System Administrators

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

What is HUBzero?

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

Minimum System Requirements

HUBzero installations require one or more dedicated physical hosts running Debian GNU/Linux 6.0. In particular there are version specific requirements on mysql (5.1), python (2.6) and php (5.3).

Other distributions might theoretically work with some modification, although they would be totally unsupported.

A typical starter HUBzero installation might consist of a single physical server with dual 64-bit quad-core CPUs, 16 Gigabytes of RAM and a terabyte of disk.

It is possible to run HUBzero inside of a virtual machine such as ones created by VMware and VirtualBox. While fully functional there will would significant performance and resource limitations in such an environment. 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).

How To Get Source Code

This is optional
You'll find most of the source code within the web root of a working hub. But you can get source code for the middleware and all other parts of HUBzero by retrieving the source code packages using the debian package manager. To do so, first, update your package mechanism to pull packages from our HUBzero repository (codename 'manny' for this release). Edit your /etc/apt/sources.list and add the following line::

```
deb-src http://packages.hubzero.org/deb manny main contrib non-free
```

Then, install the hubzero archive key so that your system can verify packages from the hubzero archive. Run the following command as root:

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

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.

### Package List

The following packages (subject to change): are available from the HUBzero repository for this release. Not all are necessarily used for any given installation.

<table>
<thead>
<tr>
<th>Package Name</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>hubzero-app</td>
<td>Command line tool to manage installation/publishing of HUBzero apps</td>
</tr>
<tr>
<td>hubzero-app-workspace</td>
<td>HUBzero App providing a lightweight Linux desktop, for app/tool development</td>
</tr>
<tr>
<td>hubzero-chuse</td>
<td>HUBzero graphical front end to the user environment manager 'use'</td>
</tr>
<tr>
<td>hubzero-cli</td>
<td>HUBzero command line interface</td>
</tr>
<tr>
<td><strong>hubzero-cms</strong></td>
<td>The HUBzero Content Management System (based on Joomla! framework)</td>
</tr>
<tr>
<td>-----------------</td>
<td>---------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>hubzero-expire-sessions</strong></td>
<td>Expires unused app/tool sessions</td>
</tr>
<tr>
<td><strong>hubzero-filexfer</strong></td>
<td>Transfer files between App Sessions and user's desktop</td>
</tr>
<tr>
<td><strong>hubzero-filexfer-xlate</strong></td>
<td>Support daemon for the filexfer program</td>
</tr>
<tr>
<td><strong>hubzero-firewall</strong></td>
<td>HUBzero firewall that protects app/tool sessions</td>
</tr>
<tr>
<td><strong>hubzero-forge</strong></td>
<td>Creates project areas for tool development</td>
</tr>
<tr>
<td><strong>hubzero-icewm</strong></td>
<td>Linux ICE window manager configuration, used in workspaces</td>
</tr>
<tr>
<td><strong>hubzero-icewm-captive</strong></td>
<td>Linux ICE window manager specially crafted to support tools in app/tool sessions</td>
</tr>
<tr>
<td><strong>hubzero-icewm-themes</strong></td>
<td>The HUBzero Linux ICE window manager theme, used in workspaces</td>
</tr>
<tr>
<td><strong>hubzero-invokeapp</strong></td>
<td>HUBzero application invoke script</td>
</tr>
<tr>
<td><strong>hubzero-mailgateway</strong></td>
<td>HUBzero mail gateway</td>
</tr>
<tr>
<td><strong>hubzero-metrics</strong></td>
<td>HUBzero metrics support package</td>
</tr>
<tr>
<td><strong>hubzero-mw-client</strong></td>
<td>HUBzero middleware - client</td>
</tr>
<tr>
<td><strong>hubzero-mw-service</strong></td>
<td>HUBzero middleware - execution host session manager</td>
</tr>
<tr>
<td><strong>hubzero-mw-session</strong></td>
<td>HUBzero middleware - per session tools</td>
</tr>
<tr>
<td><strong>hubzero-openldap</strong></td>
<td>OpenLDAP configuration package for HUBzero</td>
</tr>
<tr>
<td><strong>hubzero-policyrcd</strong></td>
<td>HUBzero policy-rc.d for invoke-rc.d</td>
</tr>
<tr>
<td><strong>hubzero-python</strong></td>
<td>HUBzero python API module</td>
</tr>
<tr>
<td><strong>hubzero-rappture</strong></td>
<td>The Rapid APPlication infrastrucTURE toolkit for building scientific tools</td>
</tr>
<tr>
<td><strong>hubzero-rappture-session</strong></td>
<td>Session support packages for Rappture</td>
</tr>
<tr>
<td><strong>hubzero-ratpoison-captive</strong></td>
<td>Linux window manager, used in app/tool sessions</td>
</tr>
<tr>
<td><strong>hubzero-submit-client</strong></td>
<td>HUBzero submit client</td>
</tr>
<tr>
<td><strong>hubzero-submit-common</strong></td>
<td>HUBzero submit common python library</td>
</tr>
<tr>
<td><strong>hubzero-submit-condor</strong></td>
<td>HUBzero condor build for hubzero-submit</td>
</tr>
<tr>
<td><strong>hubzero-submit-distributor</strong></td>
<td>HUBzero middleware submit distributor</td>
</tr>
<tr>
<td><strong>hubzero-submit-pegasus</strong></td>
<td>HUBzero pegasus build for hubzero-submit</td>
</tr>
<tr>
<td><strong>hubzero-submit-server</strong></td>
<td>HUBzero job submission server</td>
</tr>
<tr>
<td><strong>hubzero-subversion</strong></td>
<td>HUBzero Subversion support package</td>
</tr>
<tr>
<td><strong>hubzero-telequotad</strong></td>
<td>Disk quota monitor</td>
</tr>
<tr>
<td><strong>hubzero-textifier</strong></td>
<td>HUBzero textifier</td>
</tr>
<tr>
<td><strong>hubzero-texvc</strong></td>
<td>Helper utility to generate math formulas for wiki markup</td>
</tr>
<tr>
<td><strong>hubzero-trac</strong></td>
<td>HUBzero Trac support package</td>
</tr>
<tr>
<td><strong>hubzero-trac-mysqauthz</strong></td>
<td>Plug-in for MySQL user auth in project development areas</td>
</tr>
<tr>
<td><strong>hubzero-twm-captive</strong></td>
<td>Linux TWM window manager, used in app/tool sessions</td>
</tr>
<tr>
<td><strong>hubzero-use</strong></td>
<td>Command for configuring the environment</td>
</tr>
</tbody>
</table>
hubzero-use-apps
within a workspace
HUBzero apps environment for 'use'

hubzero-usermap
File permission mapping FUSE filesystem used by WebDAV

Failed to write content for page "Installation"
HUBzero VNC proxy helper
Linux

Install Basic Operating System

The latest version of Debian GNU/Linux 6.0 (6.0.7 as of this writing) should be installed on each physical host used by a HUBzero installation.

To install Debian GNU/Linux, you can easily obtain a copy, and then follow the installation instructions for your architecture.

HUBzero has packaging support for amd64 (64bit) Intel architectures. i386 (32bit) packaging was not produced for this release due to lack of demand. Debian 7.0 is not supported at this time but will be in the next (1.2) release, scheduled for Fall 2013.

Installing Debian GNU/Linux using a a small bootable CD 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

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 app 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:

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

```bash
# 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:

```bash
# 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

```
deb http://packages.hubzero.org/deb manny main
```

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

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

Once the public key for http://packages.hubzero.org has been install you can then upgrade the current packages to their latest releases.

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

Install

# apt-get install -y mysql-server

You will be prompted to enter a MySQL administrative password.

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
System Administrators

leave blank

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

No

Delivery method for local mail

mbox format in /var/mail/

Split configuration into small files?

Yes

Test

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

# Mail -v someone@gmail.com
CMS

Install

# apt-get install -y hubzero-cms

Configure

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

**Install OpenLDAP**

# apt-get install -y slapd

You will be prompted to enter a LDAP administrative password.

**Install HUBzero LDAP support**

# apt-get install -y hubzero-openldap

Some packages will ask you to configure them when you run this step. I am pretty sure you can just accept the defaults because we rewrite the configuration files during the next step anyway.

Configuring nsldc: LDAP server URI:

keep the default or enter "ldap://localhost/"

Configuring nsldc: LDAP server search base:

keep the default

Configuring libnss-ldapd

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

**Configure OpenLDAP Database**

# hzldap init
# hzcms configure ldap --enable

To finish configuration you must manually (for now) trigger the initial synchronization of the CMS user/group tables with LDAP. Go to administrator section of your site (/administrator), go to
Site->Maintenance->LDAP and press the Export Users and Export Groups buttons in order to export all CMS users/groups. On a fresh install this would be consist solely of the user 'admin'.

Manually create a home directory for the CMS admin user now, it will make it easier to test things later to have this already setup.

```
# mkdir -p /home/example/admin
# touch /home/example/admin/mytest
# chown -R admin.users /home/example/admin
```

**Test**

```
# getent passwd
```

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

**Install**

These packages aren't working properly. Please don't install until this message is removed. If you have installed them already please purge them using the "purge" option instead of "install"

```
# apt-get install hubzero-ssh
# apt-get install hubzero-sftp
```

**Configure**

These modules aren't working properly. Please don't install until this message is removed. If you have configured them already please disable them using the "--disable" option instead of "--enable"

Support for basic ssh/sftp configuration was added with hubzero-cms_20130131.1

```
# hzcms configure ssh --enable
# hzcms configure sftp --enable
```
WebDAV

Install WebDAV

# apt-get install -y hubzero-webdav

Configure WebDAV

# hzcms configure webdav --enable

Test

# ls -l /webdav/home/admin
mytest

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 a directory listing including the file "mytest".

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

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

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

```bash
# 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. To install a 64 bit kernel run the command:

```
# apt-get install linux-image-2.6-openvz-amd64
```

For 32 bit kernels, run the command:

```
# apt-get install linux-image-2.6-openvz-686
```

Configure

You will need to reboot the server to activate the new kernel.

```
# reboot
```

Verify the new kernel is running (see "openvz" in the output of the command below)

For 64 bit kernels

```
# uname -a
Linux example.com 2.6.32-5-openvz-
amd64 #1 SMP Sat May 5 02:13:13 UTC 2012 x86_64 GNU/Linux
```

or for 32 bit kernels

```
# uname -a
Linux example.com 2.6.32-5-openvz-686 #1 SMP Sat May 5 02:13:13 UTC 2012 x86_64 GNU/Linux
```
If the new kernel is running you may remove the old (non-openvz) kernel

For 64 bit kernels (the exact version number may differ on your installation)

```bash
# apt-get purge linux-image-2.6.32-5-amd64
```

or for 32 bit kernels (the exact version number may differ on your installation)

```bash
# apt-get purge linux-image-2.6.32-5-i686
```

Test

```bash
# 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 squeeze manny

# hzcms configure mw-service --enable

Test

This test will show a failure the first time it is run. After the first session is created additional sessions should start without error. Be sure to stop each session you start using this test (as described below).

# 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 ...
Removing and repopulating: root etc var tmp dev
Mounting: /var/lib/vz/template/debian-5.0-amd64-maxwell home apps
VE is mounted
Setting CPU units: 1000
Configure meminfo: 2000000
VE start in progress...
TIME: 0 seconds.
Waiting for container to finish booting.
Got the vncpasswd
Adding auth for 10.51.0.1:0 and 10.51.0.1/unix: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 values on next start (use --save flag)

# vzlist

<table>
<thead>
<tr>
<th>VEID</th>
<th>NPROC</th>
<th>STATUS</th>
<th>IP_ADDR</th>
<th>HOSTNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6</td>
<td>running</td>
<td>10.51.0.1</td>
<td>-</td>
</tr>
</tbody>
</table>

# openssl s_client -connect localhost:4001

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

___
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.

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

# mount -oremount /
# /etc/init.d/quota restart

Test

# repquota -a

Should show disk usage for all users.
Workspace

Install

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

Test

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

Install

# apt-get install -y 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.

# chroot /var/lib/vz/template/debian-6.0-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 demonstration 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-distributor
# apt-get install hubzero-submit-server
# 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:

```
Session  Edit  View  Bookmarks  Settings  Help
```

<p>| | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ssh</td>
<td>distributor.sh</td>
<td>monitorJob.py</td>
<td>monitorTunnelA.py</td>
<td>server.py</td>
<td></td>
</tr>
<tr>
<td>BatchMonitors</td>
<td>environmentwhitelist.dat</td>
<td>monitorJobDB</td>
<td>monitorTunnelD.py</td>
<td>sites.dat</td>
<td></td>
</tr>
<tr>
<td>Scripts</td>
<td>environmentwhitelist.dft</td>
<td>monitorJobQ.py</td>
<td>monitorTunnelL.py</td>
<td>sites.dft</td>
<td></td>
</tr>
<tr>
<td>bin</td>
<td>etc</td>
<td>monitorJobR.py</td>
<td>monitorTunnelR.py</td>
<td>tools.dat</td>
<td></td>
</tr>
<tr>
<td>config</td>
<td>managers.dat</td>
<td>monitorJobS.py</td>
<td>monitorTunnelT.py</td>
<td>tools.dft</td>
<td></td>
</tr>
<tr>
<td>distributor</td>
<td>managers.dft</td>
<td>monitorJobT.py</td>
<td>monitors.dat</td>
<td>tunnels.dat</td>
<td></td>
</tr>
<tr>
<td>distributor.py</td>
<td>monitorJob.dump</td>
<td>monitorTunnel.py</td>
<td>monitors.dft</td>
<td>tunnels.dft</td>
<td></td>
</tr>
</tbody>
</table>

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).

- Domain name
- Single user account
  - Community user
  - Home directory
  - Application staging directory
  - Scratch directory
- Batch job submission classification

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, 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, 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
- **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.

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.
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.
- 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 suitable for /bin/sh execution.

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

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

**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 variables will be ignored and the values will not be passed to the remote site.

OMP_NUM_THREADS
```
Tunnels

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

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

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

```
[cluster]
venue = cluster.campus.edu
venuePort = 22
gatewayHost = gateway.campus.edu
gatewayUser = yourhub
localPortOffset = 1
```
Initialization Scripts and Log Files

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

Initialize daemon scripts

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

Log files

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

submit.log

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

Sun Aug 26 17:28:24 2012] 0: ####################################################################


Sun Aug 26 17:28:24 2012] 0: Listening: protocol='tcp', host='', port =830

system administrators

[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 12:33:38 2012] 0: Server will time out in 60 seconds.
[Sun Sep 23 12:33:38 2012] 1670: Job Status: venue=1:local status=0 cpu=0.030000 real=0.000000 wait=0.000000
[Sun Sep 23 12:33:38 2012] 1670: Job Status: venue=2:local status=0 cpu=0.040000 real=0.000000 wait=0.000000
[Sun Sep 23 12:33:38 2012] 1670: Job Status: venue=3:local status=0 cpu=7.050000 real=7.000000 wait=0.000000
[Sun Sep 23 12:33:38 2012] 1670: Job Status: venue=4:local status=0 cpu=0.080000 real=0.000000 wait=0.000000
[Sun Sep 23 12:33:38 2012] 1670: Job Status: venue=5:local status=0 cpu=0.020000 real=1.000000 wait=0.000000
[Sun Sep 23 12:33:38 2012] 1670: Job Status: venue= status=0 cpu=10.428651 real=9.561828 wait=0.000000
[Sun Sep 23 12:33:38 2012] (1154) ====================================
[Sun Aug 26 17:28:17 2012] 0: Server(10836) was terminated by a signal 2.

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/0001 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_01.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/3984L/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: pegasus-status -l /home/hubzero/user/data/sessions/3984L/00001659/01/work/pegasus

2012.09.23 00:04:23.090 EDT: *** To remove your workflow run ***

2012.09.23 00:04:23.096 EDT: pegasus-remove /home/hubzero/user/data/sessions/3984L/00001659/01/work/pegasus

2012.09.23 00:04:23.101 EDT: Time taken to execute is 0.993 seconds


2012.09.23 00:04:23.106 EDT: [Sun Sep 23 00:04:23 2012] 1659: status:Job N WF-DiaGrid

2012.09.23 00:04:38 2012] 1659: status:DAG R WF-DiaGrid

2012.09.23 00:10:42 2012] 0: quotaCommand = quota -w | tail -n 1

2012.09.23 00:10:42 2012] 1660: command = tar vchf 00001660_01_input.tar --exclude='*.svn*' -C /home/hubzero/clarksm .__local_jobid.00001660_01 noerror.sh

2012.09.23 00:10:42 2012] 1660: remoteCommand ./noerror.sh

2012.09.23 00:10:42 2012] 1660: workingDirectory /home/hubzero/clarksm

2012.09.23 00:10:42 2012] 1660: command = tar vrhf 00001660_01_input.tar --exclude='*.svn*' -C /home/hubzero/clarksm/00001660/01 00001660_01.sh

2012.09.23 00:10:42 2012] 1660: command = nice -n 19 gzip 00001660_01_input.tar

2012.09.23 00:10:42 2012] 1660: command = /opt/submit/bin/receiveinput.sh /home/hubzero/clarksm/00001660/01 /home/hubzero/clarksm/00001660/01/__timestamp_transferred.00001660_01

2012.09.23 00:10:42 2012] 1660: command = /opt/submit/bin/submitbatchjob.sh /home/hubzero/clarksm/00001660/01 ./00001660_01.condor

2012.09.23 00:10:42 2012] 1660: remoteJobId = Submitting job(s). 1 job(s) submitted to cluster 953.


2012.09.23 00:10:42 2012] 1660: status:Job N DiaGrid


2012.09.23 00:12:07 2012] 1660: Received SIGINT!

2012.09.23 00:12:07 2012] 1660: waitForBatchJobs: nCompleteRemoteJobIndexes = 0, nIncompleteJobs = 1, abortGlobal = True

2012.09.23 00:12:07 2012] 1660: command = /opt/submit/bin/killbatchjob.sh 953.0 CONDOR

2012.09.23 00:12:07 2012] 1660: Job 953.0 marked for removal
The monitorJob.log file tracks the invocation and termination of each remotely executed job monitor. The remote job monitors are started on demand when jobs 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) * distributor job monitor started *
[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
```
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 requires inclusion of a public key in the authorized_keys file.
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, it runs as a standard user. Typical configuration for the daemon looks like this:

```
$ cat monitors.dat
[cluster.PBS]
venue = cluster.campus.edu
remoteUser = yourhub
remoteMonitorCommand = $HOME/Submit/monitorPBS.py
```

The directory defined by MONITORLOGLOCATION needs to be created before the daemon is started. Sample daemon scripts used for PBS, LSF, 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.

```
$ cat sites.dat
[clusterPBS]
venue = cluster.campus.edu
remoteUser = yourhub
remoteBinDirectory = ${HOME}/bin
```

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 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 jobDescriptionFile
```

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

**Batch job script - termination**

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

killbatchjob.sh jobId 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 user ids (logins) or group ids (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 boolean keyword arbitraryExecutableAllowed 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 belong the system group submit.
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/inetorgperson.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/valsord-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/backldap.la
/usr/lib/ldap/back_sql-2.4.so.2.1.0
/usr/lib/ldap/back_sql.la
/usr/lib/ldap/dyngroup.la
/usr/lib/ldap/refint-2.4.so.2.1.0
/usr/lib/ldap/valsort.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/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/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 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/ds-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_dnsrv-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/memof-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
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 nsdc is downloaded from debian.org and installed using the following command:

```bash
# apt-get install nsdc
```

This adds the following files to the file system:

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

HUBzero LDAP Schema Installation

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

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

```bash
#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.

```bash
# 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:

```bash
# 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
```
Finally, 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
SYSTEM ADMINISTRATORS

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 < /usr/share/hubzero-mysql/hubzero-fix-debian-maint.sql
```
**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
ContainerDebianRepo=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

   ```bash
   # 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
   ```
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:

```bash
# 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**

```bash
# 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>" > /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 restarted 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**

```bash
# 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`

```bash
# 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.

```bash
# 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

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

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

```bash
# modprobe fuse
```
Then try the test again
Trac

Authentication Plugin Installation

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

```bash
# apt-get install hubzero-trac-mysqauthz
```

and Apache restarted:

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

Test

```bash
# 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.

```bash
# 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:

```bash
# 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`:

```plaintext
www-data ALL=(apps)NOPASSWD:/usr/bin/expect /www/myhub
/www/myhub/components/com_contribtool/scripts/*tool.php *

www-data ALL=(apps)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:

```bash
# 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-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-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-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 -key out xvnc.pem.key
   ```

2. The Maxwell service configuration file is created and installed in

   `/etc/hubzero/maxwell.conf:`
```plaintext
mysql_host = "localhost"
mysql_user = "myhub"
mysql_password = "<myhub user's pw>"
mysql_db = "myhub"
filexfer_decoration=""""filexfer_sitelogo { <h1><a href="http://hubzero.org/" title="HUBzero home page"><span>HUBzero.org: online simulations 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-
```
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
```
3. The sample `maxwell.conf` file is deployed:

```bash
# 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):

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

default_vnc_timeout=86400
session_suffix="L"

```

- 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 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 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/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 from 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:

```bash
# 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.