Minimal Implementation of pNFS like filesystem

by

Ravikumar Manjunath
Sanjeev Bagewadi

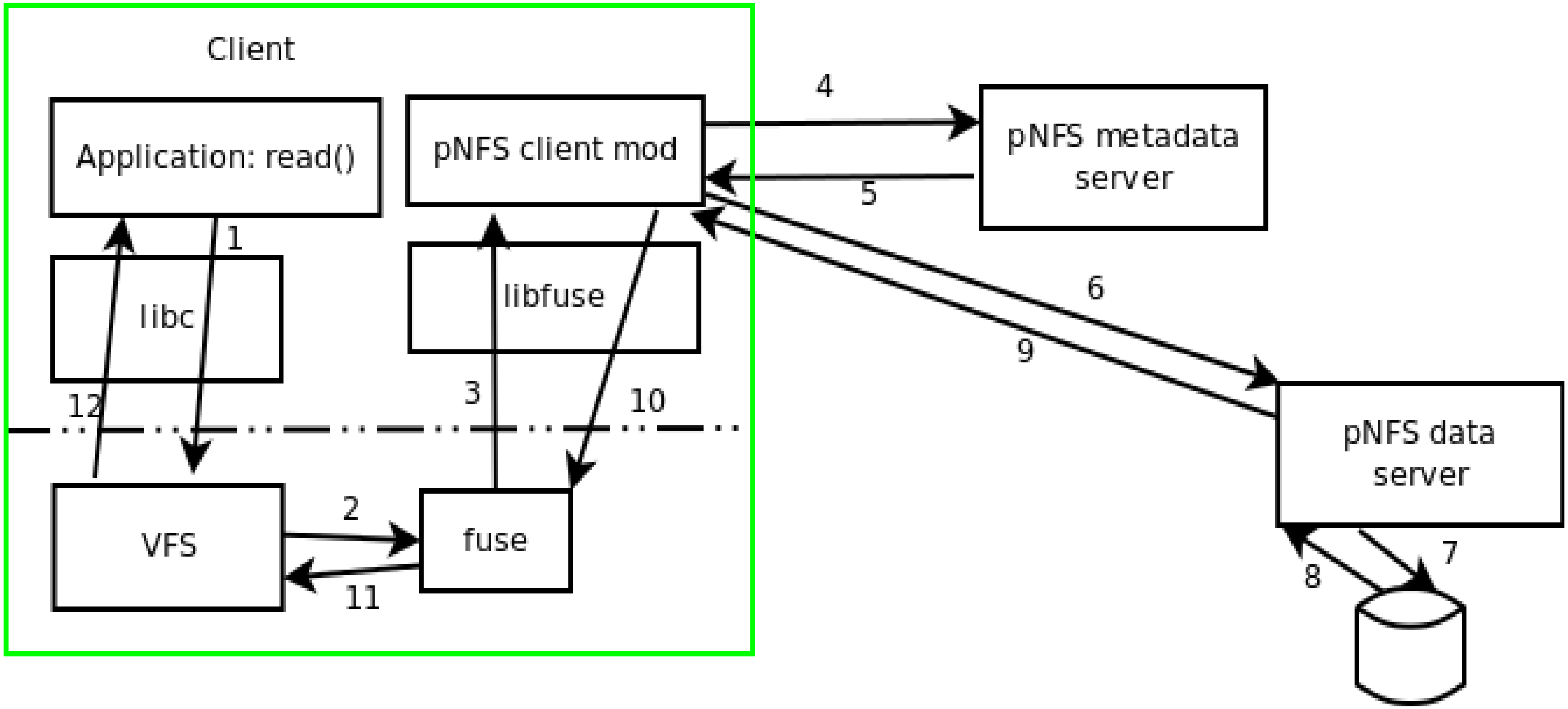
Agenda

- Problem Statement
- Basic Design
- File Layout
- Command line info
- Results
- Conclusion

Problem Statement

- The scalibility of a network filesystems is limited if we have single server.
- One of the solutions is to increase the bandwidth by spreading the load across multiple servers.
- And along with the above solution we should still have unified namespace.
- The idea is to implement a pNFS like filesystem and achieve data striping.

Basic Design



Control flow for a read operation in the proposed pNFS setup

Main Components

- FUSE Client on the client machines
  - This is responsible for contacting the metadata and data servers for the filesystem operations
  - The client machines would need fuse installed.
- MetaData Server
  - Manages the metadata of the FS
- Data Server
  - Responsible for reads/writes of the data extents

How is file metadata maintained ?

- The metadata server maintains the directory structure and all the metadata for the files/directories
- File Layouts - for each file it maintains the extent map in the file

```
<filesize>
<offset> <length> <dataserver-name> <extent-name>
<offset> <length> <dataserver-name> <extent-name>
```

```
# cat mds_share/file1
24144
0      16384  dshost1   file1.ext0
16384  16384  dshost2   file1.ext1
```

Steps of a read/write operation

- The fuse kernel modules calls into pnfs_client
- pnfs_client calls getlayout() to mds_server to fetch the layout for the given offset and length
- getlayout() searches for the <off,len> in the layout file to find the matching extent
- Returns the layout if found
- In case of write if an extent is not found allocates one, updates size and returns the extent to pnfs_client
- pnfs_client now contacts the ds_server for the actual read/write of the data.

Command Line Interface

- How to start the metadata server ?

```
$ ./mds_server -d <share-dir> -f <dslist>
```

- How to start dataservers ?

```
$ ./ds_server -S <mds-server-name> -d <share-dir>
                 -i fsid
```

  - *NOTE : Today we ignore the fsid. But, can be later used to identify the exact filesystem when multiple filesystems are shared.*

- How to mount on the client ?

```
$ ./pnfs_client -S <mds-server-name>
       -f <fs-name-shared-by-mds-server>
       <local-mnt-point>
```

Evaluation and Results

- We tested the basic file operations (create, write, read, unlink etc).
- We tested the striping of large files and its scalability

File Size : 1.4 Mbytes		
No. of Data Servers	Read time in secs	Write time in secs
1	0.417	4.457
2	0.409	3.803
3	0.397	3.119

Future Work

- Implement sharing of multiple filesystems
- Dynamic addtion of data servers.
- Page caching for better response times
- Implementation of the missing filesystem operations

Conclusion

- This is a basic pNFS like filesystem which stripes the data effectively
- Demonstrates that the speed and capacity can be made scalable with this approach
- Not all filesystem operations work. But it is easily possible to extend/implement them.