

# PVFS on Scyld Beowulf II Howto

Philip H. Carns

Jan 27, 2001

## Contents

<b>1</b>	<b>How to use this document</b>	<b>2</b>
1.1	Versions . . . . .	2
1.2	Why is this document necessary? . . . . .	2
1.3	Why are the scripts relatively incomplete? . . . . .	2
<b>2</b>	<b>Downloading and compiling PVFS</b>	<b>2</b>
2.1	Obtaining the source . . . . .	2
2.2	Untarring the packages . . . . .	3
2.3	Building the packages . . . . .	3
<b>3</b>	<b>Installing PVFS on a Scyld Beowulf II cluster</b>	<b>4</b>
3.1	Machine configuration . . . . .	4
3.2	Configuring the metadata server . . . . .	4
3.3	Configuring the I/O servers and clients . . . . .	5
3.4	Starting the PVFS daemons . . . . .	6
3.5	Client configuration . . . . .	6
3.6	Client native library interface . . . . .	7
3.7	Conclusion . . . . .	7

# 1 How to use this document

This document is intended to help system administrators setup PVFS on a cluster that has been installed using the Scyld Beowulf CD. This document assumes familiarity with both pieces of software. If you are not comfortable with this, then please have a look at the documentation that can be found at the following URL's.

## 1.1 Versions

This document coincides with the following versions of the PVFS and Scyld software:

- PVFS 1.5.1 (<http://www.parl.clemson.edu/pvfs>)
- pvfs-kernel 0.9.1 (<http://www.parl.clemson.edu/pvfs>)
- Scyld Beowulf II release 27BZ-7 (<http://www.scyld.com>)

## 1.2 Why is this document necessary?

This document isn't really necessary at all. PVFS can be installed through traditional means on a Scyld cluster, but it requires the addition of several system configuration files to the slave nodes. This introduces extra system state which undermines one of the main goals of the Scyld system image. If the steps in this documentation are followed, then only a minimum amount of modification is needed for Scyld Beowulf slave nodes to participate in PVFS.

## 1.3 Why are the scripts relatively incomplete?

As of the time of this writing, the Scyld distribution has not stabilized into a 1.0 release, so I tried to avoid using system features which may change later. Secondly, different people may have different configurations in mind. The included scripts just serve as a starting point for building what is needed to suit your own system

# 2 Downloading and compiling PVFS

## 2.1 Obtaining the source

PVFS is freely available under the GNU General Public License. There are two official download locations available on the internet:

- <ftp://ftp.parl.clemson.edu/pub/pvfs/>
- <ftp://mirror.chpc.utah.edu/pub/pvfs/> (mirror site)

Visit one of these sites and download the files `pvfs-<v1>.tgz` and `pvfs-kernel-<v2>.tgz`, where `<v1>` and `<v2>` are version numbers. Please obtain the latest version of each (note that the version numbers of these two packages do not match).

## 2.2 Untarring the packages

The two PVFS packages may be untarred into the same directory to ease compilation. The following example uses the `/usr/src/` directory:

```
[root@master /root]# cp pvfs-1.5.0.tgz pvfs-kernel-0.9.0.tgz /usr/src
[root@master /root]# cd /usr/src
[root@master /usr/src]# tar -xzf pvfs-1.5.0.tgz
[root@master /usr/src]# tar -xzf pvfs-kernel-0.9.0.tgz
[root@master /usr/src]# ln -s pvfs-1.5.0 pvfs
[root@master /usr/src]# ls -lF
total 476
lrwxrwxrwx   1 root   root           15 Dec 14 17:42 pvfs -> pvfs-1.5.0/
drwxr-xr-x  12 root   root          512 Dec 14 10:11 pvfs-1.5.0/
-rw-r--r--   1 root   root      371535 Dec 14 17:41 pvfs-1.5.0.tgz
drwxr-xr-x   6 root   root        1024 Dec 14 10:10 pvfs-kernel-0.9.0/
-rw-r--r--   1 root   root      105511 Dec 14 17:41 pvfs-kernel-0.9.0.tgz
```

## 2.3 Building the packages

Once the packages have been untarred, you may proceed to build and install them. The only modification to the normal compilation process is an additional command line switch to the configure script. The `--enable-scyld` option enables features that are useful in Scyld Beowulf environments. The remainder of the document assumes that the source was compiled in this manner.

```
[root@master /usr/src]# cd pvfs
[root@master /usr/src/pvfs-1.5.0]# ./configure --enable-scyld
[root@master /usr/src/pvfs-1.5.0]# make
[root@master /usr/src/pvfs-1.5.0]# make install
```

You should now have the server binaries, client libraries, include files, utilities, and man pages for PVFS installed on your system. The next step builds the kernel module and support programs needed for compatibility with existing system programs. The build process looks for kernel header files in `/usr/src/linux` by default, so make sure that this directory points to a copy of the kernel version that you wish to compile the module for.

```
[root@master /usr/src/pvfs-1.5.0]# cd ../pvfs-kernel-0.9.0
[root@master /usr/src/pvfs-kernel-0.9.0]# ./configure --with-libpvfsdir=./pvfs/lib
[root@master /usr/src/pvfs-kernel-0.9.0]# make
[root@master /usr/src/pvfs-kernel-0.9.0]# make install
```

At this point you need to copy `pvfs.o` into the correct directory for your system. This is usually `/lib/modules/<kernel-version>/misc/`. For example:

```
[root@master /usr/src/pvfs-kernel-0.9.0]# cp pvfs.o /lib/modules/2.2.17-33.beo/misc/
[root@master /usr/src/pvfs-kernel-0.9.0]# cp pvfs.o /home/
```

## 3 Installing PVFS on a Scyld Beowulf II cluster

This section documents the steps required to configure PVFS on a Scyld Beowulf cluster consisting of more than one node. It assumes that you have completed the preceding section entitled “Downloading and compiling PVFS” and that your cluster is already up and running. You should be able to see output from all of your slave nodes when you do a command like:

```
[root@master /root] bpsb -a uptime
```

If you do not see output from each of your slave nodes then something is probably not configured properly. Check with your Scyld documentation. Otherwise, you are set to go. There really aren't any other restrictions on the configuration.

### 3.1 Machine configuration

It is important to have in mind the roles that machines (a.k.a. nodes) will play in the PVFS system. There are three potential roles that a machine might play:

- metadata server
- I/O server
- client

A metadata server is a node that keeps up with metadata (such as permissions and time stamps) for the file system. An I/O server is a node that actually stores a portion of the PVFS file data. A client is a node that can read and write PVFS files. Your applications will typically be run on PVFS clients so that they can access the file system.

A machine can fill one, two, or all of these roles simultaneously. Each role requires a specific set of binaries and configuration information. There will be one metadata server for the PVFS file system. There can be many I/O servers and clients. In this section we will discuss the components and configuration files needed to fulfill each role.

We will configure our example system so that the “master” node provides metadata service. This node has an internal IP address of 192.168.0.1. There are eight slave nodes, all of which will provide I/O service. The slave nodes are numbered 0 through 7, and possess IP addresses 192.168.0.100 through 192.168.0.107. All nodes will be configured to act as clients.

### 3.2 Configuring the metadata server

On the machine that will act as your metadata server, you must create a directory that will be used to store file system metadata. You can then run a script called `mkmgrconf` that will configure this directory and provide information about the layout of your system. Take special note of the portion of the script that asks you to list each I/O node on the system.

```
[root@master /]# mkdir /pvfs-meta
[root@master /]# cd /pvfs-meta
```

```
[root@master /pvfs-meta]# /usr/local/bin/mkmgrrconf
This script will make the .iodtab and .pvfsdir files
in the metadata directory of a PVFS file system.
```

Enter the root directory:

**/pvfs-meta**

Enter the user id of directory:

**root**

Enter the group id of directory:

**root**

Enter the mode of the root directory:

**777**

Enter the hostname that will run the manager:

**master**

Searching for host...success

Enter the port number on the host for manager:

(Port number 3000 is the default)

**3000**

Enter the I/O nodes: (can use form node1, node2, ... or  
nodename#-#,#,#)

**192.168.0.100-7**

Searching for hosts...success

I/O nodes: 192.168.0.0 192.168.0.1 192.168.9.2 192.168.0.3 192.168.0.4 192.168.0.5  
192.168.0.6 192.168.0.7

Enter the port number for the iods:

(Port number 7000 is the default)

**7000**

Done!

```
[root@master /pvfs-meta]# ls -al
```

total 9

drwxr-xr-x	2	root	root	82	Dec 17 15:01	./
drwxr-xr-x	21	root	root	403	Dec 17 15:01	../
-rwxr-xr-x	1	root	root	84	Dec 17 15:01	.iodtab*
-rwxr-xr-x	1	root	root	43	Dec 17 15:01	.pvfsdir*

The .iodtab and .pvfsdir files were created by the mkmgrrconf script and contain information about the configuration of the file system.

### 3.3 Configuring the I/O servers and clients

Next you must prepare the slave nodes and head node by creating a few directories and a special device file. Note that the provided script assumes that you wish to use /mnt/pvfs as your mount point for the file system. You can modify this script as needed. It only needs to be run once unless you rebuild particular nodes or add new ones to your system.

```
[root@master /root/pvfs-scyld-howto-1.5.1]# ./pvfs_slave_create.sh a
```

The "a" argument specifies that you want to configure all of the currently running nodes as well as the head node. Alternatively, you can just setup specific nodes by passing the node number as an argument, or just setup the local head node by using the "l" argument.

This step only needs to be performed once if your slave nodes are using the local hard drive and do not reformat on reboot. However, if you are booting from read only media, you must perform this step every time the machine is rebooted.

### 3.4 Starting the PVFS daemons

At this point, everything is configured and all that is left is to start the file system daemons and mount the file system locally on each node. To start up the PVFS daemons on every node you can run the following script (which again should be modified if needed):

```
[root@master /root/pvfs-scyld-howto-1.5.1]# ./pvfs_servers_start.sh
```

This script assumes that you are running an I/O server on every node, which of course may not have been your desired configuration. Please modify the script if you wish to only start I/O servers on particular slave nodes. This script needs to be any time the cluster is rebooted.

The server side portion of PVFS is now ready for access. If at any point you wish to verify that one of the iod's or mgr is running correctly on your system you may check it by running the iod-ping or mgr-ping utilities, respectively. These test programs establish communications with the daemons in question and return a status report:

```
[root@master /root/pvfs-scyld-howto-1.5.1]# /usr/local/bin/iod-ping -h 192.168.0.100
192.168.0.100:7000 is responding.
[root@master /root/pvfs-scyld-howto-1.5.1]# /usr/local/bin/iod-ping -h 192.168.0.101
192.168.0.101:7000 is responding.
[root@master /root/pvfs-scyld-howto-1.5.1]# /usr/local/bin/mgr-ping -h master
master:3000 is responding.
```

### 3.5 Client configuration

There are two primary methods for accessing a PVFS file system. The first is the native PVFS interface which is made available through `libminipvfs`. This library offers the most high performance method of accessing the file system. It is therefore also the interface used by ROMIO if you configure your system to use MPI-IO. The second method relies on a kernel module to provide standard Linux file system compatability. This interface allows the user to use existing binaries and system utilities on PVFS without recompiling, although at somewhat of a performance penalty over the native library.

First we will cover the kernel module interface, and then cover the steps needed to use the native pvfs interface.

In order to mount the PVFS file system with the kernel interface, you must load the `pvfs.o` module, start the `pvfsd` daemon, and then mount the file system using the `mount.pvfs` command. An example of how to do this can be found in the `pvfs_client_start.sh` script. This script will cause all of the nodes to mount the file system on `/mnt/pvfs`. Notice that you must use an IP address rather than a hostname to specify the manager node.

```
[root@master /root/pvfs-scyld-howto-1.5.1]# ./pvfs_client_start.sh 192.168.0.1:/pvfs_meta
```

NOTE: On Scyld Beowulf II preview releases prior to 27BZ-7, this script may not work. It depends on a new feature that allows modprobe to pull modules from the head node of your cluster. If you do not have this feature, then you must copy the kernel module out to your nodes or make it available over NFS so that your slaves may use it.

At this point, you should be able to use `/mnt/pvfs` just as you would any other file system. The following shows example output from the `mount` command:

```
[root@master /root/pvfs-scyld-howto-1.5.1]# mount
/dev/hda5 on / type ext2 (rw)
none on /proc type proc (rw)
/dev/hda1 on /boot type ext2 (rw)
none on /dev/pts type devpts (rw,gid=5,mode=620)
192.168.0.1:/pvfs_meta on /mnt/pvfs type pvfs (rw)
```

When you wish to unmount the file system and remove the modules (this is probably a good idea before rebooting), you may run the following example script:

```
[root@master /root/pvfs-scyld-howto-1.5.1]# ./pvfs_client_stop.sh
```

### 3.6 Client native library interface

If you wish to use the native PVFS library functions, or if you wish to use the ROMIO MPI-IO implementation, you must perform one further step. Each of the nodes (including slave nodes) must have a file in the `/etc` directory called `pvfstab`. This file informs the pvfs library where to look for the file system. In our case, for example, it would contain:

```
192.168.0.1:/pvfs_meta /mnt/pvfs pvfs port=3000 0 0
```

This file should be copied out to each node that will be participating using either the `bpcp` command or some other means. Remember again to specify the manager node with an IP address rather than hostname.

### 3.7 Conclusion

Congratulations! You now have a fully functional PVFS configuration on your Scyld Beowulf cluster. You should now be able to write parallel applications that generate concurrent accesses to the PVFS file system mounted on `/mnt/pvfs`.