ParNFS

Version 1.2

Olaf 'Rhialto' Seibert.

This document describes ParNFS 1.2.

Copyright © 1993-1995 by Olaf 'Rhialto' Seibert.

Permission is granted to copy the whole package, in accordance with the GNU General Public License and some additional restrictions.

1 Overview

ParNFS is intended to be a replacement for the well-known Parnet file system by the Software Distillery. I wrote it because it had a number of important deficiencies:

- It came without source, which is a sin bi itself, but also meant I couldn't fix the other problems.
- It would not allow you to get a directory listing with most versions of the "list" command that I used; only the AmigaDOS 1.3 version worked.
- It did not cooperate with other applications using Parnet.
- It gave annoying requesters when you attempted to use 2.0 features.

2 Requirements

Besides what is included in this archive, you need

- Two (or possibly more) Amigas.
- A parnet cable. How to make one is described in Section “Cable.doc” in `Cable.doc`.

3 Installation

There are two different kinds of installation: the easy one and the not-so-easy one.

3.1 Simple Installation

You can do the simple installation if you don't have any other application yet that uses the parnet.device. If one of the following things is not appropriate for your case, you need to do the Section 3.2 [Complex Installation], page 6.

- You only have two Amigas involved as client and server (sometimes called "host") for each other.
- The parnet addresses are set to 1 for the primary server and 2 for the primary client.
- Parnet port number 604 and 605 are not yet in use.

In the simple case, we assume there will be a symmetric situation: both Amigas will be both client and server, so you can access files on Amiga 1 from Amiga 2 and files on Amiga 2 on Amiga 1. However, both client-server relations are distinct from the software's point of view, and therefore I will call one the primary client-server relation, and the other the secondary one.

At this point, you should grab your trusty `Installer` program, and run `Install-ParNFS`. Or you can install ParNFS by hand, and do the following.

First, you need to copy a few files to the appropriate place. Do this on both Amigas.

- Copy `NetworkFileSystem` to `L:`.
- Copy `NetworkFileServer` to `C:`.
- Append `Section ''Mountlist'' in Mountlist` to `Section ''DEVS:Mountlist'' in DEVS:Mountlist`, or copy it to `Section ''DEVS:DosDrivers/NET'' in DEVS:DosDrivers/NET` and delete the lines containing "NET:" and "#".

3.1.1 The primary client and server

Now you can bring up the primary server and client. All that is needed is

- `Run NetworkFileServer` on the server (Amiga 1).
- `Mount NET:` on the client (Amiga 2).

Poof! You're done.

3.1.2 The secondary client and server

This is only slightly more difficult. What you do is

- Edit the Mountlist on the secondary client (Amiga 1), so that the line with BlocksPerTrack reads

  BlocksPerTrack = 1.
- `Run NetworkFileServer -s` on the server (Amiga 2). The `-s` means "secondary".
- `Mount NET:` on the client.

Poof! You're done again.

If you want to use the workbench over your network, drop some copies of the `Node.rinfo` icon on the disks you wish to see icons for.

3.2 Complex Installation

First you have to read through the instructions for the Section 3.1 [Simple Installation], page 5.

Now you have to realise that any references to `NetworkFileServer` only apply to an Amiga in its capacity as server (or host), and `NetworkFileHandler` and `Mountlist` to a client Amiga.

In all applications of parnet and ParNFS, you must follow these two simple rules:

- Every Amiga must have its own parnet address.
- Every client communicates with its server using a port number unique to that server.

So what you only need to do is select addresses for all Amigas (but you probably did that before), and select a unique port number for each server. Then you install each client and each server analogous to the simple procedure, with the following modifications:

- For a client, you can indicate the selected numbers in the Mountlist as follows:

  ```
  BlocksPerTrack = 2
  Interleave = 0xppppsscc, where
  pppp: the port number the server is using (in hex)
  ss  : the parnet address of the server (in hex, 0x01 .. 0xFE)
  cc  : the parnet address of this client (in hex, 0x01 .. 0xFE)
  ```

  So, to do the simple installation in the complex way, you'd have

  ```
  BlocksPerTrack = 2
  Interleave = 0x025C0102
  ```

- For a server, you use the command

  `Run NetworkFileServer -a <this-address> -c <client-address> -p <port>`

  with, of course, the selected addresses and port number.

You will notice the redundant specification of the Amiga's address in case it acts both as a client and a server. This is because parnet does not have a way to inquire the current address, but all programs must know it.

3.3 Advanced Installation

It should be possible to have more than one client using the same server. In this case they still use all the same port number. This should be safe up to about 4 clients per server[1].

You can also run multiple servers on the same Amiga; they need of course their own port numbers or chaos will ensue.

You can of course also run several client `NET:`s on a single Amiga: just use a different name for each. But they can't talk to the same server, because that gives a port number clash on the client machine.

[1] To increase this, recompile the server code with a higher value of the #define PENDREADS.

4 Usage

Once you've installed ParNFS, you can use it almost 100% the same as the original Software Distillery Parnet system.

This means that you can access, for instance, `DH0:something` on the server (or host) as `NET:DH0/something`. This also works for assigns and character devices (such as `SER:` and `CON:`).

The `NET:` device also creates a volume node, named `Network:`. If any requesters appear, they will always refer to volume `Network:`; unfortunately they are not more specific than that. Especially "Please insert volume Network:" is a bit silly.

ParNFS remembers remote device, volume and assign names that you have successfully used, and if you get a directory listing of `NET:` you get all those names listed as subdirectories. There are also `.info` files suppled for the Workbench. The corresponding icons are actually in the files `NET:dev/Node.rinfo`, so unless these exist, Workbench will not actually display an icon. Please note that, just like Parnet, these `.info` files always appear, even if the corresponding `Node.rinfo` file does not exist.

There are a few non-obvious things. You cannot do any device-tied operations, because there is only one device with some subdirectories. (See Chapter 5 [Limitations], page 9, for which packets those are).

You can, however, relabel a disk by renaming its directory.

```
Rename NET:df0 NET:test
```

A side-effect is a "Please insert..." requester, when the Rename command is trying to find out if you a directory `NET:test` already exists and it should rename `NET:df0` to `NET:test/df0` instead.

You can safely abort and restart the server. However, access to files and locks obtained before is no longer possible and will be detected by the client which will give "Volume not mounted" errors.

You can also stop and restart the client (by sending an ACTION_DIE packet), but it will only quit voluntarily if there are no open files or locks. In case of involuntary restart (i.e., a reboot of the client Amiga), files and locks remain open at the server and will be lost until it also reboots. An automatic recovery from this situation is currently not implemented (but certainly possible).

5 Limitations

- It is desirable that everyone should upgrade from 1.3 to 2.0 (or better) as fast as possible. Therefore, the `NetworkFileServer` will gladly call 2.0 functions, if any client requests so. If the server runs on 1.3 in this case, **this means the server will crash!!!** I will **not** be fixing this; the correct way to remedy this is to upgrade.[1]
- Having said this, ParNFS supports all 2.0 packets, with the exception of the two file notification packets, ACTION_EXAMINE_ALL, device-related operations AddBuffers(), ACTION_FLUSH, Inhibit(), ACTION_WRITE_PROTECT (the Lock command) and Format.

  So record locks are supported over the net, and so is ExAll(), because AmigaDOS will substitute it with ExNext() if needed.
- File and directory names are not completely parsed by ParNFS. A directory name such as `/DH1` if you're in `NET:DH0` will not work, even though a `ParentDir()` from `NET:DH0` will give you `NET:`.
- There is a potential problem with `Examine()` and `ExNext()`. A careful reading of ExNext() and ACTION_EXAMINE_NEXT learns that for each directory scan using a series of ExNext() calls, the same FileInfoBlock should be used, and that you shouldn't copy FileInfoBlocks around. It is also implied (but not said so with so many words) that each scan should use its own FileInfoBlock (with its own unique address).

  Unfortunately, these restrictions are impossible for the server to adhere to. One could say that it would be possible to allocate a fresh FileInfoBlock for each call to Examine(), but when can you release it? There is no way to know if it is going to be used for ExNext() at some later time, or if the caller will call ExNext() again once it has done so a few times. Had there existed ExamineFirst() and ExamineEnd() calls, this would be possible.

  Fortunately, this seems to work for now, but it may break in the future. I don't know what contortions I would have to go through to make it work again in such a case...
- A similar situation exists with ExAll() and ExAllEnd(), only worse. ExAllEnd() only exists on 3.0 and above, so you can't count on it being used. And the callback hook for ExAll() would be a nightmare to implement.

[1] Be glad I didn't use any 2.0 only functions at other times; this would have been a lot easier for me!

6 Implementation

The client communicates with the server with datagrams, assuming reliable delivery. For parnet, this is a reasonable assumption.

In general, each DOS packet sent to the client results in a datagram sent to the server. In general the contents of the datagram are very similar to that of the DOS packet, except that file names and other data that does not fit in a longword must be copied entirely. A datagram is received in response, and its result values are put to the DOS packet. For received file handles and locks, a DOS compliant structure is built and the value from the server is hidden in there. (When a lock or file handle must be sent to the server, these values are retrieved and sent instead.)

There are a few exceptions to this, however, which make the implementation a bit more complicated. For the ACTION_LOCK_RECORD, no reply is sent; because of the timeout that is required it must be handled asynchronously. A reply can be sent at any time, possibly interrupting the processing of some other operation.

At any time, a RESET packet can be received; this means that the other side has restarted. A RESET-REPLY is sent in reply. The client uses these packets to detect that existing files and locks are now invalid. The server currently ignores this information as it doesn't track resources.

7 Glossary

Client
: A client program is a program that uses the services of a server. A client Amiga is the Amiga that the client program runs on. There may be multiple client programs per Amiga. Usually the Local Amiga.

Server
: A server program is a program that provides services to one or more client programs. A server Amiga is the Amiga that the server program runs on. There may be multiple server programs per Amiga. Usually the Remote Amiga.

Host
: Synonym for Server Amiga.

Local Amiga
: The Amiga in front of you. Usually the client Amiga.

Remote Amiga
: The Amiga not in front of you, but connected through the parnet cable. Usually the server Amiga.

8 License

ParNFS is Copyright © 1993-1995 by Olaf Seibert. All rights reserved.

This program is free software; you can redistribute it and/or modify it under the terms of the Section "GNU General Public License" in `COPYING`, as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

The Licence (see Section "the file COPYING" in `COPYING`) does not specify limitations on copying fees; the specification shall be that ParNFS may not be distributed in a commercial package of any kind, including floppy disks sold by pd/freeware/shareware resellers charging more than $6 or DM 10 per disk, without my written permission. For CD-ROMs the maximum copying fee shall be the price of a Fred Fish CD-ROM.

9 Release Notes

1.2 fixes a few small bugs and adapts to DICEs ever stricter type checking.

Index

Contents

Table of Contents