NFS Ganesha

An open source,
user mode,
multi-protocol,
multi-filesystem backend
file server

History

- CEA internal project – 2005
- LGPL licensing – April 2010
- IBM contributions – March 2010
- Panasas contributions – May 2011

Release Status

- V1.5 - Summer 2012
  - base for IBM GPFS product
  - Limited support (what IBM is interested in)
  - Stability and correctness changes
  - Legacy server
- V2.0 – December 2013
  - Extensive rewrite of major components.
    - FSAL API and all FSAL drivers
    - Cache Inode layer
    - Combined NFSv4.0 and 4.1 ops
    - Real pNFS (file and object support)
  - First Linux distro packaging (Fedora 21)

V2.0 Change Stats

- 2386 Changesets from 35 developers
- 15 Employers represented
- 402018 Lines added
- 840989 lines remove (delta -438971)

Developer Commits V2.0

Developer	Commits	% of total
Jim Lieb	515	21.6
Adam Emerson	376	15.8
Philippe Deniel	348	14.6
Matt Benjamin	248	10.4
Frank Filz	228	9.6
Max Matveev	87	3.6
Sachin Bhamare	81	3.4
Marc Eshel	65	2.7
Amit Dror	60	2.5
Dominique M	58	2.4
Paul Sheer	49	2.1
Thomas Favre-B		
Boaz Harrosh		
JV		
20 devs		

Developer Line Changes V2.0

Developer	# Lines	% of total
Adam Emerson	+474961/-376343	49.7/44.8
Jim Lieb	+160758/-10188	16.8/1.2
Philippe Deniel	+60691/-31890	6.4/-3.8
Frank Filz	+35476/-2054	3.7/0.2
Paul Sheer	+28442/-9585	3.0
Max Matveev	+27333/-15439	2.9/1.8
Matt Benjamin	+19227/-3944	2.0/0.5
Boaz Harrosh	+13136/-11292	1.4/1.3
Dominique M.	+12215/6232	1.3/0.7
Allison H.	11002	1.2
Marc Eshel	10474	1.1
Sachin B.	3868	0.4
Anand S.	3116	0.3
Amit Dror	+2185/-759	0.2/0.1
20 devs	+6976/-116	0.3/0.0

Contributions by Employer V2.0

Employer	# Hackers	# Commits	% of total	# Lines	% of total
Linuxbox	3	626	26.2	516160	54
Panasas	4	641	26.9	209072	21.9
CEA	4	399	16.7	86330	9.0
IBM	15	332	13.9	49915	5.2
Frank Filz	1	95	4.0	22332	2.3
Red Hat	3	8	0.3	3116	0.4
Infinidat	2	70	0.2	2783	0.2
Misc	5	173	6.1	63689	6.7

Developer Changesets V2.1

Developer	# Changesets	% of total
Frank Filz	84	34.1
Jim Lieb	79	32.1
Dominque Martinet	24	9.8
Adam Emerson	19	7.7
Philippe Deniel	14	5.7
Marc Eshel	12	4.9
Matt Benjamin	3	1.2
Malahal Naineni	2	0.8
Rong Zeng	2	0.8
Thomas Favre-Bulle	2	0.8
Gregoire Pichon	2	0.8
Jeremy Bongio	1	0.4
Suhrud	1	0.4
Srikrishan Malik	1	0.4

Developer Changed Lines V2.1

Developer	# changed lines	% of total
Jim Lieb	+9296/-2318	34.4/16.3
Frank Filz	6471	23.9
Marc Eshel	2380	8.8
Philippe Deniel	1843	6.8
Adam Emerson	+242/-39	0.9/0.3
Dominque Martinet	170	0.6
Matt Benjamin	142	0.5
Gregoire Pichon	45	0.2
Suhrud	19	0.1
Malahal Naineni	6	0.0
Rong Zeng	3	0.0
Thomas Favre-Bulle	2	0.0
Jeremy Bongio	+1/-1	0.0/0.0
Srikrishan Malik	1	0.0

Future Roadmap

- Shorten release cycle to 3-4 months
  - V2.0 has been an 18 month effort...
- V2.1 – March 2014
  - Performance (pNFS especially)
  - Better Pseudo filesystem
    - Dynamic add/remove exports
  - Configuration file processing rewrite
    - Better parser
    - Dbus enabled
  - Apply polish to 9P and pNFS
- V2.2 – Spring/Summer 2014
  - Delegations?
  - Clustering?

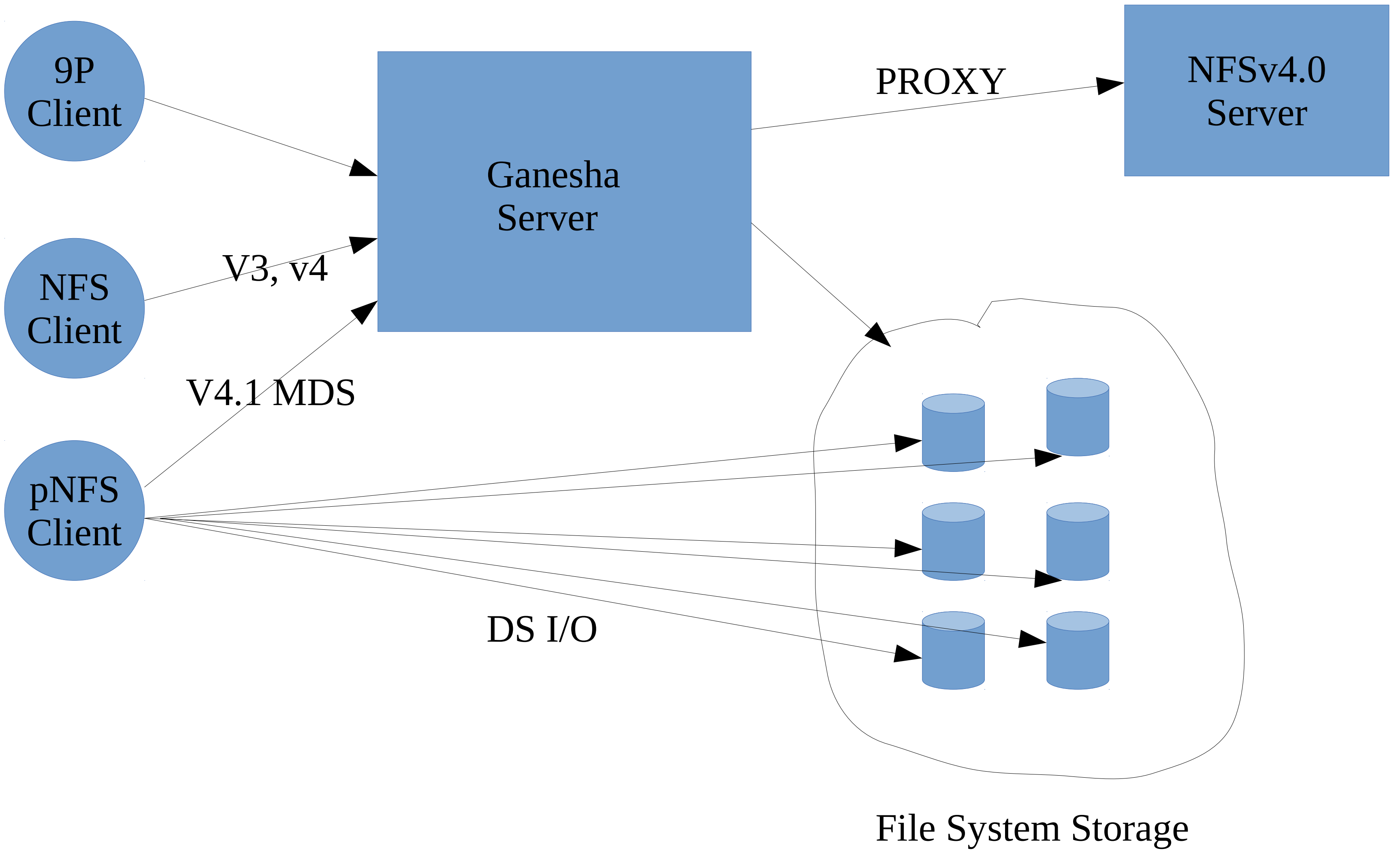
Feature Set

- Protocols:
  - NFSv3 with NLMv4, MNTv2,3, RQUOTA
  - NFSv4.0,NFSv4.1 (pNFS)
  - 9P (Plan 9 remote file protocol)
- File System Abstraction Layer (FSAL)
  - Simple (VFS, ZFS, LUSTRE, GPFS, CEPH, GLUSTER, PT)
  - pNFS (VFS, GPFS, CEPH)
  - Proxy (PROXY for NFSv4)
- Server management via Dbus
  - Python and PyQt4 based clients (reference implementations)

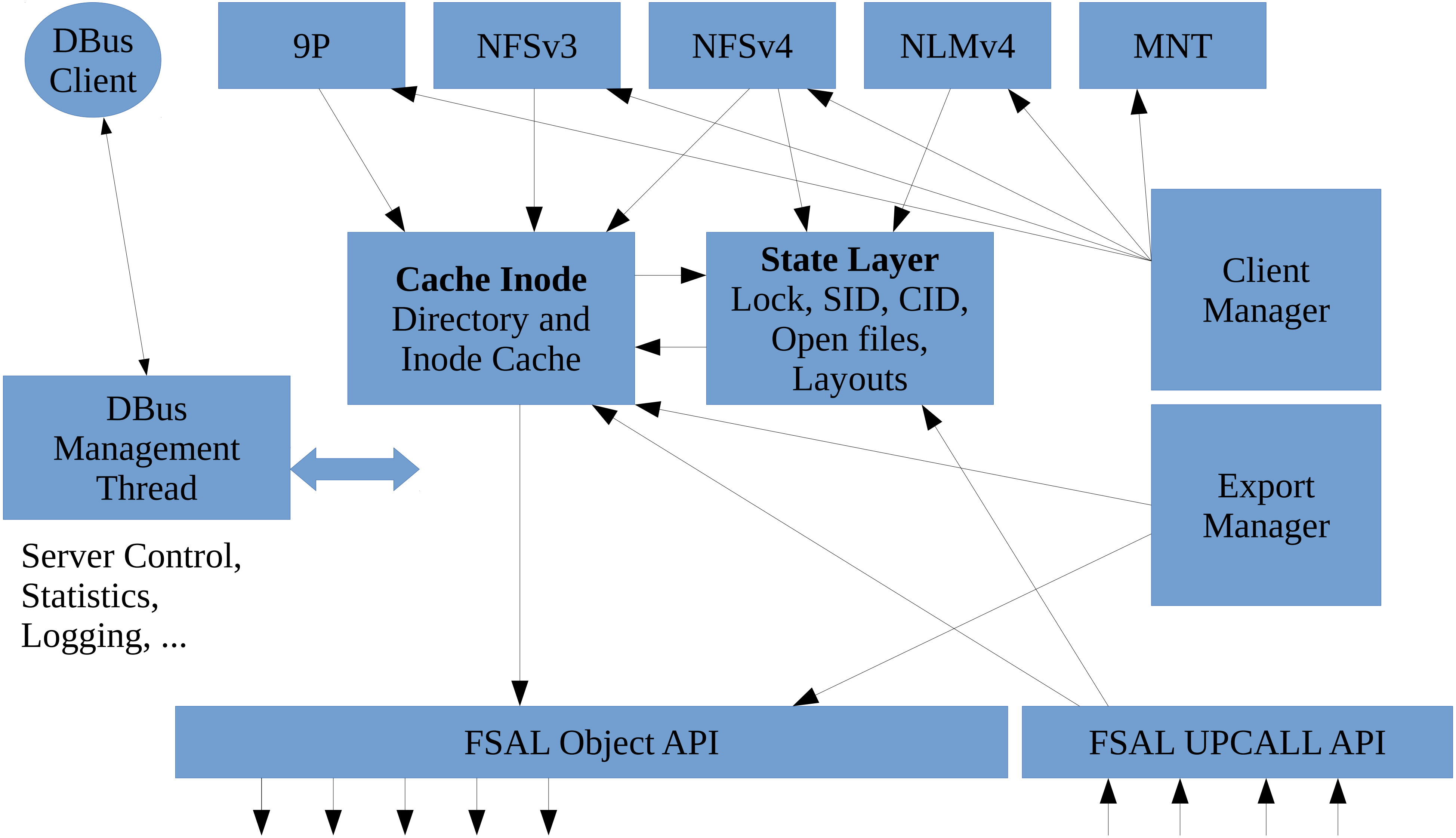
Under the hood...

- Protocol translations via PROXY FSAL
- Extensive Meta-data caching
  - AVL tree of directory contents
  - File handle keyed cache of inodes (state and attributes)
  - Open file descriptors for v3 acceleration and sane(r) POSIX locks...
- FSAL layering/stacking
  - File content cache for slow (tape drive) backends
  - File system extensions
- State management
  - Session and client ID generation and management
  - Lock state management (combined v3 and v4.x)
  - Cluster aware state management (future)
- FSAL upcalls to synchronize cache with underlying filesystem (directory morphing, layout recalls)

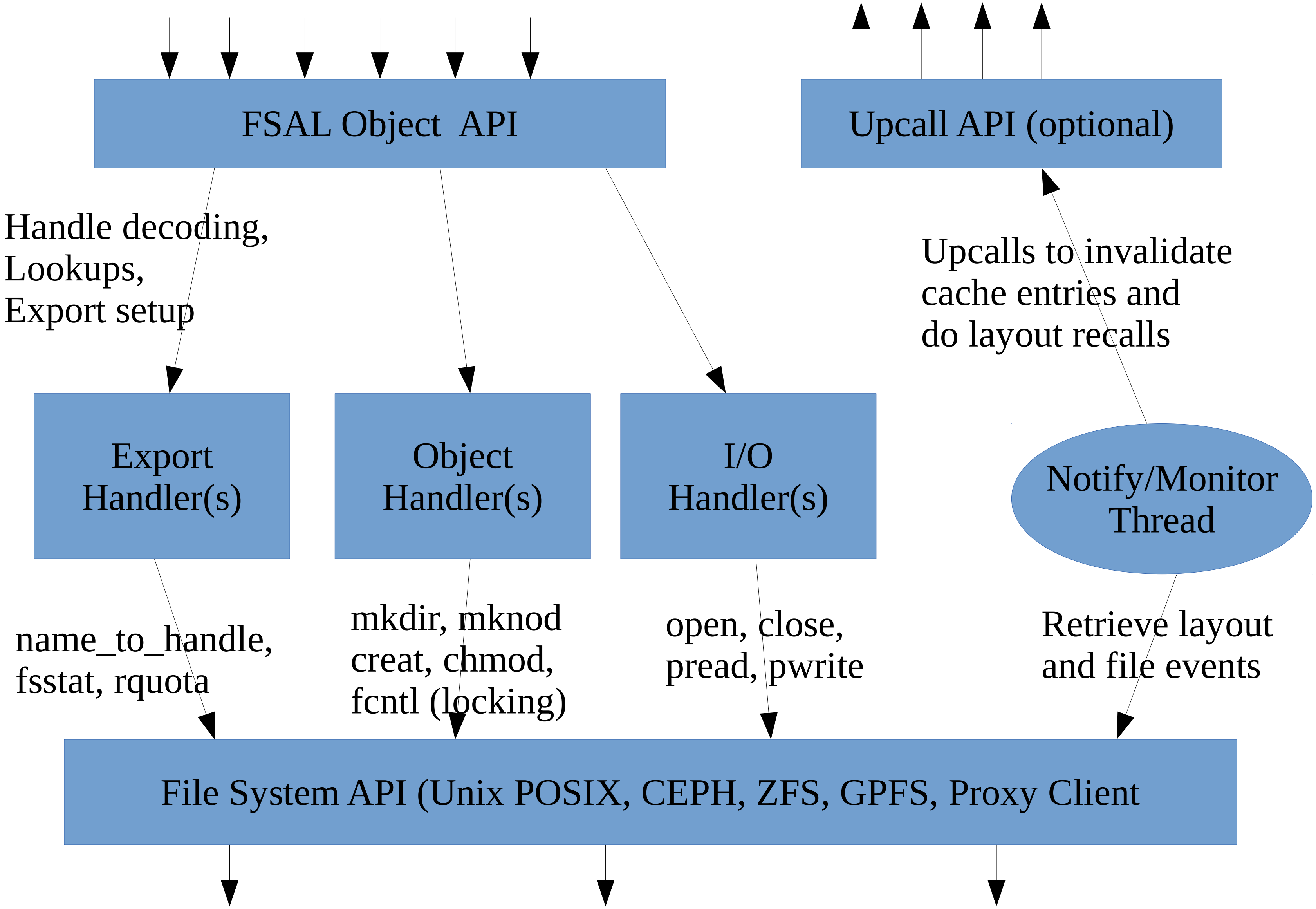
Ganesha at Work



Core Architecture



FSAL Architecture



FSAL Object API

Upcall API (optional)

Handle decoding,
Lookups,
Export setup

Upcalls to invalidate
cache entries and
do layout recalls

Export Handler(s)

Object Handler(s)

I/O Handler(s)

Notify/Monitor Thread

name_to_handle,
fsstat, rquota

mkdir, mknod
creat, chmod,
fcntl (locking)

open, close,
pread, pwrite

Retrieve layout
and file events

File System API (Unix POSIX, CEPH, ZFS, GPFS, Proxy Client

Implementation Notes

- ANSI C (C99), Linux kernel coding style
- Heavily multi-threaded (pthreads)
  - Thread pools
  - Separate decode and protocol op threads
  - Dbus control thread
  - Asynch upcalls
  - FSAL filesystem event handlers
  - GC/LRU reapers
- Standard public library interfaces to filesystems
- New and improved TIRPC library
  - Full duplex RPC
  - Fully thread safe
  - GSS support
- Portability layer (Linux, FreeBSD 8+, ??)

Questions?

- Project wiki:

  http://nfs-ganesha.github.com
- Public source repository:

  git://github.com/nfs-ganesha/nfs-ganesha.git
- Volunteers/contributors are welcome