and the installation and maintenance of a HUBzero system has a target audience of **experienced** Linux administrators (preferably experienced with Debian GNU/Linux).

Linux

Install Basic Operating System

The latest version of Debian GNU/Linux 7 (7.11 as of this writing) or Debian GNU/Linux 8 (8.10 as of this writing) on a 64-bit Intel compatible system (amd64 packaging) should be installed on each host used by a HUBzero installation.

To install Debian GNU/Linux, you can easily obtain a copy, and then follow the installation instructions for your release and architecture. Installing Debian GNU/Linux using a small bootable CD (see iso-cd subdirectory) is the simplest method. When the installation is complete your system will reboot into a Debian GNU/Linux system. Don't forget to remove your installation media and/or change your server's boot media order if you changed them prior to installation.

The precise configuration (such as disk configuration, networking, etc) is dependent on how the hub is to be used and what hardware is being used. These instructions outline the simplest "hub in a box" configuration but may not be suitable for larger sites. It is expected that the hub will be managed by an experienced Linux administrator who can help scale your site to the capacity required.

Set hostname

Throughout this documentation you will see specific instructions for running commands, with part of the text highlighted. The highlighted text should be modified to your local configuration choices. (e.g. replace "example.com" with the fully qualified hostname of your machine).

Optional. If you didn't specify the fully qualified domain name when running setup you will need to set it here.

HUBzero expects the `hostname` command to return the fully qualified hostname for the system.

```
# hostname example.com
```

To make the change permanent you must also edit the file /etc/hostname, this be done simply with:

```
# echo "example.com" > /etc/hostname
```

Fix hosts

Now edit /etc/hosts by making sure that a line exists that looks like

```
127.0.1.1    example.com     example
```

Any other lines with "127.0.1.1" should be removed.

Delete local users

HUBzero reserves all user ids from 1000 up for hub accounts. As part of the HUBzero middleware every account must map to a corresponding system account. Therefore when starting up a hub it is required to remove all accounts that have user ids 1000 or greater. On a new installation there is typically one such account that is created when you set up the hub, and this account can be removed as follows:

```
# rm -fr /home/username
# deluser username
```

If you require additional system accounts, they can be numbered between 500-999 without interfering with hub operations.

Configure Networking

Optional. If you didn't configure networking during installation you will need to do so now.

For help with networking setup try this link.

Setting up your IP address.

The IP addresses associated with any network cards you might have are read from the file **/etc/network/interfaces**. This file has documentation you can read with:

```
# man interfaces
```

A sample entry for a machine with a static address would look something like this:

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
 address 192.168.1.90
 gateway 192.168.1.1
 netmask 255.255.255.0
 network 192.168.1.0
 broadcast 192.168.1.255
```

Here we've setup the IP addresss, the default gateway, and the netmask.

For a machine running DHCP the setup would look much simpler:

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface - use DHCP to find our address
auto eth0
iface eth0 inet dhcp
```

(If you're using a DHCP based setup you must have a DHCP client package installed - usually one of pump or dhcp-client.)

If you make changes to this file you can cause them to take effect by running:

```
# /etc/init.d/networking restart
```

Setting up DNS

Use whatever nameserver and other options as recommended by your ISP. If you used DHCP to set up networking it is likely this has already been set.

When it comes to DNS setup Debian doesn't differ from other distributions. To cause your machine to consult with a particular server for name lookups you simply add their addresses to /etc/resolv.conf.

For example a machine which should perform lookups from the DNS server at IP address 192.168.1.10 would have a resolv.conf file looking like this:

```
nameserver 192.168.1.10
```

Configure Advanced Package Tool

You will need to get and install the hubzero archive key to be able to verify packages from the hubzero archive:

```
apt-key adv --keyserver pgp.mit.edu --recv-keys 143C99EF
```

Now configure the location of the HUBzero package repository

If you are using Debian 7 "wheezy" add the following line to /etc/apt/sources.list

```
deb http://packages.hubzero.org/deb ellie-deb7 main
```

If you are using Debian 8 "jessie" add the following line to /etc/apt/sources.list

```
deb http://packages.hubzero.org/deb ellie-deb8 main
```

With the above configured, update the local package database with information about the packages now available through these new repositories:

```
apt-get update
```

MySQL

Install

```
# DEBIAN_FRONTEND=noninteractive apt-get install -y hubzero-mysql
```

If you leave off setting DEBIAN-FRONTED environment variable you will be prompted to enter a MySQL administrative password. This password will get reset at a later step.

If you already have mysql-server installed, be aware that the root password for mysql will get reset at a later step unless you take preventative action outlined here <link to be added later>.

Configure

Default configuration works well for starters. But for optimal performance you will need a database administrator capable of tuning your database to your hardware configuration and site usage.

Mail

Install

We need to install exim4 to enable outgoing email

```
# apt-get install -y exim4
```

Configure

```
# dpkg-reconfigure exim4-config
```

Configure mail as appropriate for your site and IT infrastructure. We outline a sample standalone configuration below. The requirement is for php to be able to send mail (registration confirmation and other notices need to go out) and for exim4 to receive mail (for support ticket and forum email gateway functions to work).

This is just an example of a standalone mail configuration.

General type of mail configuration

internet site; mail is sent and received directly using SMTP

Mail name

enter the fully qualified domain name (FQDN) of the host (example.com)

IP-addresses to listen on for incoming SMTP connections

leave blank (listen for connections on all available network interfaces)

Other destinations for which mail is accepted

leave blank or (equivalently) with local hostname (all local domains will be treated identically)

Domains to relay mail for

leave blank

Machines to relay mail for

leave blank

Keep number of DNS-queries minimal (Dial-on-Demand)

No

Delivery method for local mail

mbox format in /var/mail/

Split configuration into small files?

Yes

Test

Use a real email address below so you can see if you get the email

```
# Mail -v someone@gmail.com
```

CMS

Install

```
# apt-get install -y hubzero-cms-2.1.0
```

Configure

```
# hzcms install example
```

It is necessary to immediately run the updater to apply fixes that have not been incorporated into the initial installation.

```
# hzcms update
```

Disable the "default" site provided by Debian and then enable the http and https hub sites

```
# a2dissite 000-default
# a2ensite example example-ssl
# /etc/init.d/apache2 restart
```

SSL Configuration

The default SSL certificate is meant for evaluation purposes only. For a production Hub, you will need to obtain a certificate from a provider. A certificate may contain two or three pieces: a public certificate, a private key, and sometimes an intermediate certificate.

Once you obtain the certificate, install their respective sub-directories in the **/etc/ssl** directory.

After the certificates are installed, you will need to modify the Apache configuration template located in /etc/apache/sites-m4/{hubname}-ssl.m4.

Replace SSLCERTFILE on line 184 with the path to your SSL certificate.

SSLCertificateFile /etc/ssl/path/to/cert

Replace SSLCERTKEYFILE on line 185 with the path to your SSL key.

SSLCertificateKeyFile /etc/ssl/path/to/key

If you have an intermediate certificate:
remove line 186: ifdef(`SSLCERTCHAINFILE',`
remove line 188: ')dnl
Replace SSLCERTCHAINFILE with the path to your SSL key.

Once the paths have been updated, reconfigure the CMS.

```
# hzcms reconfigure example
# /etc/init.d/apache2 restart
```

If you are using the HTML5 VNC Proxy Server, you must update your certificate settings as well.

Test

The default installation of the CMS uses a self signed SSL certificate. Some browers will not accept this certificate and not allow access to the site.

https://support.mozilla.org/en-US/questions/1012036

You will need to install a proper SSL certificate.

OpenLDAP

Install HUBzero LDAP support

```
# apt-get install -y hubzero-openldap
```

You will be prompted to enter a LDAP administrative password.

Some packages will ask you to configure them when you run this step

Configuring nslcd: LDAP server URI:

Enter "ldap://localhost/"

Configuring nslcd: LDAP server search base:

keep the default

Configuring libnss-ldapd

Select only "group", "passwd", "shadow"

Configure OpenLDAP Database

```
# hzldap init
# hzcms configure ldap --enable
# hzldap syncusers
```

Test

```
# getent passwd
```

You should see an entry for user 'admin' toward the end of the list if everything is working correctly.

WebDAV

Install WebDAV

```
# apt-get install -y hubzero-webdav
```

Configure WebDAV

```
# hzcms configure webdav --enable
```

Test

```
# ls -l /webdav/home/admin
total 0
```

Browse to your site's https /webdav address (e.g. https://myhub/webdav). You should get prompted for a username and password. Use the admin account. You should see an empty directory listing and no error messages.

Now test using a WebDAV client.

```
# apt-get install cadaver
# cadaver https://localhost/webdav
```

You will be prompted to accept self signed certificate (if it is still installed) and then to enter your username and password. Use the 'admin' account again to test. When you get the "dav:/webdav/>" prompt just enter "ls" and it should show the test file.

Finally clean up test case

```
# apt-get purge cadaver
```

Troubleshooting

If the test doesn't work, check if the fuse kernel module is loaded

```
#  lsmod | grep fuse
fuse                   54176  0
```

If there is no output then try starting the kernel module manually

```
# modprobe fuse
```

Then try the test again

Subversion

Install

```
# apt-get install -y hubzero-subversion
```

Configure

```
# hzcms configure subversion --enable
```

Trac

Install

```
# apt-get install -y hubzero-trac
```

Configure

```
# hzcms configure  trac --enable
```

Forge

Install

```
# apt-get install -y hubzero-forge
```

Configure

```
# hzcms configure forge --enable
```

Firewall

Install

```
# apt-get install -y hubzero-firewall
```

HUBzero requires the use of iptables to route network connections between application sessions and the external network. The scripts controlling this can also be used to manage basic firewall operations for the site. If you use manage iptables with other tools you will have to make sure the rules in these scripts are maintained. /etc/firewall_on and /etc/firewall_off turn the HUBzero firewall on and off respectively. Scripts in /etc/rc.X/ to /etc/mw/firewall_on causes the script to run at startup (these links were created for you). The firewall is enabled in all boot modes 0-6. The basic scripts installed here block all access to the host except for those ports required by HUBzero (http,https,http-alt,ldap,ssh.smtp,mysql,submit,etc).

OpenVZ

Install

```
# apt-get install hubzero-openvz-repo
# apt-get update
```

Then install

```
# apt-get install hubzero-openvz
```

Configure

```
# hzcms configure openvz --enable
```

If configuration is successful it should prompt you to reboot the server to activate the new kernel.

```
# reboot
```

Test

```
# vzlist
Container(s) not found
```

Or it will list the containers currently running if you check this on a running hub. The salient point being that the command doesn't issue any kind of error message.

Maxwell Service

Install

```
# apt-get install -y hubzero-mw-service
```

Configure

```
# mkvztemplate amd64 wheezy ellie
```

Then:

```
# hzcms configure mw-service --enable
```

Test

```
# maxwell_service startvnc 1 800x600 24
```

Enter an 8 character password when prompted (e.g., "testtest")

This should result in a newly create OpenVZ session with an instance of a VNC server running inside of it. The output of the above command should look something like:

```
Reading passphrase:
testtest
==================== begin /etc/vz/conf/hub-
session-5.0-amd64.umount ========================

Removing /var/lib/vz/root/1 :root etc var tmp dev/shm dev
==================== end /etc/vz/conf/hub-
session-5.0-amd64.umount ========================
stunnel already running
Starting VE ...
```

```
==================== begin /etc/vz/conf/1.mount ====================
=====
Removing and repopulating: root etc var tmp dev
Mounting: /var/lib/vz/template/debian-5.0-amd64-maxwell home apps
==================== end /etc/vz/conf/1.mount =======================
=====
VE is mounted
Setting CPU units: 1000
Configure meminfo: 2000000
VE start in progress...
TIME: 0 seconds.
Waiting for container to finish booting.
/usr/lib/mw/startxvnc: Becoming nobody.
/usr/lib/mw/startxvnc: Waiting for 8-byte vncpasswd and EOF.
1+0 records in
1+0 records out
8 bytes (8 B) copied, 3.5333e-05 s, 226 kB/s
Got the vncpasswd
Adding auth for 10.51.0.1:0 and 10.51.0.1/unix:0
xauth:  creating new authority file Xauthority-10.51.0.1:0
Adding IP address(es): 10.51.0.1
if-up.d/mountnfs[venet0]: waiting for interface venet0:0 before doing
NFS mounts (warning).
WARNING: Settings were not saved and will be resetted to original valu
es on next start (use --save flag)
```

```
# vzlist
      VEID      NPROC STATUS  IP_ADDR         HOSTNAME

         1          6 running 10.51.0.1       -
```

```
# openssl s_client -connect localhost:4001
```

This should report an SSL connection with a self signed certificate and output text should end with:

```
---
```

```
RFB 003.008
```

If you see this then you successfully connected to the VNC server running inside the newly created OpenVZ session.

Clean up

```
# maxwell_service stopvnc 1
```

Which should give output similar to:

```
Killing 6 processes in veid 1 with signal 1
Killing 7 processes in veid 1 with signal 2
Killing 5 processes in veid 1 with signal 15
Got signal 9
Stopping VE ...
VE was stopped
====================== begin /etc/vz/conf/1.umount ====================
=====
Unmounting /var/lib/vz/root/1/usr
Unmounting /var/lib/vz/root/1/home
Unmounting /var/lib/vz/root/1/apps
Unmounting /var/lib/vz/root/1/.root

Removing /var/lib/vz/root/1 :root etc var tmp dev/shm dev
Removing /var/lib/vz/private/1: apps bin emul home lib lib32 lib64 mnt
 opt proc sbin sys usr .root
====================== end /etc/vz/conf/1.umount ======================
====
VE is unmounted
```

Maxwell Client

Install

```
# apt-get install -y hubzero-mw-client
```

Configure

```
# hzcms configure mw-client --enable
```

Test

```
# su www-data
$ ssh -i /etc/mw-client/maxwell.key root@localhost ls
The authenticity of host 'localhost (127.0.0.1)' can't be established.
RSA key fingerprint is e5:3c:7d:41:71:0b:0f:2a:0c:0e:bb:15:4d:e7:2f:08
.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'localhost' (RSA) to the list of known host
s.
list of files
$ exit
#
```

VNC Proxy Server (HTML5)

Install

```
# apt-get install -y  hubzero-vncproxyd-ws
```

Configure

```
# hzvncproxyd-ws-config configure --enable
```

The configuration process will try to autodiscover what SSL certificate to use. If you have a custom SSL certificate you can either specify it on the command line (the files must be readable by user 'hzvncproxy'):

```
# hzvncproxyd-ws-config configure --enable --ssl-cert [cert-filename] --ssl-key [key-filename]
```

or you can copy them to files /etc/hzvncproxyd-ws/ssl-cert-hzvncproxyd-ws.pem and /etc/hzvncproxyd-ws/ssl-cert-hzvncproxyd-ws.key and make sure they are readable by the user "hzvncproxy" and they will be found automatically.

If you are using a self-signed or otherwise invalid certificate the tool viewer will likely reject it and not work. If you are using the same certificate as your website and you allowed Chrome to use the invalid cert then the tool viewer will probably accept it. If you are using Firefox the tool viewer will always reject the invalid certificate. Always use a valid SSL certificate with hzvncproxyd-ws.

VNC Proxy Server (Java)

Install

```
# apt-get install -y  hubzero-vncproxy
```

Configure

```
# hzcms configure vncproxy --enable
```

telequotad

install

```
# apt-get install -y hubzero-telequotad
```

Configure

In order for filesystems quotas to work they must be enabled when they are mounted. Determine which filesystem contains your home directories and add "quota" to the mount option of the corresponding entry in the /etc/fstab file. Only the filesystem with /home on it matters to telequotad.

If quotas weren't already in affect, the run something like the following (depending on your filesystem configuration) to start up the quota system.

```
# mount -oremount /
# /etc/init.d/quota restart
# hzcms configure telequotad --enable
```

Test

```
# repquota -a
```

Should show disk usage for all users.

```
# apt-get install telnet
# telnet localhost 300
getquota user=admin
status=good,softspace=0,hardspace=0,space=4096,files=1,remaining=0
Connection closed by foreign host.
#
```

Workspace

Install

```
# apt-get install hubzero-app
# apt-get install hubzero-app-workspace
# hubzero-
app install --publish /usr/share/hubzero/apps/workspace-1.3.hza
```

Test

You should then be able to log in to the site and see the "Workspace" tool in the tool list and launch it in your browser.

Metrics

Install

```
# apt-get install hubzero-metrics
```

Configure

```
# hzcms configure metrics --enable
```

Rappture

Install

Install rappture for Debian 7 session containers

```
# apt-get install hubzero-rappture-deb7
```

Configure

Rappture is used from inside a container and needs several other packages installed to allow use of all its features. This process has been simplified by using the hubzero-rappture-session which only contains the dependencies needed to pull in these other packages.

```
# chroot /var/lib/vz/template/debian-7.0-amd64-maxwell
# apt-get update
# apt-get upgrade
# apt-get install hubzero-rappture-session
# exit
```

A workspace may need to be opened and closed a few times before the changes to the session template appear in a workspace.

Test

A user must setup their runtime environment in order to use the Rappture toolkit. Run the following command before attempting to run any Rappture tests.

```
use rappture
```

Rappture comes with several demostration scripts that can effectively test many parts of the package. These demonstrations must be copied to a user's home directory within a workspace before running.

```
$ mkdir examples
$ cp -r /apps/share/rappture/examples/* examples/.
$ cd examples
$ ./demo.bash
```

A window should open on the workspace showing that part of the demonstration. Close that window to see the next demonstration. Some demonstrations may need something inputted to work properly (such as the graphing calculator).

Filexfer

Install

```
# apt-get install -y hubzero-filexfer-xlate
```

Configure

```
# hzcms configure filexfer --enable
```

Submit

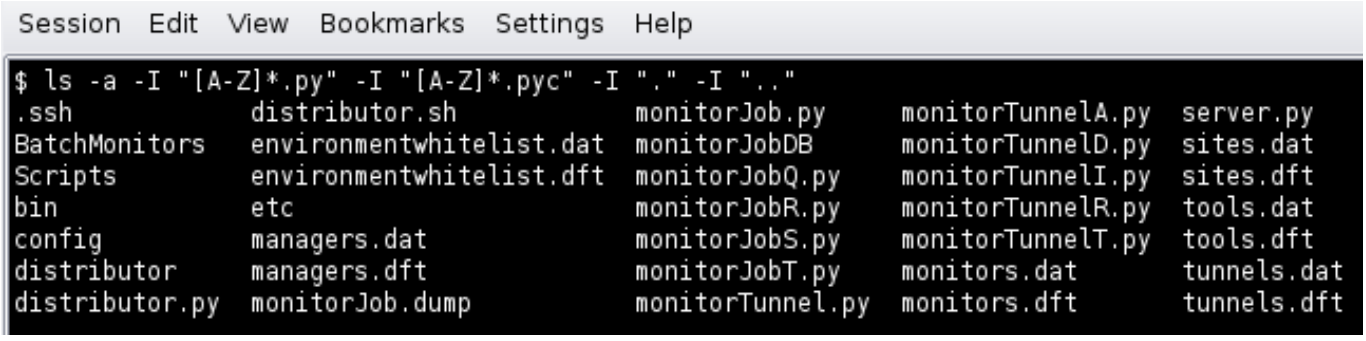
Introduction

The submit command provides a means for HUB end users to execute applications on remote resources. The end user is not required to have knowledge of remote job submission mechanics. Jobs can be submitted to traditional queued batch systems including PBS and Condor or executed directly on remote resources.

Installation

```
# apt-get install hubzero-submit-pegasus
# apt-get install hubzero-submit-condor
# apt-get install hubzero-submit-common
# apt-get install hubzero-submit-server
# apt-get install hubzero-submit-distributor
# apt-get install hubzero-submit-monitors
# hzcms configure submit-server --enable
# /etc/init.d/submit-server start
```

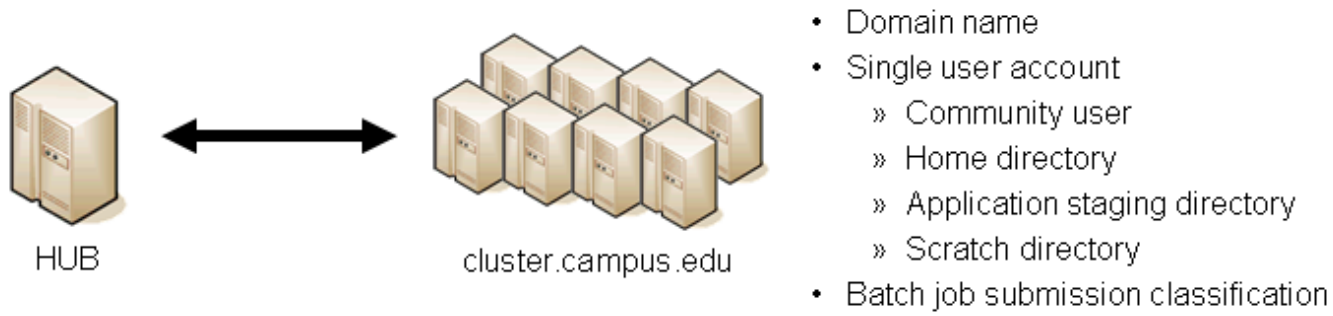
At completion of the apt-get install commands several files will be located in the directory /opt/submit. Excluding python files, the directory listing should like the following:



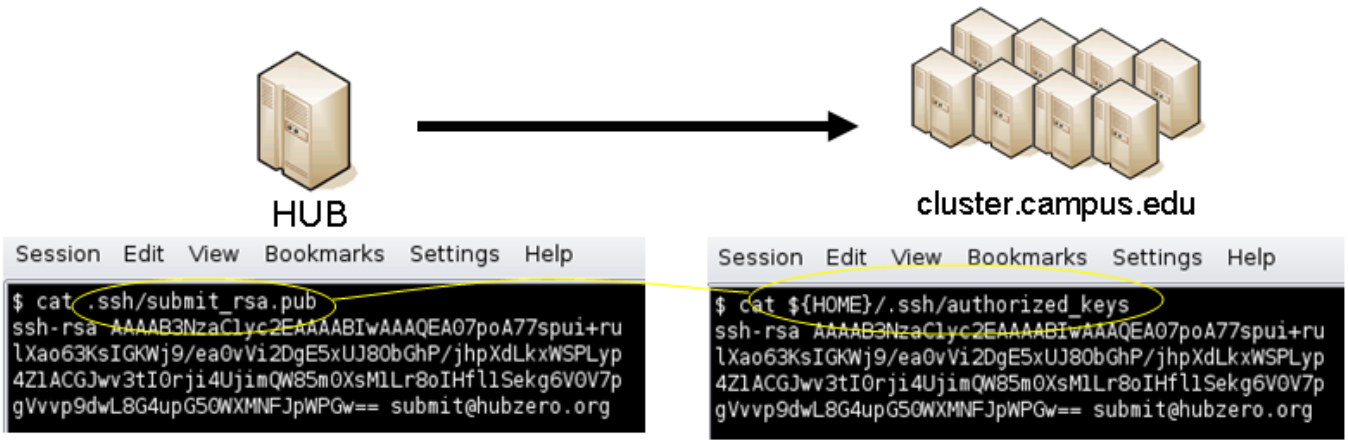
Configuration

Submit provides a mechanism to execute jobs on machines outside the HUB domain. To accomplish this feat, some configuration is required on the HUB and some additional software

must be installed and configured on hosts in remote domains. Before attempting to configure submit it is necessary to obtain access to the target remote domain(s). The premise is that a single account on the remote domain will serve as an execution launch point for all HUB end users. It further assumes that access to this account can be made by direct ssh login or using an ssh tunnel (port forwarding).



Having attained account access to one or more remote domains it is possible to proceed with submit configuration. To get started, the ssh public generated by the installation should be transferred to the remote domain host(s).



HUB Configuration

The behavior of submit is controlled through a set of configuration files. The configuration files contain descriptions of the various parameters required to connect to a remote domain, exchange files, and execute simulation codes. There are separate files for defining remote sites, staged tools, multiprocessor managers, file access controls, permissible environment variables, remote job monitors, and ssh tunneling. Most parameters have default values and it is not required that all parameters be explicitly defined in the configuration files. A simple example is given for each category of configuration file.

- Remote sites
- Staged tools
- Remote job monitors
- Multi-processor managers
- Permissible environment variables
- ssh tunnels

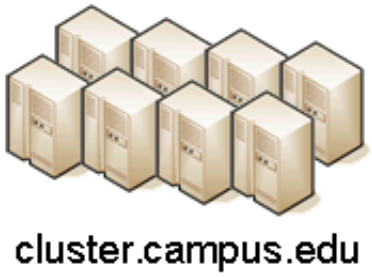
Sites

Remote sites are defined in the file sites.dat. Each remote site is defined by a stanza indicating an access mechanism and other account and venue specific information. Defined keywords are

- [name] - site name. Used as command line argument (-v/--venue) and in tools.dat (destinations)
- venues - comma separated list of hostnames. If multiple hostnames are listed one site will chosen at random.
- tunnelDesignator - name of tunnel defined in tunnels.dat.
- siteMonitorDesignator - name of site monitor defined in monitors.dat.
- venueMechanism - possible mechanisms are ssh and local.
- remoteUser - login user at remote site.
- remoteBatchAccount - some batch systems requirement that an account be provided in addition to user information.
- remoteBatchSystem - the possible batch submission systems include CONDOR, PBS, SGE, and LSF. SCRIPT may also be specified to specify that a script will be executed directly on the remote host.
- remoteBatchQueue - when remoteBatchSystem is PBS the queue name may be specified.
- remoteBatchPartition - slurm parameter to define partition for remote job
- remoteBatchPartitionSize - slurm parameter to define partition size, currently for BG machines.
- remoteBatchConstraints - slurm parameter to define constraints for remote job
- parallelEnvironment - sge parameter
- remoteBinDirectory - define directory where shell scripts related to the site should be kept.
- remoteApplicationRootDirectory - define directory where application executables are located.
- remoteScratchDirectory - define the top level directory where jobs should be executed. Each job will create a subdirectory under remoteScratchDirectory to isolated jobs from each other.
- remotePpn - set the number of processors (cores) per node. The PPN is applied to PBS and LSF job description files. The user may override the value defined here from the command line.
- remoteManager - site specific multi-processor manager. Refers to definition in

managers.dat.
- remoteHostAttribute - define host attributes. Attributes are applied to PBS description files.
- stageFiles - A True/False value indicating whether or not files should be staged to remote site. If the the job submission host and remote host share a file system file staging may not be necessary. Default is True.
- passUseEnvironment - A True/False value indicating whether or not the HUB 'use' environment should passed to the remote site. Default is False. True only makes sense if the remote site is within the HUB domain.
- arbitraryExecutableAllowed - A True/False value indicating whether or not execution of arbitrary scripts or binaries are allowed on the remote site. Default is True. If set to False the executable must be staged or emanate from /apps. (deprecated)
- executableClassificationsAllowed - classifications accepted by site. Classifications are set in appaccess.dat
- members - a list of site names. Providing a member list gives a layer of abstraction between the user facing name and a remote destination. If multiple members are listed one will be randomly selected for each job.
- state - possible values are enabled or disabled. If not explicitly set the default value is enabled.
- failoverSite - specify a backup site if site is not available. Site availability is determined by site probes.
- checkProbeResult - A True/False value indicating whether or not probe results should determine site availability. Default is True.
- restrictedToUsers - comma separated list of user names. If the list is empty all users may garner site access. User restrictions are applied before group restrictions.
- restrictedToGroups - comma separated list of group names. If the list is empty all groups may garner site access.
- logUserRemotely - maintain log on remote site mapping HUB id, user to remote batch job id. If not explicitly set the default value is False.
- undeclaredSiteSelectionWeight - used when no site is specified to choose between sites where selection weight > 0.
- minimumWallTime - minimum walltime allowed for site or queue. Time should be expressed in minutes.
- maximumWallTime - maximum walltime allowed for site or queue. Time should be expressed in minutes.
- minimumCores - minimum number of cores allowed for site or queue.
- maximumCores - maximum number of cores allowed for site or queue.
- pegasusTemplates - pertinent pegasus templates for site, rc, and transaction files.

An example stanza is presented for a site that is accessed through ssh.




```
Session  Edit  View  Bookmarks  Settings  Help
$ hostname -f
cluster.campus.edu
$ whoami
yourhub
$ echo ${HOME}
/home/yourhub
$ printenv | SCRATCH
CLUSTER_SCRATCH=/scratch/yourhub
```

```
[cluster]
venues = cluster.campus.edu
remotePpn = 8
remoteBatchSystem = PBS
remoteBatchQueue = standby
remoteUser = yourhub
remoteManager = mpich-intel64
venueMechanism = ssh
remoteScratchDirectory = /scratch/yourhub
siteMonitorDesignator = clusterPBS
```

Tools

Staged tools are defined in the file tools.dat. Each staged tool is defined by a stanza indicating where a tool is staged and any access restrictions. The existence of a staged tool at multiple sites can be expressed with multiple stanzas or multiple destinations within a single stanza. If the tool requires multiprocessors, a manager can also be indicated. Defined keywords are

- [name] - tool name. Used as command line argument to execute staged tools. Repeats are permitted to indicate staging at multiple sites.
- destinations - comma separated list of destinations. Destination may exist in sites.dat or be a grid site defined by a ClassAd file.
- executablePath - path to executable at remote site. The path may be given as an absolute path on the remote site or a path relative to remoteApplicationRootDirectory defined in sites.dat.
- restrictedToUsers - comma separated list of user names. If the list is empty all users may garner tool access. User restrictions are applied before group restrictions.
- restrictedToGroups - comma separated list of group names. If the list is empty all groups may garner tool access.
- environment - comma separated list of environment variables in the form e=v.
- remoteManager - tool specific multi-processor manager. Refers to definition in managers.dat. Overrides value set by site definition.
- state - possible values are enabled or disabled. If not explicitly set the default value is

enabled.

An example stanza is presented for a staged tool maintained in the yourhub account on a remote site.


```
Session  Edit  View  Bookmarks  Settings  Help
$ cd ${HOME}
$ ls -R
.:
apps

./apps:
planets  stars

./apps/planets:
bin

./apps/planets/bin:
earth.x  jupiter.x  mars.x  mercury.x  neptune.x  saturn.x  uranus.x  venus.x

./apps/stars:
bin

./apps/stars/bin:
antares.x  betelgeuse.x  polaris.x  sun.x
```

```
[earth]
destinations = cluster
executablePath = ${HOME}/apps/planets/bin/earth.x
remoteManager = mpich-intel

[sun]
destinations = cluster
executablePath = ${HOME}/apps/stars/bin/sun.x
remoteManager = mpich-intel
```

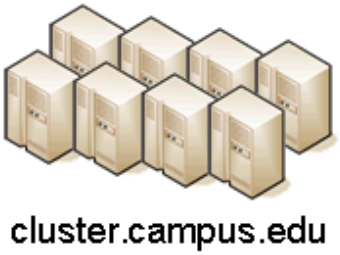
Monitors

Remote job monitors are defined in the file monitors.dat. Each remote monitor is defined by a stanza indicating where the monitor is located and to be executed. Defined keywords are

- [name] - monitor name. Used in sites.dat (siteMonitorDesignator)
- venue - hostname upon which to launch monitor daemon. Typically this is a cluster headnode.
- venueMechanism - monitoring job launch process. The default is ssh.
- tunnelDesignator - name of tunnel defined in tunnels.dat.

- remoteUser - login user at remote site.
- remoteBinDirectory - define directory where shell scripts related to the site should be kept.
- remoteMonitorCommand - command to launch monitor daemon process.
- state - possible values are enabled or disabled. If not explicitly set the default value is enabled.

An example stanza is presented for a remote monitor tool used to report status of PBS jobs.




```
Session  Edit  View  Bookmarks  Settings  Help
$ qstat -u yourhub

cluster.campus.edu:
                                                                  Req'd   Elap
Job ID          Username Queue   Jobname SessID NDS TSK Time  S Time
--------------- -------- ------- ------- ------ --- --- ----- - -----
9823388.steele- yourhub  standby earth    3508   2  16  04:00 R 02:15
9824065.steele- yourhub  standby sun      9975   1   1  04:00 R 01:26
```

```
[clusterPBS]
venue = cluster.campus.edu
remoteUser = yourhub
remoteMonitorCommand = ${HOME}/SubmitMonitor/monitorPBS.py
```

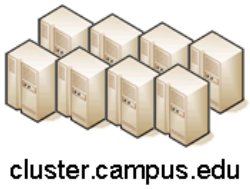
Multi-processor managers

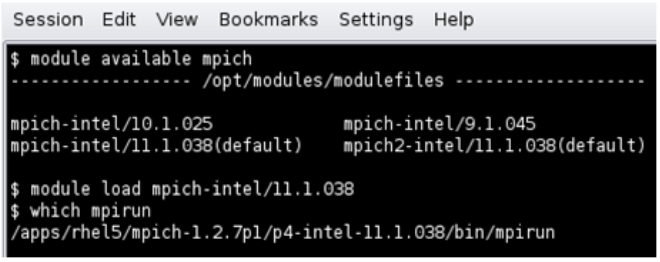
Multiprocessor managers are defined in the file managers.dat. Each manager is defined by a stanza indicating the set of commands used to execute a multiprocessor simulation run. Defined keywords are

- [name] - manager name. Used in sites.dat and tools.dat.
- computationMode - indicate how to use multiple processors for a single job. Recognized values are mpi, parallel, and matlabmpi. Parallel application request multiprocess have there own mechanism for inter process communication. Matlabmpi is used to enable the an Matlab implementation of MPI.
- preManagerCommands - comma separated list of commands to be executed before the manager command. Typical use of pre manager commands would be to define the environment to include a particular version of MPI amd/or compiler, or setup MPD.
- managerCommand - manager command commonly mpirun. It is possible to include strings that will be sustituted with values defined from the command line.
- postManagerCommands - comma separated list of commands to be executed when the manager command completes. A typical use would be to terminate an MPD setup.

- mpiRankVariable - define environment variable set by manager command to define process rank. Recognized values are: MPIRUN_RANK, GMPI_ID, RMS_RANK, MXMPI_ID, MSTI_RANK, PMI_RANK, and OMPI_MCA_ns_nds_vpid. If no variable is given an attempt is made to determine process rank from command line arguments.
- environment - comma separated list of environment variables in the form e=v.
- moduleInitialize - initialize module script for sh
- modulesUnload - modules to be unloaded clearing way for replacement modules
- modulesLoad - modules to load to define mpi and other libraries
- state - possible values are enabled or disabled. If not explicitly set the default value is enabled.

An example stanza is presented for a typical MPI instance. The given command should be suitable for /bin/sh execution.






```
[mpich-intel]
preManagerCommands = . ${MODULESHOME}/init/sh, module load mpich-
intel/11.1.038
managerCommand = mpirun -machinefile ${PBS_NODEFILE} -np NPROCESSORS
```

The token NPROCESSORS is replaced by an actual value at runtime.

File access controls

Application or file level access control is described by entries listed in the file appaccess.dat. The ability to transfer files from the HUB to remote sites is granted on a group basis as defined by white and black lists. Each list is given a designated priority and classification. In cases where a file appears on multiple lists, the highest priority takes precedence. Simple wildcard operators are allowed the in the filename declaration allowing for easy listing of entire directories. Each site lists acceptable classification(s) in sites.dat. Defined keywords are

- [group] - group name.
- whitelist - comma separated list of paths. Wildcards allowed.

- blacklist - comma separated list of paths. Wildcards allowed.
- priority - higher priority wins
- classification - apps or user. user class are treated are arbitrary executables.
- state - possible values are enabled or disabled. If not explicitly set the default value is enabled.

An example file giving permissions reminiscent of those defined in earlier submit releases is presented here

```
[public]
whitelist = /apps/.*
priority = 0
classification = apps

[submit]
whitelist = ${HOME}/.*
priority = 0
classification = home
```

The group public is intended to include all users. Your system may use a different group such as users for this purpose. The definitions shown here allow all users access to files in /apps where applications are published. Additionally members of the submit group are allowed to send files from their $HOME directory.

Environment variables

Legal environment variables are listed in the file environmentwhitelist.dat. The objective is to prevent end users from setting security sensitive environment variables while allowing application specific variables to be passed to the remote site. Environment variables required to define multiprocessor execution should also be included. The permissible environment variables should be entered as a simple list - one entry per line. An example file allowing use of a variables used by openmp and mpich is presenter here.

```
# environment variables listed here can be specified from the command
line with -e/--env option. Attempts to specify other environment varia
bles will be ignored and the values will not be passed to the remote s
ite.
```

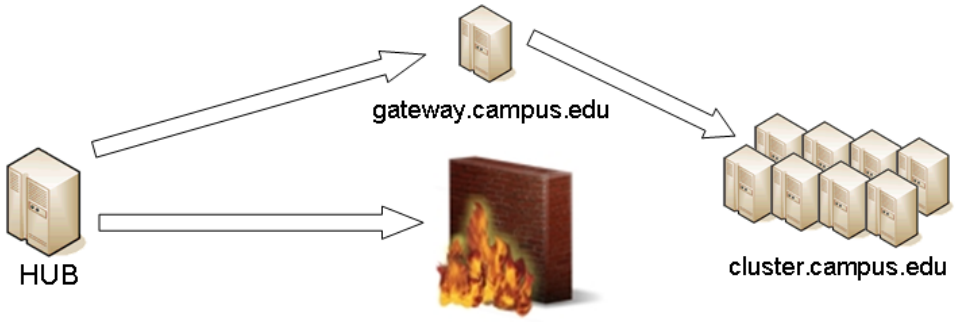
```
OMP_NUM_THREADS
MPICH_HOME
```

Tunnels

In some circumstances, access to clusters is restricted such that only a select list of machines is allowed to communicate with the cluster job submission node. The machines that are granted such access are sometimes referred to as gateways. In such circumstances, ssh tunneling or port forwarding can be used to submit HUB jobs through the gateway machine. Tunnel definition is specified in the file tunnels.dat. Each tunnel is defined by a stanza indicating gateway host and port information. Defined keywords are

- [name] - tunnel name.
- venue - tunnel target host.
- venuePort - tunnel target port.
- gatewayHost - name of the intermediate host.
- gatewayUser - login user on gatewayHost.
- localPortOffset - local port offset used for forwarding. Actual port is localPortMinimum + localPortOffset

An example stanza is presented for a tunnel between the HUB and a remote venue by way of an accepted gateway host.




```
[cluster]
venue = cluster.campus.edu
venuePort = 22
gatewayHost = gateway.campus.edu
gatewayUser = yourhub
localPortOffset = 1
```

Initialization Scripts and Log Files

The submit server and job monitoring server must be started as daemon processes running on the the submit host. If ssh tunneling is going to be used an addition server must be started as a daemon process. Each daemon process writes to a centralized log file facilitating error recording and debugging.

Initialize daemon scripts

Scripts for starting the server daemons are provided and installed in /etc/init.d. The default settings for when to start and terminate the scripts are adequate.

Log files

Submit processes log information to files located in the /var/log/submit directory tree. The exact location varies depending on the vintage of the installation. Each process has its own log file. The three most important log files are submit-server.log, distributor.log, and monitorJob.log.

submit.log

The submit-server.log file tracks when the submit server is started and stopped. Each connection from the submit client is logged with the command line and client ip address reported. All log entries are timestamped and reported by submit-server process ID (PID) or submit ID (ID:) once one has been assigned. Entries from all jobs are simultaneously reported and intermingled. The submit ID serves as a good search key when tracing problems. Examples of startup, job execution, and termination are given here. The job exit status and time metrics are also recorded in the MyQSL database JobLog table.

```
[Sun Aug 26 17:28:24 2012] 0: ########################################
###########
[Sun Aug 26 17:28:24 2012] 0: Backgrounding process.
[Sun Aug 26 17:28:24 2012] 0: Listening: protocol='tcp', host='', port
=830
```

```
[Sun Sep 23 12:33:28 2012] (1154) ==================================
===============
```

```
[Sun Sep 23 12:33:28 2012] (1154) Connection to tcp://:830 from ('192.
168.224.14', 38770)
[Sun Sep 23 12:33:28 2012] 0: Server will time out in 60 seconds.
[Sun Sep 23 12:33:28 2012] 0: Cumulative job load is 0.84.  (Max: 510.
00)
[Sun Sep 23 12:33:28 2012] 1670: Args are:['/usr/bin/submit', '--local
', '-p', '@@iv=-3:1.5:3', '/home/hubzero/user/hillclimb/bin/hillclimb1
.py', '--seed', '10', '--initialvalue', '@@iv', '--lowerbound', '-3',
'--upperbound', '3', '--function', 'func2', '--solutionslog', 'solutio
ns.dat', '--bestresultlog', 'best.dat']
[Sun Sep 23 12:33:28 2012] 1670: Server stopping.
[Sun Sep 23 12:33:28 2012] 1670: Server(JobExecuter) exiting(2).
[Sun Sep 23 12:33:38 2012] (1154) ================================
================
[Sun Sep 23 12:33:38 2012] (1154) Connection to tcp://:830 from ('192.
168.224.14', 38774)
[Sun Sep 23 12:33:38 2012] 0: Server will time out in 60 seconds.
[Sun Sep 23 12:33:38 2012] 1670: Job Status: venue=1:local status=0 cp
u=0.030000 real=0.000000 wait=0.000000
[Sun Sep 23 12:33:38 2012] 1670: Job Status: venue=2:local status=0 cp
u=0.040000 real=0.000000 wait=0.000000
[Sun Sep 23 12:33:38 2012] 1670: Job Status: venue=3:local status=0 cp
u=7.050000 real=7.000000 wait=0.000000
[Sun Sep 23 12:33:38 2012] 1670: Job Status: venue=4:local status=0 cp
u=0.080000 real=0.000000 wait=0.000000
[Sun Sep 23 12:33:38 2012] 1670: Job Status: venue=5:local status=0 cp
u=0.020000 real=1.000000 wait=0.000000
[Sun Sep 23 12:33:38 2012] 1670: Job Status: venue= status=0 cpu=10.42
8651 real=9.561828 wait=0.000000
[Sun Sep 23 12:33:38 2012] 1670: Server(JobExecuter) exiting(0).
[Sun Sep 23 12:48:44 2012] (1154) ================================
================
```

```
[Sun Aug 26 17:28:17 2012] 0: Server(10836) was terminated by a signal
 2.
[Sun Aug 26 17:28:17 2012] 0: Server(Listener) exiting(130).
```

distributor.log

The distributor.log file tracks each job as it progresses from start to finish. Details of remote site

assignment, queue status, exit status, and command execution are all reported. All entries are timestamped and reported by submit ID. The submit ID serves as the key to join data reported in submit-server.log. An example for submit ID 1659 is listed here. Again the data for all jobs are intermingled.

```
[Sun Sep 23 00:04:21 2012] 0: quotaCommand = quota -w | tail -n 1
[Sun Sep 23 00:04:21 2012] 1659: command = tar vchf 00001659_01_input.
tar --exclude='*.svn*' -C /home/hubzero/user/data/sessions/3984L .__lo
cal_jobid.00001659_01 sayhiinquire.dax
[Sun Sep 23 00:04:21 2012] 1659: remoteCommand pegasus-
plan --dax ./sayhiinquire.dax
[Sun Sep 23 00:04:21 2012] 1659: workingDirectory /home/hubzero/user/d
ata/sessions/3984L
[Sun Sep 23 00:04:21 2012] 1659: command = tar vrhf 00001659_01_input.
tar --exclude='*.svn*' -C /home/hubzero/user/data/sessions/3984L/00001
659/01 00001659_01.sh
[Sun Sep 23 00:04:21 2012] 1659: command = nice -n 19 gzip 00001659_01
_input.tar
[Sun Sep 23 00:04:21 2012] 1659: command = /opt/submit/bin/receiveinpu
t.sh /home/hubzero/user/data/sessions/3984L/00001659/01 /home/hubzero/
user/data/sessions/3984L/00001659/01/.__timestamp_transferred.00001659
_01
[Sun Sep 23 00:04:21 2012] 1659: command = /opt/submit/bin/submitbatch
job.sh /home/hubzero/user/data/sessions/3984L/00001659/01 ./00001659_0
1.pegasus
[Sun Sep 23 00:04:23 2012] 1659: remoteJobId = 2012.09.23 00:04:22.996
 EDT:   Submitting job(s).
2012.09.23 00:04:23.002 EDT:   1 job(s) submitted to cluster 946.
2012.09.23 00:04:23.007 EDT:
2012.09.23 00:04:23.012 EDT:   ----------------------------------------
--------------------------------
2012.09.23 00:04:23.017 EDT:   File for submitting this DAG to Condor
          : sayhi_inquire-0.dag.condor.sub
2012.09.23 00:04:23.023 EDT:   Log of DAGMan debugging messages
          : sayhi_inquire-0.dag.dagman.out
2012.09.23 00:04:23.028 EDT:   Log of Condor library output
          : sayhi_inquire-0.dag.lib.out
2012.09.23 00:04:23.033 EDT:   Log of Condor library error messages
          : sayhi_inquire-0.dag.lib.err
2012.09.23 00:04:23.038 EDT:   Log of the life of condor_dagman itself
          : sayhi_inquire-0.dag.dagman.log
2012.09.23 00:04:23.044 EDT:
2012.09.23 00:04:23.049 EDT:   ----------------------------------------
--------------------------------
2012.09.23 00:04:23.054 EDT:
2012.09.23 00:04:23.059 EDT:   Your Workflow has been started and runs
```

```
 in base directory given below
2012.09.23 00:04:23.064 EDT:
2012.09.23 00:04:23.070 EDT:   cd /home/hubzero/user/data/sessions/398
4L/00001659/01/work/pegasus
2012.09.23 00:04:23.075 EDT:
2012.09.23 00:04:23.080 EDT:   *** To monitor the workflow you can run
 ***
2012.09.23 00:04:23.085 EDT:
2012.09.23 00:04:23.090 EDT:   pegasus-status -l /home/hubzero/user/da
ta/sessions/3984L/00001659/01/work/pegasus
2012.09.23 00:04:23.096 EDT:
2012.09.23 00:04:23.101 EDT:   *** To remove your workflow run ***
2012.09.23 00:04:23.106 EDT:   pegasus-remove /home/hubzero/user/data/
sessions/3984L/00001659/01/work/pegasus
2012.09.23 00:04:23.111 EDT:
2012.09.23 00:04:23.117 EDT:   Time taken to execute is 0.993 seconds
[Sun Sep 23 00:04:23 2012] 1659: confirmation: S(1):N Job
[Sun Sep 23 00:04:23 2012] 1659: status:Job N WF-DiaGrid
[Sun Sep 23 00:04:38 2012] 1659: status:DAG R WF-DiaGrid
[Sun Sep 23 00:10:42 2012] 0: quotaCommand = quota -w | tail -n 1
[Sun Sep 23 00:10:42 2012] 1660: command = tar vchf 00001660_01_input.
tar --exclude='*.svn*' -C /home/hubzero/clarksm .__local_jobid.0000166
0_01 noerror.sh
[Sun Sep 23 00:10:42 2012] 1660: remoteCommand ./noerror.sh
[Sun Sep 23 00:10:42 2012] 1660: workingDirectory /home/hubzero/clarks
m
[Sun Sep 23 00:10:42 2012] 1660: command = tar vrhf 00001660_01_input.
tar --exclude='*.svn*' -C /home/hubzero/clarksm/00001660/01 00001660_0
1.sh
[Sun Sep 23 00:10:42 2012] 1660: command = nice -n 19 gzip 00001660_01
_input.tar
[Sun Sep 23 00:10:42 2012] 1660: command = /opt/submit/bin/receiveinpu
t.sh /home/hubzero/clarksm/00001660/01 /home/hubzero/clarksm/00001660/
01/.__timestamp_transferred.00001660_01
[Sun Sep 23 00:10:42 2012] 1660: command = /opt/submit/bin/submitbatch
job.sh /home/hubzero/clarksm/00001660/01 ./00001660_01.condor
[Sun Sep 23 00:10:42 2012] 1660: remoteJobId = Submitting job(s).
1 job(s) submitted to cluster 953.
[Sun Sep 23 00:10:42 2012] 1660: confirmation: S(1):N Job
[Sun Sep 23 00:10:42 2012] 1660: status:Job N DiaGrid
[Sun Sep 23 00:11:47 2012] 1660: status:Simulation I DiaGrid
[Sun Sep 23 00:12:07 2012] 1660: Received SIGINT!
[Sun Sep 23 00:12:07 2012] 1660: waitForBatchJobs: nCompleteRemoteJobI
ndexes = 0, nIncompleteJobs = 1, abortGlobal = True
[Sun Sep 23 00:12:07 2012] 1660: command = /opt/submit/bin/killbatchjo
b.sh 953.0 CONDOR
```

```
[Sun Sep 23 00:12:07 2012] 1660: Job 953.0 marked for removal

[Sun Sep 23 00:12:07 2012] 1660: status:Simulation I DiaGrid
[Sun Sep 23 00:12:52 2012] 1660: status:Simulation D DiaGrid
[Sun Sep 23 00:12:52 2012] 1660: venue=1:localCONDOR:953.0:DiaGrid sta
tus=258 cputime=0.000000 realtime=0.000000 waittime=0.000000 ncpus=1
[Sun Sep 23 00:28:14 2012] 1659: status:DAG D WF-DiaGrid
[Sun Sep 23 00:28:14 2012] 1659: waitForBatchJobs: nCompleteRemoteJobI
ndexes = 1, nIncompleteJobs = 0, abortGlobal = False
[Sun Sep 23 00:28:14 2012] 1659: command = /opt/submit/bin/cleanupjob.
sh /home/hubzero/user/data/sessions/3984L/00001659/01
[Sun Sep 23 00:28:15 2012] 1659:
*****************************************SUMMARY*****************
*****************************

Job instance statistics           : /home/hubzero/user/data/sessions/3
984L/00001659/01/work/pegasus/statistics/jobs.txt

****************************************************************************
*****************************

[Sun Sep 23 00:28:15 2012] 1659: venue=1:localPEGASUS:946.0:WF-DiaGrid
 status=0 cputime=1.430000 realtime=2.000000 waittime=0.000000 ncpus=1
[Sun Sep 23 00:28:15 2012] 1659: venue=2:PEGASUS:952.0:DiaGrid status=
0 cputime=0.003000 realtime=0.000000 waittime=681.000000 ncpus=1 event
=/sayhi_inquire-sayhi-1.0
[Sun Sep 23 00:28:15 2012] 1659: venue=3:PEGASUS:954.0:DiaGrid status=
0 cputime=0.003000 realtime=0.000000 waittime=631.000000 ncpus=1 event
=/sayhi_inquire-inquire-1.0
```

monitorJob.log

The monitorJob.log file tracks the invocation and termination of each remotely executed job monitor. The remote job monitors are started on demand when job are submitted to remote sites. The remote job monitors terminate when all jobs complete at a remote site and no new activity has been initiated for a specified amount of time - typically thirty minutes. A typical report should look like:

```
[Sun Aug 26 17:29:16 2012] (1485) ***********************************
[Sun Aug 26 17:29:16 2012] (1485) * distributor job monitor started *
[Sun Aug 26 17:29:16 2012] (1485) ***********************************
[Sun Aug 26 17:29:16 2012] (1485) loading active jobs
[Sun Aug 26 17:29:16 2012] (1485) 15 jobs loaded from DB file
```

```
[Sun Aug 26 17:29:16 2012] (1485) 15 jobs loaded from dump file
[Sun Aug 26 17:29:16 2012] (1485) 4 jobs purged
[Sun Aug 26 17:29:16 2012] (1485) 11 monitored jobs
[Sun Aug 26 18:02:04 2012] (24250) Launching wf-diagrid
[Sun Aug 26 18:02:04 2012] (1485) 12 monitored jobs
[Sun Aug 26 18:02:15 2012] (1485) Update message received from wf-
diagrid
[Sun Aug 26 18:03:15 2012] (1485) Update message received from wf-
diagrid
[Sun Aug 26 18:06:43 2012] (1485) 13 monitored jobs
...
[Thu Sep 17 17:32:51 2011] (21095) Received SIGTERM!
[Thu Sep 17 17:32:51 2011] (21095) Send TERM to child ssh process
[Thu Sep 17 17:32:51 2011] (21095) distributor site monitor stopped
[Thu Sep 17 17:32:51 2011] (17348) Send TERM to child site steele proc
ess
[Thu Sep 17 17:32:51 2011] (17348) **********************************
[Thu Sep 17 17:32:51 2011] (17348) * distributor job monitor stopped *
[Thu Sep 17 17:32:51 2011] (17348) **********************************
```

It is imperative that the job monitor be running in order for notification of job progress to occur. If users report that their job appears to hang check to make sure the job monitor is running. If necessary take corrective action and restart the daemon.

monitorTunnel.log

The monitorTunnel.log file tracks invocation and termination of each ssh tunnel connection. If users report problems with job submission to sites accessed via an ssh tunnel this log file should be checked for indication of any possible problems.

Remote Domain Configuration

For job submission to remote sites via ssh it is necessary to configure a remote job monitor and a set of scripts to perform file transfer and batch job related functions. A set of scripts can be used for each different batch submission system or in some cases they may be combined with appropriate switching based on command line arguments. A separate job monitor is need for each batch submission system. Communication between the HUB and remote resource via ssh

Job monitor daemon

A remote job monitor runs a daemon process and reports batch job status to a central job monitor located on the HUB. The daemon process is started by the central job monitor on demand. The daemon terminates after a configurable amount of inactivity time. The daemon code needs to be installed in the location declared in the monitors.dat file. The daemon requires some initial configuration to declare where it will store log and history files. The daemon does not require any special privileges any runs as a standard user. Typical configuration for the daemon looks like this:

The directory defined by MONITORLOGLOCATION needs to be created before the daemon is started. Sample daemon scripts used for PBS, LSF, SGE, Condor, Load Leveler, and Slurm batch systems are included in directory BatchMonitors.

File transfer and batch job scripts

The simple scripts are used to manage file transfer and batch job launching and termination. The location of the scripts is entered in sites.dat.

Examples scripts suitable for use with PBS, LSF, Condor, Load Leveler, and Slurm are included in directory Scripts. After modifications are made to monitors.dat the central job monitor must be notified. This can be accomplished by stopping and starting the submon daemon or a HUP signal can be sent to the monitorJob.py process.

File transfer - input files

Receive compressed tar file containing input files required for the job on stdin. The file transferredTimestampFile is used to determine what newly created or modified files should be

returned to the HUB.

```
receiveinput.sh jobWorkingDirectory jobScratchDirectory  transferredTi
mestampFile
```

Batch job script - submission

Submit batch job using supplied description file. If arguments beyond job working directory and batch description file are supplied an entry is added to the remote site log file. The log file provides a record relating the HUB end user to the remote batch job identifier. The log file should be placed at a location agreed upon by the remote site and HUB.

```
submitbatchjob.sh jobWorkingDirectory jobScratchDirectory jobDescripti
onFile
```

The jobId is returned on stdout if job submission is successful. For an unsuccessful job submission the returned jobId should be -1.

File transfer - output files

Return compressed tar file containing job output files on stdout.

```
transmitresults.sh jobWorkingDirectory
```

File transfer - cleanup

Remove job specific directory and any other dangling files

```
cleanupjob.sh jobWorkingDirectory jobScratchDirectory jobClass
```

Batch job script - termination

Terminate given remote batch job. Command line arguments specify job identifier and batch system type.

```
killbatchjob.sh jobId jobClass
```

Batch job script - post process

For some jobClassses it is appropriate to preform standard post processing actions. An example of such a jobClass is Pegasus.

```
postprocessjob.sh jobWorkingDirectory jobScratchDirectory jobClass
```

Access Control Mechanisms

By default tools and sites are configured so that access is granted to all HUB members. In some cases it is desired to restrict access to either a tool or site to a subset of the HUB membership. The keywords restrictedToUsers and restrictedToGroups provide a mechanism to apply restrictions accordingly. Each keyword should be followed by a list of comma separated values of userids (logins) or groupids (as declared when creating a new HUB group). If user or group restrictions have been declared upon invocation of submit a comparison is made between the restrictions and userid and group memberships. If both user and group restrictions are declared the user restriction will be applied first, followed by the group restriction.

In addition to applying user and group restrictions another mechanism is provided by the executableClassificationsAllowed keyword in the sites configuration file. In cases where the executable program is not pre-staged at the remote sites the executable needs to be transferred along with the user supplied inputs to the remote site. Published tools will have their executable program located in the /apps/tools/revision/bin directory. For this reason submitted programs that reside in /apps are assumed to be validated and approved for execution. The same cannot be said for programs in other directories. The common case where such a situation arises is when a tool developer is building and testing within the HUB workspace environment. To grant a tool developer the permission to submit such arbitrary applications the site configuration must allow arbitrary executables and the tool developer must be granted permission to send files from their $HOME directory. Discrete permission can be granted on a file by file basis in appaccess.dat.

Add-ons

Introduction

Add-ons for HUBzero are available here. Currently these consist of a couple projects that have not yet been fully integrated into the the HUBzero packaging and installation process.

Separate execution host setup

Intro

Our standard open source install sets up a web server and execution host on a single machine. For smaller hubs, this is an adequate setup, however on hubs with greater needs, often separate (and sometimes multiple) execution hosts are required for an installation. The following is a set of directions for setting up an execution host.

Note, these directions are not complete step by step directions, but more of a guideline for setting up an execution host. A user should have a very thorough understanding of hub architecture before attempting to setup an additional execution host.

Setup steps

1. Do standard debian OS install
2. Configure hostname and /etc/sources.list appropiately
3. Setup standard hub through the openldap step
4. Install openvz kernel
5. Install hubzero mw-service on execution host 'apt-get install -y hubzero-mw-service'
6. Run 'mkvztemplate amd64 wheezy diego'
7. run 'hzcms configure mw-service --enable'
8. configure /etc/nslcd.conf and restart. /etc/nslcd.conf will need the following modifications:
   URI - modified to point to the ldap on the web host
   binddn - set to the search user dn on the webserver (do a 'slapcat | grep search' to get the DN for the search user on the web server)
   bindpw - set to the value contained for the LDAP-SEARCHPW in the /etc/hubzero.secrets file on the web server
9. Install hubzero mw-client and configure on execution host
10. copy /etc/mw-client/maxwell.key.pub from web host to /root/.ssh/authorized keys file on execution host
11. On web server, add execution host to tools component
    login to webserver admin section (webserver/administrator)
    select components->tools
    on host tab click on + sign in upper right to add an execution host
    When you are returned to the list of hosts, you should see two, one for the web server, likely called localhost and the IP for your execution host
    Under the provisions section, click on pubnet, sessions, and workspace for the new execution host
    Under the provisions section, uncheck everything but fileserver for your web server
12. Setup nfs server on web server
    'apt-get install nfs-kernel-server'

edit /etc/exports to export /home and /apps, something similar to this:
/home executionhost.ip.address(rw,no_subtree_check)
/apps executionhost.ip.address(rw,no_subtree_check)

13. Setup nfs client on execution host
apt-get install nfs-common
mount -t nfs webserver:/home /home
mount -t nfs webserver:/apps /apps
NOTE: user will want to add appropiate sections in the /etc/fstab file to remount these locations after a reboot. Something similar to:
your.webserver.org:/apps /apps nfs vers=3,rw 0 0
your.webserver.org:/home /home nfs vers=3,rw 0 0

InCommon

These instructions are for Debian 7 (wheezy) ONLY. They have not been updated for Debian 8

Introduction

This plugin provides some code necessary to allow your hub to accept credentials using the Shibboleth system. Most commonly, this implies membership in the InCommon network.

Shibboleth has some particular architectural demands, namely that it will install a new daemon and a new Apache module on your system. InCommon has some administrative demands, in that you will need to negotiate to get your hub added to their XML manifest as a service provider.

Installation

Shibboleth wiki entry on service provider installation

Debian

It is necessary on a Debian 7 Wheezy host to add the backports repository to install this the required packages, see https://backports.debian.org/Instructions/#index2h2

```
deb http://ftp.debian.org/debian wheezy-backports main
```

Then run the follow command to install and do some initial configuration for you such as enabling the module.:

```
apt-get -t wheezy-backports install libapache2-mod-shib2
```

```
aptitude install libapache2-mod-shib2
```

If your distribution does not have this package, refer to the Shibboleth wiki page above for information about other installation methods.

Redhat Enterprise Linux & other distributions

See the wiki page above for information on how to add the Shibboleth software to your list of repositories so that it can be installed and upgraded through yum, or, failing that, how to install from SRPMS.

Configuration

Shibboleth

Certificates

As root, run the script shib-keygen, which was installed as part of the package. This will generate a key pair for your service provider to use. No further configuration is required for this; the software will find the keys when the shibd service is restarted.

output

```
Generating a 2048 bit RSA private key
.....................................................................
.................+++
.....................+++
writing new private key to '/etc/shibboleth/sp-key.pem'
-----
```

/etc/shibboleth/attribute-map.xml

This file controls which attributes (bits of user information) the software will extract during login [when the identity provider makes them available](#).

Make sure the following pertinent attributes are not commented out in both forms of the “name” attribute.

eppn (username, probably already enabled in the shipped configuration):

```
    <Attribute name="urn:mace:dir:attribute-
def:eduPersonPrincipalName" id="eppn">
```

```
        <AttributeDecoder xsi:type="ScopedAttributeDecoder"/>
    </Attribute>
    <Attribute name="urn:oid:1.3.6.1.4.1.5923.1.1.1.6" id="eppn">
        <AttributeDecoder xsi:type="ScopedAttributeDecoder"/>
    </Attribute>
```

Name & email (probably not enabled by default):

```
    <Attribute name="urn:mace:dir:attribute-def:sn" id="sn"/>
    <Attribute name="urn:mace:dir:attribute-
def:givenName" id="givenName"/>
    <Attribute name="urn:mace:dir:attribute-
def:displayName" id="displayName"/>
    <Attribute name="urn:mace:dir:attribute-def:mail" id="mail"/>
    <Attribute name="urn:oid:2.5.4.4" id="sn"/>
    <Attribute name="urn:oid:2.5.4.42" id="givenName"/>
    <Attribute name="urn:oid:2.16.840.1.113730.3.1.241" id="displayNam
e"/>
    <Attribute name="urn:oid:0.9.2342.19200300.100.1.3" id="mail"/>
```

/etc/shibboleth/shibboleth2.xml

This is the main configuration, which controls how the software federates with identity providers.

First, replace $YOUR_HOSTNAME with, uh, your hostname, in the entityID attribute near the top of the file:

```
       <ApplicationDefaults entityID="https://$YOUR_HOSTNAME/login/shi
bboleth" REMOTE_USER="eppn persistent-id targeted-id">
```

In the block, delete or comment-out any SSO or SessionInitiator blocks that shipped, and add the two listed below, again interpolating your real hostname. This tells the software to check with the CMS plugin about where to redirect for a given authentication request, and allows the CMS to selectively enable providers.

```
        <Sessions lifetime="28800" timeout="3600" relayState="ss:mem"
                  checkAddress="true" handlerSSL="true" cookieProps="h
```

```
ttps">
            <SSO discoveryProtocol="SAMLDS" ECP="true" discoveryURL="h
ttps://$YOUR_HOSTNAME/login?authenticator=shibboleth&wayf">
                  SAML2 SAML1
            </SSO>
            <SessionInitiator type="Chaining" Location="/login/shibbol
eth" isDefault="true" id="Login">
                <SessionInitiator type="SAML2" template="bindingTempla
te.html"/>
                <SessionInitiator type="Shib1"/>
                <SessionInitiator type="SAMLDS" URL="https://$YOUR_HOS
TNAME/login?authenticator=shibboleth&wayf"/>
            </SessionInitiator>
            <!-- Default <Handler> tags not pictured, but they should
stay -->
       </Sessions>
```

If you run into issues where you seem to be stuck in a redirect loop between the idp and the sp, changing the cookie properties to use a less specific path may help.

```
cookieProps="; path=/; secure; HttpOnly"
```

If this is a production machine you will want to set a real email for the support contact:

```
       <Errors supportContact="support@$YOUR_HOSTNAME"
            helpLocation="/about.html"
            styleSheet="/shibboleth-sp/main.css"/>
```

Finally, you will need to configure how and where the software looks for metadata about identity providers. This is just a list of providers you can support, including some helpful annotations like where the service URLs and what public key to use when communicating with it.

Metadata provider: TestShib

For development and test machines it is often useful to use TestShib, and its configuration looks like this, below the Sessions tag and at the same scope:

```
      <MetadataProvider type="XML" uri="http://www.testshib.org/metada
ta/testshib-providers.xml" backingFilePath="testshib-two-idp-
metadata.xml" reloadInterval="180000"/>
```

Visit the TestShib site for more information about how to set this up, if you're interested. Hopefully you do not need the "Install" selection, but pick up from "Register". During "Configure" it recommends replacing your whole shibboleth2.xml with one it generated. Make a backup if you do, or else just add the MetadataProvider above to your existing configuration.

When you reach "Test", see below for the CMS configuration that will add TestShib to the list of available identity providers on your hub.

Metadata provider: InCommon

If your plans include membership in the InCommon consortium, this is the incantation, below the Sessions tag and at the same scope:

```
        <MetadataProvider type="XML" uri="https://wayf.incommonfederat
ion.org/InCommon/InCommon-metadata.xml" backingFilePath="federation-
metadata.xml" reloadInterval="7200">
            <MetadataFilter type="RequireValidUntil" maxValidityInterv
al="2419200"/>
            <MetadataFilter type="Signature" certificate="inc-md-
cert.pem"/>
        </MetadataProvider>
```

Install https://ds.incommon.org/certs/inc-md-cert.pem as /etc/shibboleth/inc-md-cert.pem so it's available for this provider.

Metadata provider: others?

If you are doing one-on-one negotiations with identity providers the metadata situation gets a bit more hairy, but the identity providers in question will probably be able to guide your configuration.

Apache

Quoth the Shibboleth wiki entry on service provider installation:

- UseCanonicalName On
- Ensure that the ServerName directive is properly set, and that Apache is being started with SSL enabled.

Make sure installing the software enabled both the module shib2 and the support daemon shibd.

Typically this means that there is a symlink /etc/apache2/mods-enabled/shib2.load that points to /etc/apache2/mods-available/shib2.load and that this report works:

```
# service shibd status
[ ok ] shibd is running.
```

/etc/apache2/sites-enabled/{your-ssl-enabled-config-file}

Your EntityID is something like https://hostname/login/shibboleth, but the actual URL to pick up the login process again in CMS terms is more complicated, so we rewrite it. I recommend putting this statement as high as possible in the config (after RewriteEngine on) so that the “L“ast last triggers and you can be assured the URL is not subsequently rewritten by anything else you’re doing.

```
        RewriteCond     %{REQUEST_URI}          ^/login/shibboleth
        RewriteRule     (.*) /index.php?option=com_users&authenticator
=shibboleth&task=user.login [NC,L]
```

Bind an endpoint to the module. This is used during the login process and is also useful to get a basis for your service provider’s metadata, which is served at /Shibboleth.sso/Metadata when the request comes from localhost.

```
        <Location /Shibboleth.sso>
                SetHandler shib
        </Location>
```

You probably have a rule that directs all requests that appear to be for CMS content to the index.php bootstrap, and we need to note that /Shibboleth.sso isn’t CMS business, so make

sure you have a RewriteCond like this:

```
        RewriteCond     %{REQUEST_URI}          !^/Shibboleth.sso/.*$
[NC]
        RewriteRule     (.*)                    index.php
```

Finally, we actually protect the entityID location /login/shibboleth. We can redirect a user to this path to require them to make a Shibboleth login. Shibboleth won't know specifically how to do that so it will make a request to the wayf location defined above in shibboleth2.xml. This is part of the CMS that knows already which provider the user selected from the login page, so it spits back the appropriate identity provider entityId. From there the metadata is referenced to find the endpoint associated with that institution, and the user is sent to the login page. They come back to /login/shibboleth upon submission, but now the requirement to have a Shibboleth session is satisified, and the rewritten URL referencing user.login is served to complete the process.

```
        <Location /login/shibboleth>
                AuthType shibboleth
                ShibRequestSetting requireSession 1
                Require valid-user
        </Location>
```

Restart the shibd and apache2 services when satisified with this configuration.

CMS

Plugin

Log in to /administrator, choose Extensions and then Plugin Manager, and locate the Shibboleth plugin in the Authentication category.

If you would like to selectively hide the Shibboleth login options for testing, enter something in the "Testing mode key" field. This term must appear in the query string for the controls of the plugin to render. For example, if you enter "incommon" you should test the login page by visiting "/login?reset=1&incommon" (reset=1 in case it remembers your logging in with a different method, in which case you'll only see the controls for that method anyway).

The links section is useful only for testing, but it can be used to destroy a link between your test

account and a particular institution so that you can try it again.

The Institutions section is where you manage which options are presented on the login page.

An example of an entry here, for the TestShib public identity provider test mechanism described above:

```
Entity ID: https://idp.testshib.org/idp/shibboleth
Label: TestShib
Host: testshib.org
```

The entity ID must strictly match what you have in your metadata provider, but the label is free-form and the host is optional. The login page attempts to do a reverse-DNS of the user's IP to see if they are on a particular network. If it turned out in this case that the client was from *.testshib.org this option would be pre-selected in the plugin's controls.

Save your settings with the button near the top right of the page when you're done.

Solr-powered Search

Introduction

Apache Solr is a search engine platform which is relatively mature and has a lot of powerful and flexible configurations. There has been extensive work to implement it into the HUBzero CMS and is currently a work-in-progress.

Solr is an open-source, mature, and stable searching service that is built upon the Apache Lucene search engine. The service provides features which lend itself to scaling and has a rich open source community. It is a Java-based service which provides search results through HTTP. Many companies such as Instagram, eBay, and StubHub rely on Solr to provide advanced searching capabilities.

The integration with Solr is currently under heavy development. It is **strongly recommended** to test on a QA / Stage host before using in a production environment.

Installation & First Time Configuration

Step 1: Install the hubzero-solr package

A system administrator must install the hubzero-solr RedHat or Debian Package using a package manager such as yum or aptitude. The package contains a version of Apache Solr and the configuration necessary for Solr to integrate with the CMS.

```
For RedHat / CentOS:
$    sudo yum install hubzero-solr
```

```
For Debian:
$    sudo apt-get install hubzero-solr
```

Once installed the service will need to be enabled.

```
$    sudo service hubzero-solr start
```

Step 2: Configure Search Service in the CMS

The HUBzero CMS needs to know to use Solr Search instead of Basic Search. To do this, a Hub administrator will need to log into the Administrative Backend and Configure the Search Component.

You will need to set **Engine** to *Apache Solr.* Then click the "Solr tab".

The Solr tab's default settings will work for the open-source distribution.

HUBzero-hosted hubs are configured with different ports! The following scheme is used:

```
Development (dev.hub.org): 2090
Stage (stage.hub.org): 2091
Scan / QA (qa.hub.org): 2092
Production (hub.org): 2093
```

Click "Save and Close" to save the settings. If the hubzero-solr service is started and the correct settings were set in the steps above, the status screen should indicate that the search engine is responding.

If there were any issues with configuration, the following screen will appear.

This would be a point where a support ticket is filed for the system administrator to confirm that the service is running. Please include all configuration parameters contained in Step #3 when

filing the ticket.

Step 3: Enable the Search Background Worker

In order to keep the search index fresh, a background worker is implemented to process data from the CMS and push it into the Solr service.

Currently the background worker is implemented as a Cron task that is called once a minute. There is work being done to develop a daemon which listens to CMS events and processes data without relying on Cron.

To setup the Cron-based worker a Hub administrator must go into the Administrative Backend, go to Components, Cron, and add the Task as shown below:

Click "Save and Close".

Step 4: Build the Initial Index

This implementation of Solr has hooks into the CMS which updates the index when a new record is added or marked for deletion. It will be necessary to add items which have been added before Solr was activated.

This operation should only need to be completed once. You will be unable to start this operation until it finishes for the first time.

The "Full Index" button populates a Queue which is periodically serviced by a worker. The worker will process the records and format for consumption by the Solr service. **This may take several hours to fully complete if the Hub has a lot of content.**

If an error with the worker occurs, a warning message such as this will appear.

Upgrading

Introduction

Upgrade processes are developed as needed and will be documented in this chapter as they are created. If you need an upgrade process developed you may contact support@hubzero.org to inquire about contracting for the necessary work.

You can select the upgrade section you want to view from the side menu.

NOTE: It will NOT be possible to automatically upgrade to the 2.1.0 release from the 1.0.x or earlier releases. Upgrades from earlier releases is a detailed manual process. You may contact support@hubzero.org to inquire about contracting for the necessary support if you need to perform such an upgrade.

Upgrading from 2.0.0

Performing the upgrade

This is a manual process due to the wide range of possible site configurations. So you will need to adapt to the specific needs of your HUBzero installation as necessary. The steps outlined below have been tested with the HUBzero 2.0.0 VMWare virtual machine image.

Some instructions will differ depending on how your site was installed. We have highlighted items like passwords, template names, and hub names which you may need to substitute with values specific to your hub.

All actions listed here must be performed as the root user.

You changed your root password right? If hubzero2015 is still your root password you really must change it as soon as possible.

Using a terminal, log into the host.

```
example.com login: root
Password: hubzero2015
```

The first thing that needs to be done is update the available package list, otherwise package installations will fail with obsolete package references.

```
apt-get update
```

In the case of the VMware virtual machine image the Network Time Protocol daemon isn't running so the time and date on the site could be wrong. While not critical, updating this can eliminate a number of warning errors that might otherwise be confusing. Let's install the ntp package.

```
apt-get install -y ntp
```

Then manually have it update the time and wait 10 seconds to give it enough time to take effect.

```
ntpd -gq
```

```
sleep 10
```

Now we will back up the HUBzero databases and the HUBzero CMS installation. We do not back up the "site" directory here, you may do so as you see fit. Be careful, some of these lines are wrapping around but are supposed to be on a single line. Remember to replace "mytemplate" with the name of any custom template you may be using. Custom and modified templates now are located in /var/www/example/app/templates. Preexisting Stock templates exist in /var/www/example/app/templates.

```
cd /root
```

```
mysqldump --defaults-file=/etc/mysql/debian.cnf --default-character-se
t=utf8 --skip-extended-insert --compact --order-by-primary --result-
file=example.`date +"%Y%m%d"`.sql example
```

```
mysqldump --defaults-file=/etc/mysql/debian.cnf --default-character-se
t=utf8 --skip-extended-insert --compact --order-by-primary --result-
file=example-metrics.`date +"%Y%m%d"`.sql example_metrics
```

```
tar -czf template-mytemplate
.`date +"%Y%m%d"`.tar.gz -C /var/www/example/app/templates mytemplate
```

```
tar -czf cms.`date +"
%Y%m%d"`.tar.gz -C /var/www/example --exclude=app/site/* .
```

Next we will make sure your HUBzero 2.0.0 and Debian 7 installation is completely up to date to ensure the upgrade to HUBzero 2.1.0 will go smoothly. If you are running Debian 6 you will need to upgrade to Debian 7 first (which is outside the scope of this document). The next step will rewrite your package repository configuration, if you have customized yours you can just

update the lines related to packages.hubzero.org to change the HUBzero package repository "ellie-deb7". Copy and paste the all of the following as one command:

```
cat << HERE >  /etc/apt/sources.list
deb http://ftp.us.debian.org/debian/ wheezy main contrib non-free
deb-src http://ftp.us.debian.org/debian/ wheezy main contrib non-free

deb http://ftp.us.debian.org/debian/ wheezy-updates main contrib non-
free
deb-src http://ftp.us.debian.org/debian/ wheezy-
updates main contrib non-free

deb http://security.debian.org/ wheezy/updates main contrib non-free
deb-src http://security.debian.org/ wheezy/updates main contrib non-
free

deb http://download.openvz.org/debian wheezy main

deb http://packages.hubzero.org/deb ellie-deb7 main
deb-src http://packages.hubzero.org/deb ellie-deb7 main
HERE
```

Now we can do the actual package upgrade.

```
apt-get update
```

```
apt-get dist-upgrade -y
```

You may be prompted with a message during the update of some packages. Enter "q" to quit the message.

Do some cleanup.

```
apt-get clean
```

```
apt-get autoremove -y
```

Update the existing HUBzero 2.0.0 installation with the most recent version (updated during previous step).

```
hzcms update
```

Install the HUBzero 2.1.0 package.

```
apt-get install -y hubzero-cms-2.1.0
```

Apply the HUBzero CMS 2.1.0 package to the current instance and let it reapply any configuration that may have been undone by the operating system upgrade. We also re-enable webdav and openvz so that their configuration files get updated (they were replaced during the operating system upgrade).

```
hzcms update
```

Remove old hubzero-cms packages that are no longer needed.

```
apt-get purge -y hubzero-cms-2.0.0
```

Clean up cached packages one more time then reboot to make sure everything is working correctly. This reboot should be relatively quick.

```
apt-get clean
```

```
reboot
```

Update the tool container image, if you don't have any Debian 7 tool container images you can skip this section.

```
chroot /var/lib/vz/template/debian-7.0-amd64-maxwell
```

```
cat << HERE >  /etc/apt/sources.list
deb http://http.us.debian.org/debian/ wheezy main contrib non-free
deb-src http://http.us.debian.org/debian/ wheezy main contrib non-free

deb http://security.debian.org/ wheezy/updates main contrib non-free
deb-src http://security.debian.org/ wheezy/updates main contrib non-
free

deb http://http.us.debian.org/debian/ wheezy-updates main contrib non-
free
deb http://packages.hubzero.org/deb ellie-deb7 main
deb-src http://packages.hubzero.org/deb ellie-deb7 main
HERE
```

```
apt-get update
```

Install the hubzero-policyrcd package which will prevent all future package installations and updates for this container image from stopping or restarting services (which would start/stop services running on the host which would not be a good thing.

```
apt-get install -y hubzero-policyrcd
```

Some packages require access to the /proc filesystem to get information about the system. Mount this special filesystem inside the container image chroot environment.

```
mount -t proc proc /proc
```

Upgrade packages.

```
apt-get upgrade -y
```

After the upgrade is complete unmount the temporary /proc mount and remove any packages that are no longer needed.

```
umount /proc
```

```
apt-get autoremove -y
```

Remove any cached packages from the image in order to conserve disk space and exit the chroot environment.

```
apt-get clean
```

```
exit
```

If you modified a core template in-place instead of creating your own see the last section to upgrade your modified core template.

If you were using your own template you will find it has been moved to /app/templates/yourtemplate and automatically modified it to conform with the new release. If your template is more complex it may require additional work to be made compatible with HUBzero 2.1.0. If you need to switch to the default template until you complete upgrading your template you may do so as follows:

```
Login to /administrator with the username 'admin' and the JOOMLA-
ADMIN password in /etc/hubzero.secrets.
```

```
From the main menu is /administrator go to
Extension Manager -> Template Manager ->  Click on the star for the Ki
```

```
mera (site) template in the default column.
```

That's it. Your hub should now be upgraded to HUBzero 2.1.0!

Upgrading a custom template

If you modified a default HUBzero template (hubbasic2012,hubbasic2013) without copying it to a new name you will need to restore it from the backups we made earlier as the upgrade process overwrites the standard HUBzero templates. Restoring this old template will likely break your site as your template will need to be upgraded to be compatible with this version of the HUBzero CMS. The web developer upgrade notes outline the basic changes that will be needed for your template, this should include renaming your template as it will now conflict with a core template name.

```
cd /var/www/example/app/templates
```

```
tar -xvpf /root/template-mytemplate.`date +"%Y%m%d"`.tar.gz
```

Upgrading from 1.3.1

Performing the upgrade

This is a manual process due to the wide range of possible site configurations. So you will need to adapt to the specific needs of your HUBzero installation as necessary. The steps outlined below have been tested with the HUBzero 1.3.1 VMWare virtual machine image we distributed at the HUBbub 2015 Conference.

Some instructions will differ depending on how your site was installed. We have highlighted items like passwords, template names, and hub names which you may need to substitute with values specific to your hub. The text given here is valid for the HUBzero VMware virtual machine image that was distributed at the Hubbub 2015 Conference.

All actions listed here must be performed as the root user.

You changed your root password right? If hubzero2015 is still your root password you really must change it as soon as possible.

Using a terminal, log into the host.

```
example.com login: root
Password: hubzero2015
```

The first thing that needs to be done is update the available package list, otherwise package installations will fail with obsolete package references.

```
apt-get update
```

In the case of the HUBzero 2015 VMware virtual machine image the Network Time Protocol daemon isn't running so the time and date on the site could be wrong. While not critical, updating this can eliminate a number of warning errors that might otherwise be confusing. Let's install the ntp package.

```
apt-get install -y ntp
```

Then manually have it update the time and wait 10 seconds to give it enough time to take effect.

```
ntpd -gq
```

```
sleep 10
```

Now we will back up the HUBzero databases and the HUBzero CMS installation. We do not back up the "site" directory here, you may do so as you see fit. Be careful, some of these lines are wrapping around but are supposed to be on a single line. Remember to replace "mytemplate" with the name of any custom template you may be using.

```
cd /root
```

```
mysqldump --defaults-file=/etc/mysql/debian.cnf --default-character-se
t=utf8 --skip-extended-insert --compact --order-by-primary --result-
file=example.`date +"%Y%m%d"`.sql example
```

```
mysqldump --defaults-file=/etc/mysql/debian.cnf --default-character-se
t=utf8 --skip-extended-insert --compact --order-by-primary --result-
file=example-metrics.`date +"%Y%m%d"`.sql example_metrics
```

```
tar -czf template-mytemplate
.`date +"%Y%m%d"`.tar.gz -C /var/www/example/templates mytemplate
```

```
tar -czf cms.`date +"
%Y%m%d"`.tar.gz -C /var/www/example --exclude=site/* .
```

Next we will make sure your HUBzero 1.3.1 and Debian 7 installation is completely up to date to ensure the upgrade to HUBzero 2.0.0 will go smoothly. If you are running Debian 6 you will need to upgrade to Debian 7 first (which is outside the scope of this document). The next step will rewrite your package repository configuration, if you have customized yours you can just update the lines related to packages.hubzero.org to change the HUBzero package repository

"ellie-deb7". Copy and paste the all of the following as one command:

```
cat << HERE >  /etc/apt/sources.list
deb http://ftp.us.debian.org/debian/ wheezy main contrib non-free
deb-src http://ftp.us.debian.org/debian/ wheezy main contrib non-free

deb http://ftp.us.debian.org/debian/ wheezy-updates main contrib non-
free
deb-src http://ftp.us.debian.org/debian/ wheezy-
updates main contrib non-free

deb http://security.debian.org/ wheezy/updates main contrib non-free
deb-src http://security.debian.org/ wheezy/updates main contrib non-
free

deb http://download.openvz.org/debian wheezy main

deb http://packages.hubzero.org/deb ellie-deb7 main
deb-src http://packages.hubzero.org/deb ellie-deb7 main
HERE
```

Now we can do the actual package upgrade. We perform a "dist-upgrade" instead of a simple "upgrade" because we are changing the HUBzero package repository and this causes a change in package dependencies that can only be resolved during a dull "dist-upgrade." We found during testing that stopping the mysql server manually was necessary.

```
apt-get update
```

```
apt-get dist-upgrade -y
```

While running it will ask you a few interactive questions which you may want to answer as follows:

- page through any upgrade notices it displays and continue
- /etc/vz/vz.conf update (Y, install the package maintainer's version)
- /etc/submit/submit-server.conf (Y, install the package maintainer's version)

Do some cleanup.

```
apt-get clean
```

```
apt-get autoremove -y
```

Update the existing HUBzero 1.3.1 installation with the most recent version (updated during previous step).

```
hzcms update
```

Install the HUBzero 2.0.0 package.

```
apt-get install -y hubzero-cms-2.0.0
```

Apply the HUBzero CMS 2.0.0 package to the current instance and let it reapply any configuration that may have been undone by the operating system upgrade. We also re-enable webdav and openvz so that their configuration files get updated (they were replaced during the operating system upgrade).

```
hzcms update
```

```
hzcms configure openvz --enable
```

Reboot to apply the new kernel that was installed during the operating system upgrade.

```
reboot
```

Remove old hubzero-cms packages that are no longer needed.

```
apt-get purge -y hubzero-cms-1.2.0
```

```
apt-get purge -y hubzero-cms-1.3.1
```

Clean up cached packages one more time then reboot to make sure everything is working correctly. This reboot should be relatively quick.

```
apt-get clean
```

```
reboot
```

Update the tool container image to the HUBzero 2.0 (ellie) repository, if you don't have any Debian 7 tool container images you can skip this section.

```
chroot /var/lib/vz/template/debian-7.0-amd64-maxwell
```

```
cat << HERE >  /etc/apt/sources.list
deb http://ftp.us.debian.org/debian/ wheezy main contrib non-free
deb-src http://ftp.us.debian.org/debian/ wheezy main contrib non-free

deb http://ftp.us.debian.org/debian/ wheezy-updates main contrib non-
free
deb-src http://ftp.us.debian.org/debian/ wheezy-
updates main contrib non-free

deb http://security.debian.org/ wheezy/updates main contrib non-free
deb-src http://security.debian.org/ wheezy/updates main contrib non-
free

deb http://packages.hubzero.org/deb ellie-deb7 main
deb-src http://packages.hubzero.org/deb ellie-deb7 main
```

HERE

```
apt-get update
```

Install the hubzero-policyrcd package which will prevent all future package installations and updates for this container image from stopping or restarting services (which would start/stop services running on the host which would not be a good thing.

```
apt-get install -y hubzero-policyrcd
```

Some packages require access to the /proc filesystem to get information about the system. Mount this special filesystem inside the container image chroot environment.

```
mount -t proc proc /proc
```

Switch to HUBzero 2.0 repository and upgrade packages.

```
apt-get dist-upgrade -y
```

After the upgrade is complete unmount the temporary /proc mount and remove any packages that are no longer needed.

```
umount /proc
```

```
apt-get autoremove -y
```

Remove any cached packages from the image in order to conserve disk space and exit the chroot environment.

```
apt-get clean
```

```
exit
```

Switch to the new HUBzero 2.0.0 template (Kimera). See the last section to upgrade your custom template, if you have one.

```
Login to /administrator with the username 'admin' and the JOOMLA-
ADMIN password in /etc/hubzero.secrets.
```

```
From the main menu is /administrator go to
Extension Manager -> Template Manager ->  Click on the star for the Ki
mera (site) template in the default column.
```

That's it. Your hub should now be upgraded to HUBzero 2.0.0!

Upgrading a custom template

If you modified a default HUBzero template (hubbasic2012,hubbasic2013) without copying it to a new name you will need to restore it from the backups we made earlier as the upgrade process overwrites the standard HUBzero templates. Restoring this old template will likely break your site as your template will need to be upgraded to be compatible with this version of the HUBzero CMS. The web developer upgrade notes outline the basic changes that will be needed for your template, this should include renaming your template as it will now conflict with a core template name.

```
cd /var/www/example/app/templates
```

```
tar -xvpf /root/template-mytemplate.`date +"%Y%m%d"`.tar.gz
```

Upgrading from 1.1.x

Performing the upgrade

It is possible to upgrade from HUBzero 1.1.x to 2.1.x. It is a manual process due to the wide range of possible site configurations there could be. So you will need to adapt to the specific needs of your HUBzero installation. The steps outlined below have been tested on the HUBzero 1.1.x VMWare virtual machine images that were distributed at the HUBbub 2012 conference and made available on the HUBzero website. This environment had particular issues with limited disk space that complicates the process somewhat.

Some instructions will differ depending on how your site was installed. We have highlighted items like passwords, template names, and hub names which you may need to substitute with values specific to your hub. The text given here is valid for the HUBzero 1.1.x VMware virtual machine images that were distributed at the Hubbub 2012 conference and made available on the HUBzero website.

Be careful if you use cut and paste. Some commands will consume all standard input which will cause subsequent commands pasted to not get executed. The steps below are intentionally broken down into chunks that avoid this problem.

You will have to perform the upgrade as the root user.

You changed your root password right? If hubzero2012 is still your root password you really must change it as soon as possible.

You changed your MySQL root password right? If hubzero2012 is still your MySQL root password you really must change it as soon as possible.

Template incompatability

Be aware that your existing template (unless using an unmodified HUBzero core template) will not work on your upgraded hub. You may want to develop a HUBzero CMS 2.1 compatible template on a fresh installation before upgrading an existing hub.

Backup HUBzero 1.1.x

Before we start we will back up the HUBzero databases and the HUBzero CMS installation. We do not back up the "site" directory (/var/www/example/site) here, you may do so however you see fit. Be careful, some of these lines are wrapping around but are supposed to be on a single line.

```
cd /root
mysqldump --defaults-file=/etc/mysql/debian.cnf --default-character-se
t=utf8 --skip-extended-insert --compact --order-by-primary --result-
```

```
file=example.`date +"%Y%m%d"`.sql example
mysqldump --defaults-file=/etc/mysql/debian.cnf --default-character-se
t=utf8 --skip-extended-insert --compact --order-by-primary --result-
file=example-metrics.`date +"%Y%m%d"`.sql example_metrics
tar -czf tem
plate-
hubbasic2012.`date +"%Y%m%d"`.
tar.gz -C /var/www/example/templates hubbasic2012
tar -czf cms.`date +"%Y%m%d"`.
tar.gz -C /var/www/example --exclude=site/* .
```

Update Debian 6

We are assuming you are running Debian 6.x right now. If you aren't then you may need to modify this section. The goal here is to get your current operating system version all the way up to date. Since Debian 6 is no longer even on long term support we have to configure the machine to use the the archived Debian 6 repositories. Then we perform a full package update.

```
cat << HERE > /etc/apt/sources.list
deb http://archive.debian.org/debian squeeze main contrib non-free
deb http://archive.debian.org/debian squeeze-lts main contrib non-free
deb http://packages.hubzero.org/deb manny main
HERE
cat << HERE >> /etc/apt/apt.conf

Acquire::Check-Valid-Until false;

HERE
apt-key adv --keyserver pgp.mit.edu --recv-keys 143C99EF
apt-get update
apt-get dist-upgrade -y
```

While running it will ask you a few interactive questions which you will want to answer as follows:

- read and dismiss ('q') apt-listchanges output
- let it restart services when asked

In the case of the HUBzero 1.1.x VMware virtual machine image the Network Time Protocol daemon isn't installed so the time and date on the site could be wrong. While not critical it can

eliminate a number of warning errors that might otherwise be confusing.

```
apt-get install -y ntp
```

We initiate a manual time update and add a 10 second delay at the end to give it enough time to take effect.

```
ntpd -gq
sleep 10
```

This is a good spot to clean up a little in case your system has limited disk space for later package updates. Autoremove any packages no longer needed.

```
apt-get autoremove -y
```

Then remove all the cached installation packages from the system

```
apt-get clean
```

Re-run the hubzero openvz configuration in case there was an OpenVZ kernel update

```
hzcms configure openvz --enable
```

Reboot the machine to ensure that any packages (especially kernel) are fully applied.

```
reboot
```

Update HUBzero 1.1.x

Now we update the existing HUBzero installation:

```
hzcms update
```

Upgrade to Debian 7

It is now time to update Debian to 7.x. We will do this in a couple steps in order to reduce disk usage which is necessary when trying to apply this procedure to the VMware virtual machine image that was distributed for HUBzero.

The first thing we do is change the package repositories to Debian wheezy (7.x). Then fetch the package keys which are used to verify the integrity of the packages.

```
cat << HERE >  /etc/apt/sources.list
deb http://ftp.us.debian.org/debian/ wheezy main contrib non-free
deb-src http://ftp.us.debian.org/debian/ wheezy main contrib non-free

deb http://ftp.us.debian.org/debian/ wheezy-updates main contrib non-
free
deb-src http://ftp.us.debian.org/debian/ wheezy-
updates main contrib non-free

deb http://security.debian.org/ wheezy/updates main contrib non-free
deb-src http://security.debian.org/ wheezy/updates main contrib non-
free

deb http://packages.hubzero.org/deb manny main
HERE
apt-get update
apt-get install  -y debian-archive-keyring
```

In order to free up disk space the upgrade we will remove the hubzero-texvc package temporarily

```
apt-get purge -y hubzero-texvc
```

Then autoremove all the dependencies that package had brought in (TeX)

```
apt-get autoremove -y
```

Then remove all the cached installation packages from the system

```
apt-get clean
```

Now we can do the actual Debian 6 to Debian 7 upgrade. We have to do an extra step to preinstall the mysql-server-5.5 package due to a problem with APT::Immediate-Configure failing to resolve upgrade issues before doing the distribution upgrade.

```
apt-get update
apt-get install -y mysql-server-5.5 libc-dev
```

While running it will ask you a few interactive questions which you will want to answer as follows:

- read and dismiss ('q') apt-listchanges output
- let it restart services when asked
- don't change the MySQLl root password when asked

Then perform the full distribution upgrade:

```
apt-get dist-upgrade -y
```

While running it will ask you a few interactive questions which you will want to answer as follows:

- read and dismiss ('q') apt-listchanges output
- /etc/vz/vz.conf update (Y, install the package maintainer's version)
- /etc/auto.master update (Y, install the package maintainer's version)
- /etc/default/grub (Y, install the package maintainer's version)
- /etc/logrotate.d/apache2 update (Y, install the package maintainer's version)

Remove packages no longer required.

```
apt-get autoremove -y
```

Remove cached packages again to keep disk usage low for systems where that is a concern.

```
apt-get clean
```

Re-install the hubzero-texvc package.

```
apt-get install -y texlive-latex-base ghostscript imagemagick texlive-latex-extra
```

We remove cached packages again to keep disk usage low for systems where that is a concern.

```
apt-get clean
```

Now we can remove a leftover configuration file for the suhosin php package that was deprecated in Debian 7.

```
rm /etc/php5/conf.d/suhosin.ini
```

Now we reapply the current HUBzero updates to the current instance and let it reapply any configuration that may have been undone by the operating system upgrade. We also re-enable webdav and openvz so that their configuration files get updated (they were replaced during the operating system upgrade).

```
hzcms update
hzcms configure webdav --enable
hzcms configure openvz --enable
```

Upgrade OpenVZ Kernel

Debian 7 does not have an OpenVZ kernel package available from Debian so we need to install one that has been created by the OpenVZ kernel developers.

First add the OpenVZ package repository to your apt repository configuration

```
cat << HERE >> /etc/apt/sources.list

deb http://download.openvz.org/debian wheezy main
HERE
wget http://ftp.openvz.org/debian/archive.key -q -O - | apt-key add -
```

Then install the OpenVZ kernel package

```
apt-get update
apt-get install -y vzkernel
```

Configure system to boot with new kernel by default

```
hzcms configure openvz --enable
```

Reboot to apply the new kernel. This step can take many minutes for some reason as it sits at one spot for some time before finally shutting down. More investigation here might lead to another step that could be taken beforehand to reduce the reboot time.

```
reboot
```

Now we remove the old Debian 6 OpenVZ kernel

```
apt-get purge -y linux-image-2.6.32-5-openvz-amd64 linux-
image-2.6-amd64
```

Remove any packages that are no longer required

```
apt-get autoremove -y
```

Remove all the cached installation packages from the system

```
apt-get clean
```

Upgrade to HUBzero 2.1

Change the HUBzero package repository to ellie-deb7 in /etc/apt/sources.list

```
sed -i -e 's/manny/ellie-deb7/g' /etc/apt/sources.list
```

Then run a full distribution upgrade (a regular upgrade doesn't handle package renames and other versioning issues that have changed since HUBzero 1.1.x).

```
apt-get update
apt-get dist-upgrade -y
```

While running it will ask you a few interactive questions which you will want to answer as follows:

- /etc/vz/vz.conf update (Y, install the package maintainer's version)

The HUBzero 1.1.x installation failed to record the MySQL root password. It is necessary to add the MySQL root password to the /etc/hubzero.secrets file.

```
cat << HERE >> /etc/hubzero.secrets

MYSQL-ROOT=hubzero2012

HERE
```

Then autoremove any packages that are no longer needed after the upgrade

```
apt-get autoremove -y
```

Then remove all the cached installation packages from the system

```
apt-get clean
```

Apply the HUBzero upgrade

```
hzcms update
```

Updating the HUBzero CMS Template

If you were using a default HUBzero template (hubbasic or hubbasic2012) without copying it to a new name you will need to restore it from the backups we made earlier as the upgrade process overwrites all the standard HUBzero templates.

```
cd /var/www/example/app/templates
tar -xvpf /root/template-hubzero2012.`date +"%Y%m%d"`.tar.gz
```

You will then need to rename the template and upgrade it to be compatible with HUBzero CMS 2.1. After renaming it you can have the site discover the new template via the Extension discovery process. The web developer release notes outline the basic changes that will be needed for your template.

If you were using a custom template it SHOULD have been moved to /var/www/example/app/templates/YOUR_TEMPLATE_NAME and may have been partially upgraded for HUBzero CMS 2.1 but will almost certainly require extensive work to make it fully compatible.

Updating the HUBzero Tool Container Image

Finally it is necessary to update your Debian 6 tool container image to the last Debian 6 release and HUBzero 2.1.x support packages. If you don't have any Debian 6 tool container images you can skip this section.

```
chroot /var/lib/vz/template/debian-6.0-amd64-maxwell
cat << HERE >  /etc/apt/sources.list
deb http://archive.debian.org/debian squeeze main contrib non-free
deb http://archive.debian.org/debian squeeze-lts main contrib non-free
deb http://packages.hubzero.org/deb ellie-deb6 main
HERE
cat << HERE >> /etc/apt/apt.conf

Acquire::Check-Valid-Until false;

HERE
apt-key adv --keyserver pgp.mit.edu --recv-keys 143C99EF
apt-get update
```

We install the hubzero-policyrcd package which will prevent all future package installations and updates for this container image from stopping or restarting services (which would start/stop services running on the host which would not be a good thing.

```
apt-get install -y hubzero-policyrcd
```

Some packages require access to the /proc filesystem to get information about the system. So we mount this special filesystem inside the container image chroot environment for the duration of the update then unmount it

```
mount -t proc proc /proc
```

Do the package upgrade to switch to Debian 6 LTS.

```
apt-get dist-upgrade -y
```

After the upgrade is complete unmount the temporary /proc

```
umount /proc
```

Remove any packages no longer required

```
apt-get autoremove -y
```

Lastly we remove any cached packages from the image in order to conserve disk space.

```
apt-get clean
```

Exit the chroot environment and go back to the host computer environment.

```
exit
```

Conclusion

That's it. Your hub should now be upgraded to Debian 7.x and HUBzero 2.1.0

Your tool container images remain Debian 6 as it may be necessary to update your tools for Debian 7 and that process is outside the scope of this document.

Upgrading to Debian 8

Performing the upgrade

It is possible to upgrade from HUBzero 2.1.x running on Debian 7 "wheezy" to HUBzero 2.1.x running on Debian 8 "jessie". It is a manual process due to the wide range of possible site configurations there could be. So you will need to adapt to the specific needs of your HUBzero installation. The steps outlined below have been tested on the HUBzero 2.1.x VMWare virtual machine images that were made available on the HUBzero website. This environment had particular issues with limited disk space that complicates the process somewhat.

Some instructions will differ depending on how your site was installed. We have highlighted items like passwords, template names, and hub names which you may need to substitute with values specific to your hub. The text given here is valid for the HUBzero 2.1.x VMware virtual machine images that were made available on the HUBzero website.

Be careful if you use cut and paste. Some commands will consume all standard input which will cause subsequent commands pasted to not get executed. The steps below are intentionally broken down into chunks that avoid this problem.

You will have to perform the upgrade as the root user.

You changed your root password right? If hubzero2015 is still your root password you really must change it as soon as possible.

You changed your MySQL root password right? If hubzero2015 is still your MySQL root password you really must change it as soon as possible.

Update Debian7

We are assuming you are running Debian 7.x right now. If you aren't then you may need to modify this section. The goal here is to get your current operating system version all the way up to date.

```
apt-key adv --keyserver pgp.key-server.io --recv-keys 143C99EF
apt-get update
apt-get dist-upgrade -y
```

While running it will ask you a few interactive questions which you will want to answer as follows:

- read and dismiss ('q') apt-listchanges output
- let it restart services if asked

In the case of the HUBzero 2.1.x VMware virtual machine image the Network Time Protocol daemon isn't installed so the time and date on the site could be wrong. While not critical it can eliminate a number of warning errors that might otherwise be confusing.

```
apt-get install -y ntp
```

We initiate a manual time update and add a 10 second delay at the end to give it enough time to take effect.

```
ntpd -gq
sleep 10
```

This is a good spot to clean up a little in case your system has limited disk space for later package updates. Autoremove any packages no longer needed.

```
apt-get autoremove -y
```

Then remove all the cached installation packages from the system

```
apt-get clean
```

Re-run the hubzero openvz configuration in case there was an OpenVZ kernel update

```
hzcms configure openvz --enable
```

Reboot the machine to ensure that any packages (especially kernel) are fully applied.

```
reboot
```

Update HUBzero 2.1.x

Now we update the existing HUBzero installation:

```
hzcms update
```

Upgrade to Debian 8

It is now time to update Debian to 8.x. We will do this in a couple steps in order to reduce disk usage which is necessary when trying to apply this procedure to the VMware virtual machine image that was distributed for HUBzero.

In order to free up disk space the upgrade we will remove the hubzero-texvc and tex-common packages temporarily

```
apt-get purge -y hubzero-texvc tex-common
```

Then remove all the cached installation packages from the system

```
apt-get clean
```

Then change the package repositories to Debian jessie (8.x).

```
cat << HERE >  /etc/apt/sources.list
deb http://ftp.us.debian.org/debian/ jessie main contrib non-free
deb-src http://ftp.us.debian.org/debian/ jessie main contrib non-free

deb http://ftp.us.debian.org/debian/ jessie-updates main contrib non-
free
deb-src http://ftp.us.debian.org/debian/ jessie-
updates main contrib non-free

deb http://security.debian.org/ jessie/updates main contrib non-free
deb-src http://security.debian.org/ jessie/updates main contrib non-
free

deb http://packages.hubzero.org/deb ellie-deb8 main

deb http://download.openvz.org/debian wheezy main
deb http://download.openvz.org/debian jessie main
```

HERE

Update package database

```
apt-get update
```

Then perform the full distribution upgrade:

```
apt-get dist-upgrade -y
```

While running it will ask you a few interactive questions which you will want to answer as follows:

- read and dismiss ('q') apt-listchanges output
- Disable SSH password authentication for root? No
- let it restart services when asked
- /etc/sysctl.conf update (Y, install the package maintainer's version)
- /etc/logrotate.d/apache2 update (Y, install the package maintainer's version)
- /etc/php5/apache2/php.ini update (install the package maintainer's version)
- /etc/updatedb.conf update (Y, install the package maintainer's version)
- /etc/default/spamassassin update (Y, install the package maintainer's version)

Remove packages no longer required.

```
apt-get autoremove -y
```

Remove cached packages again to keep disk usage low for systems where that is a concern.

```
apt-get clean
```

Re-install some of the TeX dependency chain (done separately to conserve disk space)

```
apt-get install -y texlive-latex-base
```

We remove cached packages again to keep disk usage low for systems where that is a concern.

```
apt-get clean
```

Re-install hubzero-texvc

```
apt-get install -y hubzero-texvc
```

We remove cached packages again to keep disk usage low for systems where that is a concern.

```
apt-get clean
```

Rebuild man-page database in case we ran out of disk space during install

```
mandb
```

Now we reapply the current HUBzero updates to the current instance and let it reapply any configuration that may have been undone by the operating system upgrade. We also re-enable openvz so that their configuration files get updated (they may have been replaced during the operating system upgrade).

```
service nscd restart
hzcms update
hzcms configure openvz --enable
```

Reboot to ensure we are still properly configured with OpenVZ kernel.

```
reboot
```

Remove any packages that are no longer required

```
apt-get autoremove -y
```

Remove all the cached installation packages from the system

```
apt-get clean
```

Updating the HUBzero Tool Container Image

Finally it is necessary to update your tool container image.

**Check disk space before before updating the tool container image.

If using a Debian 6 tool container image edit the container image's repository configuration (do this step OR the next):

```
chroot /var/lib/vz/template/debian-6.0-amd64-maxwell

cat << HERE >  /etc/apt/sources.list
deb http://archive.debian.org/debian squeeze main contrib non-free
deb http://archive.debian.org/debian squeeze-lts main contrib non-free
deb http://packages.hubzero.org/deb ellie-deb6 main
HERE
cat << HERE >> /etc/apt/apt.conf

Acquire::Check-Valid-Until false;

HERE
apt-key adv --keyserver pgp.mit.edu --recv-keys 143C99EF
apt-get update
```

If using a Debian 7 tool container change into the image's context (do this step OR the previous):

```
cp /etc/resolv.conf /var/lib/vz/template/debian-7.0-amd64-maxwell/etc/
resolv.conf
```

```
chroot /var/lib/vz/template/debian-7.0-amd64-maxwell
```

```
wget http://packages.hubzero.org/deb/hubzero-signing-
key.asc -q -O - | apt-key add -
```

Update package database

```
apt-get update
```

Once inside the container image context we install the hubzero-policyrcd package which will prevent all future package installations and updates for this container image from stopping or restarting services (which would start/stop services running on the host which would not be a good thing.

```
apt-get install -y hubzero-policyrcd
```

Some packages require access to the /proc filesystem to get information about the system. So we mount this special filesystem inside the container image chroot environment for the duration of the update then unmount it

```
mount -t proc proc /proc
```

Do the package upgrade

```
apt-get dist-upgrade -y
```

After the upgrade is complete unmount the temporary /proc

```
umount /proc
```

Remove any packages no longer required

```
apt-get autoremove -y
```

Lastly we remove any cached packages from the image in order to conserve disk space.

```
apt-get clean
```

Exit the chroot environment and go back to the host computer environment.

```
exit
```

Conclusion

That's it. Your hub should now be upgraded to Debian 8.x and HUBzero 2.1.x

Your tool container images remain Debian 6 or 7 as it may be necessary to update your tools for Debian 8 and that process is outside the scope of this document.

Updates

Introduction

The 2.1.0 release of HUBzero has received regular updates through its release cycle.

Updates are typically applied simply by running:

```
# apt-get update
# apt-get upgrade
# hzcms update
```

Source Code

Configure Advanced Package Tool

If you are using Debian 7 "wheezy" add the following line to /etc/apt/sources.list

```
deb-src http://packages.hubzero.org/deb ellie-deb7 main
```

If you are using Debian 8 "jessie" add the following line to /etc/apt/sources.list

```
deb-src http://packages.hubzero.org/deb ellie-deb8 main
```

With the above configure update the local package database with information about the packages now available through these new repositories:

```
# apt-get update
```

How to Download a source package

Sources are normally **not** installed. You can only install them if you know the package name.

How to find the name of the source package

A source package could generate many .debs. To know the source package name, see the **Source:** field in the output of

```
apt-cache show package_name
```

Sometimes the **SOURCE:** field is not present, then you can check using:

```
apt-cache showsrc package_name
```

Downloading with apt-get source

One way to obtain source packages is with

```
apt-get source <package name>
```

A source package is downloaded in the current directory and is not installed (it will not appear in the installed package list), you need not be root to use `apt-get source`.