System Administrators

The Setup and Maintenance Guide details hardware and software requirements, how to bring up a new HUB, upgrade system software, etc.

Installation (Debian 6/7)

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 6 (squeeze) or 7 (wheezy).

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 6.0 (6.0.10 as of this writing) or Debian GNU/Linux 7.0 (7.6 as of this writing) should be installed on each host used by a HUBzero installation.

HUBzero has packaging support for amd64 (64bit) Intel architectures. i386 (32bit) packaging is possible but was not produced for this release due to lack of demand.

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

Now configure the location of the HUBzero package repository by adding the following line to /etc/apt/sources.list

For Debian 6:

```
deb http://packages.hubzero.org/deb diego-deb6 main
```

For Debian 7:

```
deb http://packages.hubzero.org/deb diego-deb7 main
deb http://download.openvz.org/debian wheezy main
```

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
```

For Debian 7 you will aso need the OpenVZ archive key to be able to verify packages from the OpenVZ archive:

```
wget http://ftp.openvz.org/debian/archive.key -q -O - | apt-key add -
```

With the above configure 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
```

NOTE: After being prompted for a subject and pressing return, the Mail app will read the body of your email. Just type a test message and press return after each line. Once you are done with your email's body, enter a single '.' on it's own line and press return. The mail program will stop reading your input and will print a bunch of low level SMTP connection info to the screen as it delievers your message. At the bottom you should see a "Completed" line. You might have to press return after the "Completed" message to return to a command prompt.

CMS

Install

```
# apt-get install -y hubzero-cms-1.3.1
```

Configure

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

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
```

OpenVZ

Install

HUBzero makes extensive use of OpenVZ containers so it is recommended to use the OpenVZ enabled kernel on all HUBzero servers.

```
# 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.

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).

Maxwell Service

Install

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

Configure

```
# mkvztemplate amd64 wheezy diego
```

or

```
# mkvztemplate amd64 squeeze diego
```

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
#
```

vncproxy

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.

```
Determine which filesystem contains your home directories and add "quo
ta" to the mount option of the corresponding entry in the /etc/fstab f
ile.
```

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 /  (may fail if there is only one filesystem mount o
n the host, reboot is required before quotas can be used)
# /etc/init.d/quota restart  (will fail is "mount -oremount" fails)
# 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.

Login to the site with the following credentials:

username: admin

password: located in "/etc/hubzero.secrets" as the "JOOMLA-ADMIN"

> You can access this file as the root user with the command cat /etc/hubzero.secrets .

Metrics

Install

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

Configure

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

Rappture

Install

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

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 with only contains the dependencies needed to pull in these other packages.

Note depending on which template you made, the chroot might be to "debian-**7**.0-amd64-maxwell" or "debian-**6**.0-amd64-maxwell"

```
# 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 is 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.

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.

```
[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 an 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.

```
[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.

```
[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

```
[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.

The rsyslog service is used to collect messages written to distributor.log. Using this service avoids the necessity of making distributor.log world writable. To use rsyslog a couple of rules must be addded to /etc/rsyslog.conf. The required rules are

```
################
#### RULES ####
################
local6.* /var/log/submit/distributor/distributor.log
local6.* ~
```

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  transferredTimestampFile
```

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.

OpenLDAP

OpenLDAP Installation

The base OpenLDAP package is obtained from debian.org and installed using the command

```
# apt-get install slapd
```

This adds the following files to the file system:

```
/etc/ldap/sasl2/
/etc/ldap/schema/nis.ldif
/etc/ldap/schema/cosine.ldif
/etc/ldap/schema/duaconf.schema
/etc/ldap/schema/core.ldif
/etc/ldap/schema/nadf.schema
/etc/ldap/schema/openldap.ldif
/etc/ldap/schema/inetorgperson.ldif
/etc/ldap/schema/cosine.schema
/etc/ldap/schema/nis.schema
/etc/ldap/schema/core.schema
/etc/ldap/schema/openldap.schema
/etc/ldap/schema/collective.schema
/etc/ldap/schema/dyngroup.schema
/etc/ldap/schema/ppolicy.schema
/etc/ldap/schema/java.schema
/etc/ldap/schema/misc.schema
/etc/ldap/schema/README
/etc/ldap/schema/inetorgperson.schema
/etc/ldap/schema/corba.schema
/etc/init.d/slapd
/etc/default/slapd
/usr/lib/libslapi-2.4.so.2.1.0
/usr/lib/ldap/auditlog-2.4.so.2.1.0
/usr/lib/ldap/back_bdb.la
/usr/lib/ldap/refint.la
/usr/lib/ldap/auditlog.la
/usr/lib/ldap/memberof.la
/usr/lib/ldap/back_null.la
/usr/lib/ldap/dynlist.la
/usr/lib/ldap/valsort-2.4.so.2.1.0
/usr/lib/ldap/dynlist-2.4.so.2.1.0
```

```
/usr/lib/ldap/back_passwd-2.4.so.2.1.0
/usr/lib/ldap/back_relay-2.4.so.2.1.0
/usr/lib/ldap/seqmod.la
/usr/lib/ldap/back_monitor.la
/usr/lib/ldap/pcache.la
/usr/lib/ldap/back_sock-2.4.so.2.1.0
/usr/lib/ldap/back_perl.la
/usr/lib/ldap/syncprov-2.4.so.2.1.0
/usr/lib/ldap/memberof-2.4.so.2.1.0
/usr/lib/ldap/back_perl-2.4.so.2.1.0
/usr/lib/ldap/rwm-2.4.so.2.1.0
/usr/lib/ldap/ppolicy-2.4.so.2.1.0
/usr/lib/ldap/accesslog.la
/usr/lib/ldap/accesslog-2.4.so.2.1.0
/usr/lib/ldap/back_dnssrv.la
/usr/lib/ldap/back_dnssrv-2.4.so.2.1.0
/usr/lib/ldap/dyngroup-2.4.so.2.1.0
/usr/lib/ldap/syncprov.la
/usr/lib/ldap/back_passwd.la
/usr/lib/ldap/back_relay.la
/usr/lib/ldap/back_monitor-2.4.so.2.1.0
/usr/lib/ldap/retcode.la
/usr/lib/ldap/back_sock.la
/usr/lib/ldap/pcache-2.4.so.2.1.0
/usr/lib/ldap/constraint-2.4.so.2.1.0
/usr/lib/ldap/back_ldap.la
/usr/lib/ldap/back_sql-2.4.so.2.1.0
/usr/lib/ldap/back_hdb.la
/usr/lib/ldap/dds.la
/usr/lib/ldap/back_ldap-2.4.so.2.1.0
/usr/lib/ldap/back_hdb-2.4.so.2.1.0
/usr/lib/ldap/unique.la
/usr/lib/ldap/back_sql.la
/usr/lib/ldap/back_meta.la
/usr/lib/ldap/ppolicy.la
/usr/lib/ldap/back_bdb-2.4.so.2.1.0
/usr/lib/ldap/seqmod-2.4.so.2.1.0
/usr/lib/ldap/translucent.la
/usr/lib/ldap/back_shell.la
/usr/lib/ldap/back_meta-2.4.so.2.1.0
/usr/lib/ldap/rwm.la
/usr/lib/ldap/translucent-2.4.so.2.1.0
/usr/lib/ldap/unique-2.4.so.2.1.0
/usr/lib/ldap/refint-2.4.so.2.1.0
/usr/lib/ldap/dyngroup.la
/usr/lib/ldap/valsort.la
```

```
/usr/lib/ldap/back_shell-2.4.so.2.1.0
/usr/lib/ldap/retcode-2.4.so.2.1.0
/usr/lib/ldap/back_null-2.4.so.2.1.0
/usr/lib/ldap/dds-2.4.so.2.1.0
/usr/lib/ldap/constraint.la
/usr/sbin/slapcat
/usr/sbin/slappasswd
/usr/sbin/slapdn
/usr/sbin/slapauth
/usr/sbin/slapd
/usr/sbin/slapadd
/usr/sbin/slapindex
/usr/sbin/slapacl
/usr/sbin/slaptest
/usr/share/lintian/
/usr/share/lintian/overrides/
/usr/share/lintian/overrides/slapd
/usr/share/man/man8/slapauth.8.gz
/usr/share/man/man8/slapcat.8.gz
/usr/share/man/man8/slapacl.8.gz
/usr/share/man/man8/slapadd.8.gz
/usr/share/man/man8/slaptest.8.gz
/usr/share/man/man8/slapindex.8.gz
/usr/share/man/man8/slappasswd.8.gz
/usr/share/man/man8/slapd.8.gz
/usr/share/man/man8/slapdn.8.gz
/usr/share/man/man5/slapd-monitor.5.gz
/usr/share/man/man5/slapd-sock.5.gz
/usr/share/man/man5/slapo-accesslog.5.gz
/usr/share/man/man5/slapd-dnssrv.5.gz
/usr/share/man/man5/slapo-dds.5.gz
/usr/share/man/man5/slapo-constraint.5.gz
/usr/share/man/man5/slapd-ldbm.5.gz
/usr/share/man/man5/slapd-config.5.gz
/usr/share/man/man5/slapo-dyngroup.5.gz
/usr/share/man/man5/slapo-memberof.5.gz
/usr/share/man/man5/slapo-auditlog.5.gz
/usr/share/man/man5/slapo-translucent.5.gz
/usr/share/man/man5/slapd-ldif.5.gz
/usr/share/man/man5/slapd.backends.5.gz
/usr/share/man/man5/slapd.access.5.gz
/usr/share/man/man5/slapd-meta.5.gz
/usr/share/man/man5/slapd.conf.5.gz
/usr/share/man/man5/slapd-shell.5.gz
/usr/share/man/man5/slapd-sql.5.gz
/usr/share/man/man5/slapd-passwd.5.gz
```

```
/usr/share/man/man5/slapd-bdb.5.gz
/usr/share/man/man5/slapd-hdb.5.gz
/usr/share/man/man5/slapo-pcache.5.gz
/usr/share/man/man5/slapo-valsort.5.gz
/usr/share/man/man5/slapo-chain.5.gz
/usr/share/man/man5/slapo-unique.5.gz
/usr/share/man/man5/slapo-retcode.5.gz
/usr/share/man/man5/slapd-perl.5.gz
/usr/share/man/man5/slapd-null.5.gz
/usr/share/man/man5/slapo-ppolicy.5.gz
/usr/share/man/man5/slapd.plugin.5.gz
/usr/share/man/man5/slapd-ldap.5.gz
/usr/share/man/man5/slapo-rwm.5.gz
/usr/share/man/man5/slapo-syncprov.5.gz
/usr/share/man/man5/slapd.overlays.5.gz
/usr/share/man/man5/slapo-dynlist.5.gz
/usr/share/man/man5/slapd-relay.5.gz
/usr/share/man/man5/slapo-refint.5.gz
/usr/share/doc/slapd/copyright
/usr/share/doc/slapd/changelog.Debian.gz
/usr/share/doc/slapd/NEWS.Debian.gz
/usr/share/doc/slapd/README.Debian.gz
/usr/share/doc/slapd/examples/
/usr/share/doc/slapd/examples/slapd.backup
/usr/share/doc/slapd/TODO.Debian
/usr/share/doc/slapd/README.DB_CONFIG.gz
/usr/share/slapd/ldiftopasswd
/usr/share/slapd/slapd.conf
/usr/share/slapd/DB_CONFIG
/var/lib/slapd/
/usr/lib/ldap/back_meta.so
/usr/lib/ldap/dynlist.so
/usr/lib/ldap/memberof.so
/usr/lib/ldap/accesslog-2.4.so.2
/usr/lib/ldap/seqmod.so
/usr/lib/ldap/seqmod-2.4.so.2
/usr/lib/ldap/ppolicy.so
/usr/lib/ldap/back_dnssrv.so
/usr/lib/ldap/back_bdb.so
/usr/lib/ldap/back_ldap.so
/usr/lib/ldap/back_ldap-2.4.so.2
/usr/lib/ldap/back_sql.so
/usr/lib/ldap/back_null.so
/usr/lib/ldap/refint.so
/usr/lib/ldap/refint-2.4.so.2
/usr/lib/ldap/back_relay-2.4.so.2
```

```
/usr/lib/ldap/back_hdb-2.4.so.2
/usr/lib/ldap/back_meta-2.4.so.2
/usr/lib/ldap/back_shell.so
/usr/lib/ldap/ppolicy-2.4.so.2
/usr/lib/ldap/back_sock-2.4.so.2
/usr/lib/ldap/auditlog-2.4.so.2
/usr/lib/ldap/dds-2.4.so.2
/usr/lib/ldap/dyngroup-2.4.so.2
/usr/lib/ldap/back_sql-2.4.so.2
/usr/lib/ldap/constraint-2.4.so.2
/usr/lib/ldap/rwm.so
/usr/lib/ldap/back_dnssrv-2.4.so.2
/usr/lib/ldap/auditlog.so
/usr/lib/ldap/syncprov.so
/usr/lib/ldap/syncprov-2.4.so.2
/usr/lib/ldap/rwm-2.4.so.2
/usr/lib/ldap/memberof-2.4.so.2
/usr/lib/ldap/valsort.so
/usr/lib/ldap/back_perl-2.4.so.2
/usr/lib/ldap/dyngroup.so
/usr/lib/ldap/back_passwd-2.4.so.2
/usr/lib/ldap/back_monitor.so
/usr/lib/ldap/valsort-2.4.so.2
/usr/lib/ldap/back_relay.so
/usr/lib/ldap/back_perl.so
/usr/lib/ldap/unique-2.4.so.2
/usr/lib/ldap/translucent.so
/usr/lib/ldap/back_null-2.4.so.2
/usr/lib/ldap/accesslog.so
/usr/lib/ldap/back_passwd.so
/usr/lib/ldap/back_shell-2.4.so.2
/usr/lib/ldap/constraint.so
/usr/lib/ldap/back_sock.so
/usr/lib/ldap/pcache.so
/usr/lib/ldap/pcache-2.4.so.2
/usr/lib/ldap/translucent-2.4.so.2
/usr/lib/ldap/dynlist-2.4.so.2
/usr/lib/ldap/back_monitor-2.4.so.2
/usr/lib/ldap/unique.so
/usr/lib/ldap/dds.so
/usr/lib/ldap/back_hdb.so
/usr/lib/ldap/retcode-2.4.so.2
/usr/lib/ldap/back_bdb-2.4.so.2
/usr/lib/ldap/retcode.so
/usr/lib/libslapi-2.4.so.2
/usr/share/doc/slapd/examples/slapd.conf
```

```
/usr/share/doc/slapd/examples/DB_CONFIG
```

OpenLDAP Configuration

The **/etc/ldap/slapd.conf** is edited and the base DN changed to myhub.org (we are assuming that myhub.org was the DN specified initially when asked during the initial six questions step).

Next, the LDAP admin password is set to that specified in the six questions.

nscd Installation

The Name Service Cache Daemon nscd is downloaded from debian.org and installed using the following command:

```
# apt-get install nscd
```

This adds the following files to the file system:

```
/etc/init.d/nscd
/etc/nscd.conf
/usr/share/doc/nscd/
/usr/share/doc/nscd/changelog.Debian.gz
/usr/share/doc/nscd/NEWS.Debian.gz
/usr/share/doc/nscd/copyright
/usr/share/man/man8/nscd.8.gz
/usr/share/man/man5/nscd.conf.5.gz
/usr/sbin/nscd
/var/run/nscd/
/var/cache/nscd/
```

HUBzero LDAP Schema Installation

The HUBzero OpenLDAP package is downloaded from packages.hubzero.org and installed using the following command:

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

This adds the following files to the file system:

```
/etc/ldap/schema/hub.schema
/usr/share/doc/hubzero-openldap/changelog.Debian.gz
/usr/share/doc/hubzero-openldap/copyright
/usr/share/hubzero-openldap/do-hz-openldap-install.tmpl
/usr/share/hubzero-openldap/HUB-INIT-SLAPD.tmpl
```

Next, the **/etc/ldap/slapd.conf** is edited and the following line added as the last "include" statement.

```
include         /etc/ldap/schema/hub.schema
```

Then OpenLDAP is restarted with

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

PAM Configuration to Use LDAP

The PAM module for LDAP is downloaded from debian.org and installed using the following command:

```
# apt-get install libpam-ldap
```

This adds the following files to the file system:

```
/usr/share/doc/libpam-ldap/LDAP-Permissions.txt
/usr/share/doc/libpam-ldap/README.Debian
/usr/share/doc/libpam-ldap/changelog.gz
/usr/share/doc/libpam-ldap/copyright
/usr/share/doc/libpam-ldap/ldapns.schema
/usr/share/doc/libpam-ldap/AUTHORS
/usr/share/doc/libpam-ldap/examples/
/usr/share/doc/libpam-ldap/examples/pam.d/
/usr/share/doc/libpam-ldap/examples/pam.d/halt
/usr/share/doc/libpam-ldap/examples/pam.d/linuxconf-pair
/usr/share/doc/libpam-ldap/examples/pam.d/rsh
/usr/share/doc/libpam-ldap/examples/pam.d/gdm
/usr/share/doc/libpam-ldap/examples/pam.d/samba
```

```
/usr/share/doc/libpam-ldap/examples/pam.d/ftp
/usr/share/doc/libpam-ldap/examples/pam.d/other
/usr/share/doc/libpam-ldap/examples/pam.d/mcserv
/usr/share/doc/libpam-ldap/examples/pam.d/reboot
/usr/share/doc/libpam-ldap/examples/pam.d/pop
/usr/share/doc/libpam-ldap/examples/pam.d/ppp
/usr/share/doc/libpam-ldap/examples/pam.d/vlock
/usr/share/doc/libpam-ldap/examples/pam.d/poweroff
/usr/share/doc/libpam-ldap/examples/pam.d/xdm
/usr/share/doc/libpam-ldap/examples/pam.d/xlock
/usr/share/doc/libpam-ldap/examples/pam.d/imap
/usr/share/doc/libpam-ldap/examples/pam.d/linuxconf
/usr/share/doc/libpam-ldap/examples/pam.d/xscreensaver
/usr/share/doc/libpam-ldap/examples/pam.d/passwd
/usr/share/doc/libpam-ldap/examples/pam.d/chsh
/usr/share/doc/libpam-ldap/examples/pam.d/login
/usr/share/doc/libpam-ldap/examples/pam.d/su
/usr/share/doc/libpam-ldap/examples/pam.d/ssh
/usr/share/doc/libpam-ldap/examples/pam.d/chfn
/usr/share/doc/libpam-ldap/examples/pam.d/shutdown
/usr/share/doc/libpam-ldap/examples/pam.d/rexec
/usr/share/doc/libpam-ldap/examples/pam.d/rlogin
/usr/share/doc/libpam-ldap/examples/pam.d/kde
/usr/share/doc/libpam-ldap/examples/pam.d/xserver
/usr/share/doc/libpam-ldap/examples/chsh
/usr/share/doc/libpam-ldap/examples/chfn
/usr/share/doc/libpam-ldap/examples/pam.conf
/usr/share/doc/libpam-ldap/changelog.Debian.gz
/usr/share/doc/libpam-ldap/buildinfo.gz
/usr/share/doc/libpam-ldap/README.gz
/usr/share/libpam-ldap/ldap.conf
/usr/share/man/man5/pam_ldap.conf.5.gz
/lib/security/pam_ldap.so
```

Next, **/etc/pam.d/common-auth** is modified to allow authentication against LDAP so it contains

```
#auth required pam_unix.so nullok_secure

auth sufficient pam_unix.so nullok_secure
auth sufficient pam_ldap.so try_first_pass
auth required pam_deny.so
```

Also, **/etc/pam_ldap.conf** is modified by adding the following section.

```
# HUBzero Mappings
nss_base_passwd ou=users,?one
nss_base_shadow ou=users,?one?host=web
pam_filter host=web
pam_password crypt
nss_map_attribute uniqueMember member
nss_base_group ou=groups,dc=myhub,dc=org?sub
```

NSS Configuration to Use OpenLDAP

The NSS module for using LDAP as a naming service is downloaded from debian.org and installed using the following command:

```
# apt-get install libnss-ldapd
```

This adds the following files to the file system:

```
/usr/sbin/nslcd
/usr/share/doc/libnss-ldapd/copyright
/usr/share/doc/libnss-ldapd/NEWS.gz
/usr/share/doc/libnss-ldapd/AUTHORS
/usr/share/doc/libnss-ldapd/README.gz
/usr/share/doc/libnss-ldapd/examples/
/usr/share/doc/libnss-ldapd/examples/nss-ldapd.conf.gz
/usr/share/doc/libnss-ldapd/changelog.gz
/usr/share/man/man8/nslcd.8.gz
/usr/share/man/man5/nss-ldapd.conf.5.gz
/lib/libnss_ldap.so.2
```

Next, the file **/etc/nss-ldapd.conf** is created with `root:nslcd` ownership. The only lines that are enabled are

```
uid nslcd
gid nslcd
uri ldap://127.0.0.1
base dc=myhub,dc=hubzero,dc=org
```

Finallt, NSS configuration file **/etc/nsswitch.conf**

is modified by adding ldap to the first three lines:

```
passwd:         compat ldap
group:          compat ldap
shadow:         compat ldap

hosts:          files dns
networks:       files

protocols:      db files
services:       db files
ethers:         db files
rpc:            db files

netgroups:      nis
```

LDAP Services Restart

All LDAP services are restarted:

```
# /etc/init.d/slapd restart
# /etc/init.d/nslcd restart
# /etc/init.d/nscd restart
```

Test

```
# getent passwd
```

To test configuration. You should see entries for users 'hubrepo' and 'apps' toward the end of the list if everything is working correctly.

Troubleshooting

If you have a problem with the system apparently not recognizing up to date account or group information (eg., in the next section some people report receiving an error about unknown username 'hubadmin') you can nscd to flush it data cache and restart using the following commands:

```
# nscd -i passwd
# nscd -i group
# /etc/init.d/nscd restart
# getent passwd
```

If you still don't see the hubadmin account listed then re-read the instructions and check your work very carefully. These instructions assume a fresh install, if you are working with an existing LDAP/PAM/NSS installation you will have to do more advanced troubleshooting outside the scope of this documentation.

Home Directories Creation

Home directories for the apps and admin users are created:

```
# mkdir /home/myhub/apps
# chown apps.public /home/myhub/apps
# chmod 0700 /home/myhub/apps
# mkdir -p /home/myhub/admin
# chown admin.public /home/myhub/admin
# chmod 0700 /home/myhub/admin
```

MySQL

Installation

The default MySQL server on Debian is used but the HUBzero MySQL package is downloaded from packages.hubzero.org and installed using the following command:

```
# apt-get install hubzero-mysql
```

This adds the following files to the file system:

```
/usr/share/doc/hubzero-mysql/changelog.Debian.gz
/usr/share/doc/hubzero-mysql/copyright
/usr/share/hubzero-mysql/hubzero-fix-debian-maint.sql
```

Next, the MySQL root password is set to the one specified in the six questions.

Bug Fix

A bug in the Debian installation that disallows debian-sys-maint account to GRANT privileges is fixed via the following command:

```
# /usr/bin/mysql --defaults-file=/etc/mysql/debian.cnf -D mysql
```

Apache

Installation & Configuration

The default Apache on Debian is used but the `hubzero-apache2` HUBzero Apache package is downloaded from packages.hubzero.org and installed using the following command:

```
# apt-get install hubzero-apache2
```

This adds the following files to the file system:

```
/var/log/apache2/daily/imports
/usr/share/doc/hubzero-apache2/changelog.Debian.gz
/usr/share/doc/hubzero-apache2/README.Debian
/usr/share/doc/hubzero-apache2/copyright
/etc/apache2/sites-available/hub
/etc/apache2/sites-available/hub-ssl
```

Next, the hub Apache configuration files are enabled:

```
# /usr/sbin/a2ensite hub
# /usr/sbin/a2ensite hub-ssl
```

Finally, Apache is restarted:

```
apache2ctl restart
```

PHP

Configuration

The following lines in the PHP configuration file `/etc/php5/apache2/php.ini` is modified from

```
display_errors = On
log_errors = Off
```

to

```
display_errors = Off
log_errors = On
```

and permission/ownership changed as follows.

```
# chmod 0644 /etc/php5/apache2/php.ini
# chown root.root /etc/php5/apache2/php.ini
```

Test

```
# echo "<?php phpinfo();?>" > /var/www/index.php
```

Go to "/index.php" on your site and you should see a php status page.

Delete the test page when you are done.

```
# rm /var/www/index.php
```

CMS

Global HUBzero Configuration

Create the file `/etc/hubzero.conf` using values from the initial six questions (assuming that the user chose the default "myhub" as the name for the hub):

```
[default]
site=myhub

[myhub]
DocumentRoot=/www/myhub
HubName=myhub
HubHost=myhub.org
BaseDN=dc=myhub,dc=org
Org=dc=myhub.org
EmailAddress=
ContainerArch=i386
ConainerDebianRepo=lenny
ContainerHubzeroRepo=buck
```

CMS Installation

The Joomla! content management system is downloaded from packages.hubzero.org and installed using the following command:

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

and the web publishing area created:

```
# mkdir /www/myhub
# cp -rp /usr/lib/hubzero/cms/* /www/myhub
# chown -R www-data.www-data /www/myhub
```

Database Creation

MySQL databases for the CMS and for HUBzero metrics are created:

```
CREATE DATABASE `myhub`;
CREATE DATABASE `myhub_metrics`;
```

```
GRANT ALL PRIVILEGES ON `myhub`.* TO \'myhub
\'@\'%\' IDENTIFIED BY \'<password>\';
GRANT ALL PRIVILEGES ON `myhub_metrics`.* TO \'myhub
\'@\'%\' IDENTIFIED BY \'<password>\';
FLUSH PRIVILEGES;
```

where `password` is the CMS "admin" user password specified in the six questions step.

CMS Configuration

1. The HUBZero package `hubzero-cms-setup` is downloaded from packages.hubzero.org and installed using the command

   ```
   # apt-get install hubzero-cms-setup
   ```

   This adds the following files to the file system:

   ```
   /usr/share/hubzero-cms-setup/old-default-hub-db.sql
   /usr/share/hubzero-cms-setup/admin_user.sql
   /usr/share/hubrepo_user.sql
   /usr/share/groups.sql
   /usr/share/configuration.php.tmpl
   /usr/share/make_joomla_password.php.tmpl
   /usr/share/get_tools_params.sql
   /usr/share/get_contrib_params.sql
   /usr/share/hubzero-cms-setup/clean_params.sh
   /usr/share/doc/hubzero-cms-setup/copyright
   /usr/share/doc/hubzero-cms-setup/changelog.Debian.gz
   /usr/share/doc/hubzero-cms-setup/README.Debian
   ```

2. A number of SQL files are used to populate MySQL tables. These include

   ```
   /usr/share/hubzero-cms-setup/default-hub-db.sql
   /usr/share/hubzero-cms-setup/admin_user.sql
   /usr/share/hubzero-cms-setup/hubrepo_user.sql
   /usr/share/hubzero-cms-setup/groups.sql
   /usr/share/hubzero-cms-setup/get_tools_params.sql
   /usr/share/hubzero-cms-setup/get_contrib_params.sql
   ```

```
/usr/share/hubzero-cms-joomla-installation/cms/installation/sql/m
ysql/hz_sample_data.sql (where does /h-c-j-i dir come from?)
/usr/share/hubzer-cms-joomla-
installation/cms/installation/sql/mysql/hubzero.sql
```

3. The CMS installation directory **/var/www/myhub/installation** is removed.
4. The hubname and password is set in MySQL for the tools and contribtools components using the values specified in the six questions step initially.
5. `/var/www/myhub/hubconfiguration.php, configuration.php` is generated with values from the initial, six questions step.

Subversion

Configuration

This step executes the following:

```
# install --owner www-data --group www-
data --mode 0770 -d /opt/svn/tools
# touch /etc/apache2/svn.conf /etc/apache2/svn.bak
# chown www-data /etc/apache2/svn.conf /etc/apache2/svn.bak
```

Test

```
# svnadmin create /opt/svn/tools/test --fs-type fsfs
# chown -R www-data.www-data /opt/svn/tools/test
# echo "<Location /tools/test/svn>
        DAV svn
        SVNPath /opt/svn/tools/test
        AuthType Basic
        AuthBasicProvider ldap
        AuthName "Test"
        AuthzLDAPAuthoritative on
        AuthLDAPGroupAttributeIsDN on
        AuthLDAPGroupAttribute owner
        AuthLDAPGroupAttribute member
        AuthLDAPURL ldap://localhost/ou=users,dc=myhub,dc=org
        Require ldap-group gid=apps,ou=groups,dc=myhub,dc=org
</Location&gt" > /etc/apache2/svn.conf
# /etc/init.d/apache2 restart
```

Be sure to the BASEDN in the above to match that used by your configuration.

Now browse to "/tools/test/svn" using an https connection and you should get prompted for a username and password, use the apps account you created earlier when you installed LDAP. You should see "svn - Revision 0: /".

Delete test file.

```
# echo "" > /etc/apache2/svn.conf
# rm -fr /opt/svn/tools/test
# /etc/init.d/apache2 restart
```

WebDAV

WebDAV Modules Deployment

```
# a2enmod dav_fs
# a2enmod dav_lock
# /etc/init.d/apache2 restart
```

WebDAV Configuration

The `/etc/apache2/sites-available/hub-ssl`

file is edited to specify the hub DN. The relevant line in the files looks like:

```
        <Directory /webdav>
        ...

AuthLDAPURL ldap://localhost/ou=users,dc=myhub,dc=org?uid
        ...
        </Directory>
```

Apache is restarterd to enable changes.

```
# /etc/init.d/apache2 restart
```

Test

```
# install --owner www-data --group www-
data --mode 0770 -d /webdav/home/apps
# touch /webdav/home/apps/test
```

Browse to your site's https /webdav address (e.g. https://myhub/webdav). You should get prompted for a username and password. Use the apps account. You should see a directory listing including the file "test".

Now test using a WebDAV client.

```
# apt-get install cadaver
# cadaver https://myhub.org/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 'apps' 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
# rm /webdav/home/apps/test
# rmdir /webdav/home/apps /webdav/home /webdav
```

Usermap

Installation

The hubzero-usermap package is downloaded from hubzero-usermap packages.hubzero.org and installed using the following command:

```
# apt-get install hubzero-usermap
```

which adds the following files to the file system:

```
/sbin/mount.usermap
/usr/share/doc/hubzero-usermap/
/usr/share/doc/hubzero-usermap/changelog.Debian.gz
/usr/share/doc/hubzero-usermap/copyright
```

Configuration

1. The installation scripts add the following lines to the /etc/auto.master file:

   ```
   /webdav/home /etc/auto.webdav --timeout=60
   ```

2. **/etc/auto.webdav** is configured with the following line:

   ```
   * -fstype=usermap,user=www-data,allow_other :&
   ```

3. autofs is started:

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

4. **fuse** is added to the /etc/modules file so that it is loaded upon startup.

   This automounts a usermap filesystem of a users home directory inside of /webdav/home on demand. This version of the users home directory is owned and accessible to the user www-data which allows WebDAV to serve its contents.

Test

```
# touch /home/myhub/apps/test
# ls -l /webdav/home/apps
```

You should see a list of files in apps's home directory ("test") which will appear to be owned by www-data.www-data

```
# mount -l
```

You should see something like:
mount.usermap on /webdav/home/apps type fuse.mount.usermap
(rw,nosuid,nodev,allow_other)

Finally clean up.

```
# umount -f /webdav/home/apps
# rm /webdav/home/apps/test
```

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

Trac

Authentication Plugin Installation

The MySQL authentication plugin for Trac is installed from packages.hubzero.org using:

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

and Apache restarted:

```
# /etc/init.d/apache2 restart
```

Test

```
# svnadmin create /opt/svn/tools/test --fs-type fsfs
# chown -R www-data.www-data /opt/svn/tools/test
# trac-admin /opt/trac/tools/test initenv "Test" "sqlite:db/trac.db" "svn" "/opt/svn/tools/test"
# chown -R www-data.www-data /opt/trac/tools/test
```

Now browse to "/tools/test/wiki" using an https connection; you should see a default Trac project page.

Delete test data.

```
# rm -fr /opt/svn/tools/test
# rm -fr /opt/trac/tools/test
# /etc/init.d/apache2 restart
```

addrepo

Installation

The hubzero-addrepo package is installed from packages.hubzero.org:

```
# apt-get install hubzero-addrepo
```

which adds the following files:

```
/opt/svn/tools/
/opt/trac/tools/
/etc/apache2/svn.conf
/etc/apache2/svn.bak
/usr/share/doc/hubzero-addrepo/
/usr/share/doc/hubzero-addrepo/changelog.Debian.gz
/usr/share/doc/hubzero-addrepo/copyright
/usr/share/hubzero-addrepo/images.txt.in
/usr/share/hubzero-addrepo/image.py.in
/usr/share/hubzero-addrepo/link.py.in
/usr/share/hubzero-addrepo/templates/
/usr/share/hubzero-addrepo/templates/site_footer.cs
/usr/share/hubzero-addrepo/templates/site_header.cs
/usr/share/hubzero-addrepo/templates/site_css.cs
/usr/share/hubzero-addrepo/templates/site.html
/usr/share/hubzero-addrepo/svn/
/usr/share/hubzero-addrepo/svn/branches/
/usr/share/hubzero-addrepo/svn/trunk/
/usr/share/hubzero-addrepo/svn/trunk/doc/
/usr/share/hubzero-addrepo/svn/trunk/examples/
/usr/share/hubzero-addrepo/svn/trunk/rappture/
/usr/share/hubzero-addrepo/svn/trunk/src/
/usr/share/hubzero-addrepo/svn/trunk/bin/
/usr/share/hubzero-addrepo/svn/trunk/data/
/usr/share/hubzero-addrepo/svn/trunk/middleware/
/usr/share/hubzero-addrepo/svn/tags/
/usr/share/hubzero-addrepo/cover.txt.in
/usr/share/hubzero-addrepo/getstart.txt.in
/usr/share/hubzero-addrepo/ldapconf.in
/usr/bin/gensvnapache
/usr/bin/addrepo
```

Configuration

sudo privileges are configured by adding the following lines to `/etc/sudoers`:

```
www-
data        ALL=(apps)NOPASSW
D:/bin/bash /www/myhub/components/com_contribtool/scripts/*tool.php *
www-
data        ALL=(apps)N
OPASSWD:/usr/bin/expect /www/myhub
/components/com_contribtool/scripts/*tool.php *
www-data        ALL=NOPASSWD:/etc/init.d/apache2
```

This is so that the web process can run a number of scripts as the "apps" user.

iptables

Installation

The HUBzero firewall package hubzero-firewall is installed from packages.hubzero.org:

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

which installs the following files:

```
/usr/share/doc/hubzero-firewall/copyright
/usr/share/doc/hubzero-firewall/changelog.Debian.gz
/usr/share/doc/hubzero-firewall/README.Debian
/etc/hubzero/firewall_on
/etc/hubzero/firewall_off
```

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. 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).

Maxwell Service

Installation

The HUBzero Maxwell middleware services package hubzero-mw-service is installed from packages.hubzero.org:

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

which adds the following files:

```
/usr/lib/hubzero/bin/maxwell_service
/usr/lib/hubzero/bin/set_quotas
/etc/vz/conf/hub-session-4.0-i386.conf
/etc/vz/conf/hub-session-4.0-i386.mount
/etc/vz/conf/hub-session-4.0-i386.umount
/etc/vz/conf/hub-session-5.0-i386.conf
/etc/vz/conf/hub-session-5.0-i386.mount
/etc/vz/conf/hub-session-5.0-i386.umount
/etc/vz/conf/hub-session-4.0-amd64.conf
/etc/vz/conf/hub-session-4.0-amd64.mount
/etc/vz/conf/hub-session-4.0-amd64.umount
/etc/vz/conf/hub-session-5.0-amd64.conf
/etc/vz/conf/hub-session-5.0-amd64.mount
/etc/vz/conf/hub-session-5.0-amd64.umount
/var/log/hubzero/sessions
/usr/lib/hubzero/bin/mkvztemplate
/etc/hubzero/quota.conf
/etc/rc.boot
```

Configuration

1. A self-signed SSL key for use by xvnc is created:

   ```
   # cd /etc/hubzero
   # openssl req -new -x509 -days 365 -nodes -out xvnc.pem.cert -keyout xvnc.pem.key
   ```

2. The Maxwell service configuration file is created and installed in `/etc/hubzero/maxwell.conf`:

```
mysql_host = "localhost"
mysql_user = "myhub"
mysql_password = "<myhub user's pw>"
mysql_db = "myhub"
filexfer_decoration="""filexfer_sitelogo { <h1><a href="http://hu
bzero.org/" title="HUBzero home page"><span>HUBzero.org: online s
imulations and more</span></a></h1> }
filexfer_stylesheet http://$huburl/templates/filexfer/upload.css

hub_name="myhub"
hub_url="http://myhub.org"
hub_homedir="/home/myhub"
hub_template="hubbasic"
```

3. The host table is initialized in the database:

   ```
   MySQL> INSERT INTO 'host' VALUES ('localhost',27,'up',1,1);
   ```

4. The OpenVZ template for creating containers dynamically is created:

   ```
   # /usr/lib/mw/bin/mkvztemplate amd64 lenny
   ```

Test

```
# /usr/lib/hubzero/bin/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/hubzero/startxvnc: Becoming nobody.
/usr/lib/hubzero/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

```
# /usr/lib/hubzero/bin/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

Installation

The Maxwell client is installed through the hubzero-mw-client package from packages.hubzero.org:

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

This adds the following files:

```
/usr/share/hubzero-mw-client
/usr/lib/hubzero/bin/maxwell
/usr/share/hubzero-mw-client/maxwell.conf-dist
```

Configuration

1. SSH keys for the Maxwell client are created and installed in `/etc/hubzero`:

   ```
   # ssh-keygen -t rsa -f /etc/hubzero/maxwell.key -N '' -C www-data@`hostname`
   # ssh-keygen -t rsa -f /etc/hubzero/notify.key -N '' -C root@`hostname`

   # chown www-data:www-data /etc/hubzero/maxwell.key
   # chown www-data:www-data /etc/hubzero/maxwell.key.pub
   # chown www-data:www-data /etc/hubzero/notify.key
   # chown www-data:www-data /etc/hubzero/notify.key.pub

   # chmod 0400 /etc/hubzero/maxwell.key
   # chmod 0400 /etc/hubzero/maxwell.key.pub
   # chmod 0400 /etc/hubzero/notify.key
   # chmod 0400 /etc/hubzero/notify.key.pub
   ```

2. SSH is configured to allow notify and Maxwell keyed clients to connect as www-data and root:

   ```
   # mkdir -p /root/.ssh
   # cat /etc/mw/maxwell.key.pub >>  /root/.ssh/authorized_keys

   # mkdir -p ~www-data/.ssh
   ```

```
# grep -q -f /etc/hubzero/notify.key.pub ~www-data/.ssh/authorize
d_keys || (echo -n "COMMAND=\"/usr/lib/hubzero/bin/maxwell notify
\" " >> ~www-data/.ssh/authorized_keys
# cat /etc/hubzero/notify.key.pub >> ~www-
data/.ssh/authorized_keys
# chown www-data.www-data ~www-data/.ssh
# chmod 0700 ~www-data/.ssh
# chown www-data.www-data ~www-data/.ssh/authorized_keys
# chmod 0400 ~www-data/.ssh/authorized_keys
```

3. The sample `maxwell.conf` file is deployed:

```
#  cp /usr/lib/mw/maxwell.conf-dist  /etc/hubzero/maxwell.conf
```

4. `/etc/mw/maxwell.conf` is modified to configure the following variables (using values specified in the six questions step initially):

```
mysql_host = "localhost"
mysql_user="myhub"
mysql_password="<MySQL password>"
mysql_db="myhub"

default_vnc_timeout=86400
session_suffix="L"

fi
lexf
er_decor
ation="""filexfe
r_sitelogo { <h1><a href="http:/
/myhub.org/" title="myHUB home page"><span>myhub.org
: online simulations and more</span></a></h1> }
filexfer_stylesheet http://myhub.org
/templates/filexfer/upload.css"
"""
hub_name="myhub"
hub_url="http://myhub.org/"
hub_homedir="/home/myhub"
hub_template="hubbasic"
```

- Insert values for localhost in the MySQL 'host' table. ???

Test

```
# su www-data
$ ssh -i /etc/mw/maxwell.key root@localhost ls
The authenticity of host 'localhost (127.0.0.1)' can't be establi
shed.
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
 hosts.
list of files
$ exit
#
```

VNC Client

Installation

The VNC client is downloaded from packages.hubzero.org and installed:

```
# apt-get install tightvnc-java
```

This adds the following files:

```
/usr/share/man/man1/jtightvncviewer.1.gz
/usr/share/doc/tightvnc-java/copyright
/usr/share/doc/tightvnc-java/changelog.Debian.gz
/usr/share/doc/tightvnc-java/README.gz
/usr/share/tightvnc-java/index.vnc
/usr/share/tightvnc-java/*.class (32 files)
/usr/share/java/tightvncviewer-1.3.8.jar
/usr/share/java/signed-tightvncviewer-1.3.8.jar
/usr/bin/jtightvncviewer
/usr/share/tightvnc-java/SignedVncViewer.jar
/usr/share/tightvnc-java/VncViewer.jar
/usr/share/java/tightvncviewer.jar./usr/share/java/signed-
tightvncviewer.jar
```

vncproxy

Installation

HUBzero vncproxy packages are installed form packages.hubzero.org:

```
# apt-get install hubzero-vncproxy
# apt-get install hubzero-vncproxy2
```

This adds the following files:

```
/usr/share/doc/hubzero-vncproxy2/copyright
/usr/share/doc/hubzero-vncproxy2/changelog.Debian.gz
/usr/share/doc/hubzero-vncproxy2/README.Debian
/usr/lib/hubzero/bin/vncproxy2-helper
/usr/lib/hubzero/bin/vncproxy2
/etc/default/vncproxy2
/etc/init.d/vncproxy2
/usr/share/doc/hubzero-vncproxy/changelog.Debian.gz
/usr/share/doc/hubzero-vncproxy/copyright
/usr/bin/
/usr/bin/vncproxy
/usr/lib/hubzero/bin/vncproxy.py
/etc/init.d/vncproxy
```

expire

Installation

The hubzero-expire package is installed from packages.hubzero.org:

```
# apt-get install hubzero-mw-expire
```

This adds the following files:

```
/usr/share/doc/hubzero-mw-expire/changelog.Debian.gz
/usr/share/doc/hubzero-mw-expire/copyright
/usr/lib/hubzero/bin/expire-sessions.py
/etc/init.d/expire-sessions
```

telequotad

Installation

The `hubzero-telequotad` package is installed from packages.hubzero.org:

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

This compiles telequotad and installs the following files:

```
/usr/share/doc/hubzero-telequotad/changelog.Debian.gz
/usr/share/doc/hubzero-telequotad/copyright
/usr/sbin/telequotad
/usr/lib/hubzero/bin/telequotad
/etc/hubzero/telequotad.conf
/usr/lib/hubzero/bin/trim
/etc/init.d/telequotad
```

App Container

Configuration

Users in the "apps" group are granted permission to manage HUBzero Apps by adding the following lines to `/etc/sudoers`:

```
%apps  ALL=NOPASSWD:/bin/su - apps
```

Workspace

Installation

The workspace app (tool) is installed using packages from packages.hubzero.org:

```
# apt-get install hubzero-app
# apt-get install hubzero-app-workspace
```

This adds the following files:

```
/usr/share/doc/hubzero-app/copyright
/usr/share/doc/hubzero-app/changelog.Debian.gz
/usr/bin/hubzero-app
```

The Workspace tool is then published:

```
# hubzero-app setup
# hubzero-
app install --publish /usr/lib/hubzero/apps/workspace-1.0.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.

Installation (RHEL 6)

rhel6

The documentation for RedHat 6 installation is currently located here:

https://hubzero.org/documentation/1.3.0/installationrh

It will be moved to the current page in the near future.

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.

Hubgraph - Alternate Search Engine

Shibboleth/InCommon - Authentication Plug-In

Hubgraph search

Prerequisites

Hubgraph is a node.js module, requiring an installation of node.js of at least the 0.10.x branch, and npm, the node package manager.

If your distribution does not package an appropriate version of node, see [the node.js downloads page]

The current reference version of node.js is 0.10.29, the latest version packaged by Debian as of this writing.

There are also build dependencies. npm is a source-based package manager. These packages are:

libboost-filesystem-dev libboost-regex-dev libboost-serialization-dev libboost-graph-dev libboost-system-dev libicu-dev

These should be available through your distribution's package manager.

Installation

The recommended way to install hubgraph is with npm through its global repository:

```
npm install hubgraph-hubzero
```

This installs a few packages we maintain, with git repositories listed here:

hubgraph-hubero	https://bitbucket.org/snyder13/hubgraph-hubzero
hubgraph-ext	https://bitbucket.org/snyder13/hubgraph-ext
hubgraph	https://bitbucket.org/snyder13/hubgraph

Installing from git is not recommended. The latest `master` copy of each branch is not guaranteed to be mutually compatible with the other branches. The npm modules are synced with each other to guarantee a common vintage.

If you do install from git, npm is still required: cd to the `source` directory and run `npm install .`

Configuration

Step-by-step instructions for configuring hubgraph are available through the package:

```
node -e 'require("hubgraph-hubzero").install()'
```

A few notes:

- `node` may be called`nodejs`, particularly on Debian installations
- You might need to supply the path where `npm` installed hubgraph if it happens to be different from the path your `node` executable looks, eg: NODE_PATH=/usr/local/lib/node_modules nodejs -e 'require("hubgraph-hubzero").install()'

Running

```
node -e "require('hubgraph-hubzero').server()"
```

Help?

If you have any problems please visit https://hubzero.org/support and submit a support ticket describing the situation. Please include specific details about your Linux distribution and your version of node.js and npm.

Shibboleth authentication

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.

If those are steps you're prepared to take, carry on.

Download

- git repository
- source tarball
- Debian Wheezy package (architecture independent)
- Redhat 6.x package (architecture independent) (note dependencies section)

Dependencies

If you installed the hubzero-shibboleth package on Debian, you're set. The relevant packages were included as dependencies. The packages are:

shibboleth2-sp-utils shibboleth2-sp-schemas libapache2-mod-shib2

At the time of this writing, Shibboleth is not distributed in the core repositories for Redhat/CentOS. You can read about how to add a repo that has what you need here:

https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPLinuxRPMInstall

Generate a private key

Use shib-keygen to generate /etc/shibboleth/sp-key.pem. Note that this utility may not be on your path unless you are root.

Configure Shibboleth

Shibboleth official quickstart documentation reference

The main configuration file is located at /etc/shibboleth/shibboleth2.xml. There are some other files that might be of interest to you here, but the defaults are acceptable to get your hub working with InCommon.

In shibboleth2.xml:

- Update <ApplicationDefaults entityID="{url}" ...> so {url} is https://{your hostname}/Shibboleth.sso. This is your Shibboleth endpoint, designated later by the Apache configuration as the location where the shib2 module will manage communication with ID providers.
- Update <Sessions ...handlerSSL="false" ...> to handlerSSL="true", if it is not already

Configure Apache

Shibboleth official Apache configuration reference

Ensure that Apache is loading the module. Typically this means that there is a link in mods-enabled to shib2.load in mods-available

ln -s /etc/apache2/mods-available/shib2.load /etc/apache2/mods-enabled

If you do not have this directory structure you can also enable the module directly in the next step by adding this to your Apache configuration file:

LoadModule mod_shib /usr/lib/apache2/modules/mod_shib2.so

In the conf file defining your SSL host, (usually located in /etc/apache2/sites-enabled):

- If not already set in the SSL <VirtualHost> UseCanonicalName on;
- To enable shibd's endpoint, add: <Location /Shibboleth.sso> SetHandler shib </Location>

- Joomla! routing will stomp on /Shibboleth.sso unless you change the mod_rewrite rules a bit.
    - You should have a line like: RewriteRule (.*) index.php probably preceded by a few 'RewriteCond's. Add a new condition to exempt the shib2-controlled path:

RewriteCond %{REQUEST_URI} !/Shibboleth.sso/.*$ [NC]

Restart apache: /etc/init.d/apache2 restart

Verify

From the same host (this is IP-restricted):

wget -q --no-check-certificate https://localhost/Shibboleth.sso/Metadata -O - | tee /etc/shibboleth/sp-metadata.xml

This command should write XML to the listed file (and stdout) wrapped in <md:EntityDescriptor xmlns:md="urn:oasis:names:tc:SAML:2.0:metadata" ...>

If it does not, review the references above to troubleshoot.

You may skip to "Configuring Joomla!" if you do not want to test interop more thoroughly with the TestShib ID provider, but I would recommend you do this test.

Upload metadata to TestShib.org

- Copy the metadata generated above to some unique name, for example:

cp /etc/shibboleth/sp-metadata ~/{your hostname}-sp-metadata.xml

- Upload that file here: https://www.testshib.org/register.html. Uploading a file of the same name will overwrite it on the testshib server, should you need to make any adjustments.

Change your local configuration to accept TestShib as an ID provider

- Visit this URL to get an appropriate test configuration XML file:

https://www.testshib.org/cgi-bin/sp2config.cgi?dist=Others&hostname={your hostname from the Shibboleth Configuration step above})

- Assuming that looks OK, copy the output over your existing /etc/shibboleth/shibboleth2.xml:

  wget -q --no-check-certificate "https://www.testshib.org/cgi-bin/sp2config.cgi?dist=Others&hostname={your hostname}" -O /etc/shibboleth/shibboleth2.xml

Restart services: /etc/init.d/shibd restart && /etc/init.d/apache2 restart

Configuring Joomla!

If you do not already have plg_authentication_shibboleth installed, this package installs a tarball in $(PREFIX)/usr/lib/hubzero that you may install using Joomla!'s package management interface at /administrator.

If you are building from git, running make in the source directory will build this file.

Manage ID providers on Joomla!'s admin page

In Extensions->Plugins, select Authentication - Shibboleth

Ideally it should look a lot like the screenshot in that it found testshib in your XML configuration. If so, you can click the down arrow by that entry to move it into your active provider list.

This may fail if, for example, shibboleth2.xml is not readable by the web user, or if you changed your configuration so that the file is located somewhere unexpected.

It is not necessary, however, for the web server to read this file. If you'd like you can simply enter the EntityID for testshib (https://idp.testshib.org/idp/shibboleth) in the white box with the button labeled "Add ID provider". Enter something, eg "TestShib" for the label.

Quick run-down of the fields here:

- Entity id: (required) corresponds to the corresponding entityId in shibboleth2.xml and must match exactly for things to work out.
- Label: (required) name to show on the log-in button of your hub for this provider
- Initialism: (optional) if you have more than ten supported ID providers, the log-in list becomes searchable, and in this case you can add a short name for institutions so that they will come up when the user types that as well as when they type a portion of the label. (For example, if you federated with the National

Science Foundation you might add "NSF" here)
- Host: (optional) institutions may be pre-selected if the IP address of the user looks like it is in a particular network, eg, to follow the previous example, nsf.gov to pre-select the National Science Foundation
- Logo: (optional) also shown on the button. Enter a URL here to make a iconified copy of it. You may have better results in some cases if you resize to no more than 28px in either extent yourself.

Finally, you can select the order in which you would like the button to appear on the login page here. When you're done, click "Save & Close" in the top right. This will take you back to the screen where you can click the icon in the "Status" column to enable the plug-in.

Try logging in!

(If you run into any problems here, there might be a clue in TestShib's logs of its ID provider actions)

Since TestShib doesn't release any attributes, you'll have to enter a name when you log in. Hopefully you can negotiate to get names and emails released to your hub with "real" ID providers, which you're now clear to do if everything worked out.

Help?

- If you have a problem that you can't resolve that appears to be related to Shibboleth's machinery, please consult the official documentation carefully.

- If you can't resolve the problem, there is a mailing list: https://shibboleth.net/community/lists.html

-

  If the problem you are experiencing appears to be related not to the Shibboleth interchange mechanism but to something in the hub's implementation of the log-in procedure, visit https://hubzero.org/support to enter a support ticket describing the situation.

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
```

Upgrading

Introduction

It will be possible to automatically upgrade to the 1.3.0 release from the 1.2.x and 1.1.x releases via a simple package upgrade and upgrade script. The upgrade script is still being developed and will be release along with detailed instructions here in the next few weeks.

It will NOT be possible to automatically upgrade to the 1.3.0 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.

Updates

Introduction

You can update a HUBzero 1.3.1 instance to a newer revision of HUBzero 1.3.1 through a relatively simple command line process

First update the hubzero-cms-1.3.1 package

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

Then update you HUBzero instance

```
# hzcms update
```

The "hzcms update" step is what copies the latest version of the HUBzero CMS 1.3.1 source code into your installed instance and updates the database schema as appropriate. It also fixes any other issues it detects.

Source Code

How to get source code

HUBzero is an Open Source project. Source code for each distributed package is available. There are source packages for Debian 6 and 7 as well as RedHat Enterprise Server 6. Additionally some parts are also available via GitHub (with more coming soon). See below for details.

Debian Packages

Source code for the HUBzero Debian packages is available via Debian Source Packages. These can be obtained using standard Debian package management tools. First you need to configure the location of the HUBzero source package repository by adding the following line to /etc/apt/sources.list

For Debian 6:

```
deb-src http://packages.hubzero.org/deb diego-deb6 main
```

For Debian 7:

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

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
```

Once the public key for http://packages.hubzero.org has been installed, you can request the hubzero source code for each package:

```
apt-get source packageName
```

where *packageName* is the desired package from the table below. Not all are necessarily used for any given installation.

Package Name	*Purpose*
hubzero-app	Command line tool to manage installation/publishing of HUBzero apps
hubzero-app-workspace	HUBzero App providing a lightweigth Linux desktop, for app/tool development
hubzero-chuse	HUBzero graphical front end to the user environment manager 'use'
hubzero-cli	HUBzero command line interface tools
hubzero-cms-1.3.1	The HUBzero Content Management System (based on Joomla! framework)
hubzero-expire-sessions	Expires unused app/tool sessions
hubzero-filexfer	Transfer files between App Sessions and user's desktop
hubzero-filexfer-xlate	Support daemon for the filexfer program
hubzero-firewall	HUBzero firewall that protects app/tool sessions
hubzero-forge	Creates project areas for tool development
hubzero-icewm	Linux ICE window manager configuration, used in workspaces
hubzero-icewm-captive	Linux ICE window manager specially crafted to support tools in a sessions
hubzero-icewm-themes	The HUBzero Linux ICE window manager theme, used in workspaces
hubzero-invokeapp	HUBzero application invoke script
hubzero-mailgateway	HUBzero mail gateway
hubzero-metrics	HUBzero metrics support package
hubzero-mw-client	HUBzero middleware - client
hubzero-mw-service	HUBzero middleware - execution host session manager
hubzero-mw-session	HUBzero middleware - per session tools
hubzero-mysql	MySQL configuration package for HUBzero
hubzero-openldap	OpenLDAP configuration package for HUBzero
hubzero-openvz	OpenVZ configuration package for HUBzero
hubzero-policyrcd	HUBzero policy-rc.d for invoke-rc.d
hubzero-python	HUBzero python API module
hubzero-rappture	The Rapid APPlication infrastrucTURE toolkit

	for building scientific tools
hubzero-rappture-session	Session support packages for Rappture
hubzero-ratpoison-captive	Linux window manager, used in app/tool sessions
hubzero-submit-client	HUBzero submit client
hubzero-submit-common	HUBzero submit common python library
hubzero-submit-condor	HUBzero condor build for hubzero-submit
hubzero-submit-distributor	HUBzero middleware submit distributor
hubzero-submit-monitors	HUBzero middleware submit monitors
hubzero-submit-pegasus	HUBzero pegasus build for hubzero-submit
hubzero-submit-server	HUBzero job submission server
hubzero-subversion	HUBzero Subversion support package
hubzero-telequotad	Disk quota monitor
hubzero-textifier	HUBzero textifier
hubzero-texvc	Helper utility to generate math fomulas for wiki markup
hubzero-tigervnc-server	X/VNC server for remote display of HUBzero tool sessions
hubzero-trac	HUBzero Trac support package
hubzero-trac-mysqlauthz	Plug-in for MySQL user auth in project development areas
hubzero-twm-captive	Linux TWM window manager, used in app/tool sessions
hubzero-use	Command for configuring the environment within a workspace
hubzero-use-apps	HUBzero apps environment for 'use'
hubzero-usermap	File permission mapping FUSE filesystem used by WebDAV
hubzero-vncproxy	HUBzero VNC proxy helper
hubzero-webdav	HUBzero WebDAV support package
libapache2-vncproxy	HUBzero vncproxy module for apache2-mpm-prefork
php5-oauth	OAuth PECL Library for PHP 5
php5-stem	Stemming PECL Library for PHP5

tightvnc-java	Modified VNC Client that receives app/tool sessions within a web browser

RedHat Packages

To be announced.

GitHub

The raw source code to the HUBzero CMS is available on github.com

https://github.com/hubzero/hubzero-cms on branch 1.3.1 (do not use the master branch)

Note that you can not install the HUBzero CMS using just this repository. You need to use the packages for Debian or Redhat.

Over time the full package source trees will get posted to github.