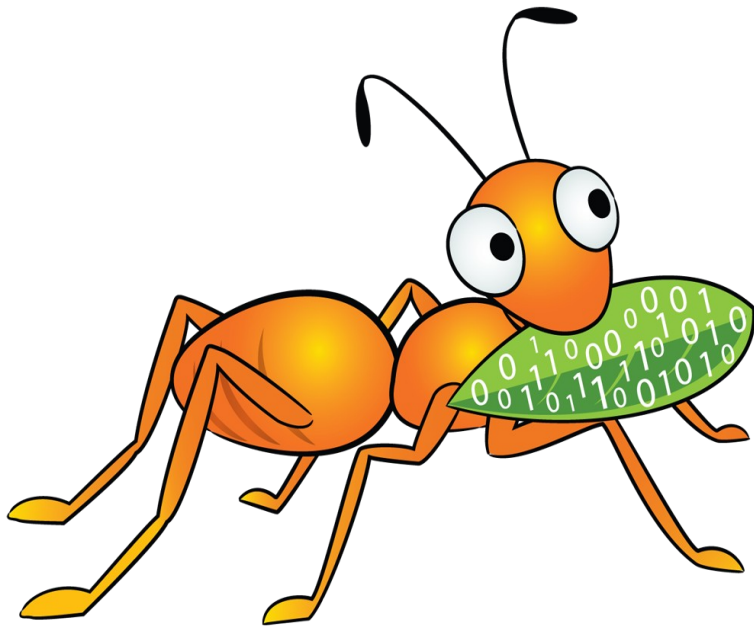


Open Source, Scale-out clustered NAS using nfs-ganesha and GlusterFS



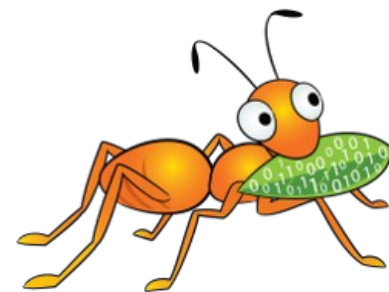
Anand Subramanian

**Senior Principal Engineer,
Red Hat**

anands@redhat.com

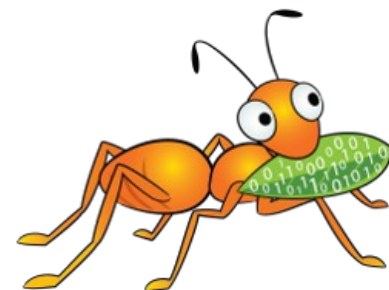
Agenda

- **Introduction**
 - **GlusterFS**
 - **NFSv4**
 - **nfs-ganesha**
- **Nfs-ganesha Architecture**
- **Integration of nfs-ganesha with GlusterFS**
 - **Low-cost scalable clustered NAS solution**
- **Performance**
- **Future Directions**



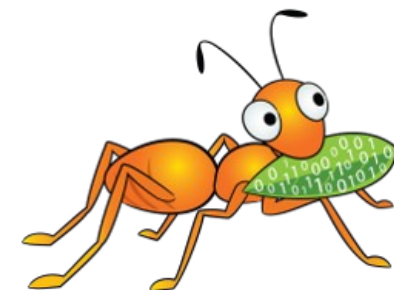
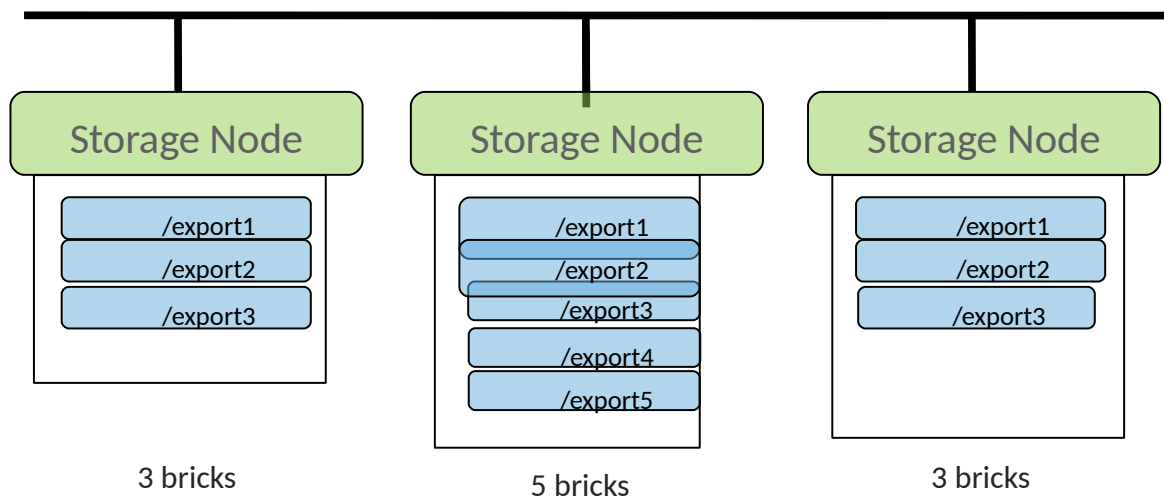
What is GlusterFS?

- A general purpose scale-out distributed file system.
- Aggregates storage exports over network interconnect to provide a single unified namespace.
- Filesystem is stackable and completely in userspace.
- Layered on disk file systems that support extended attributes.

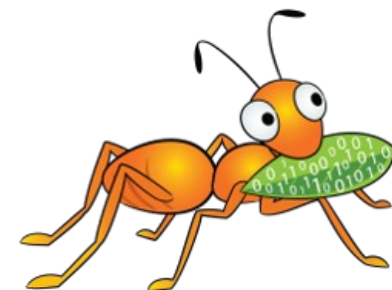
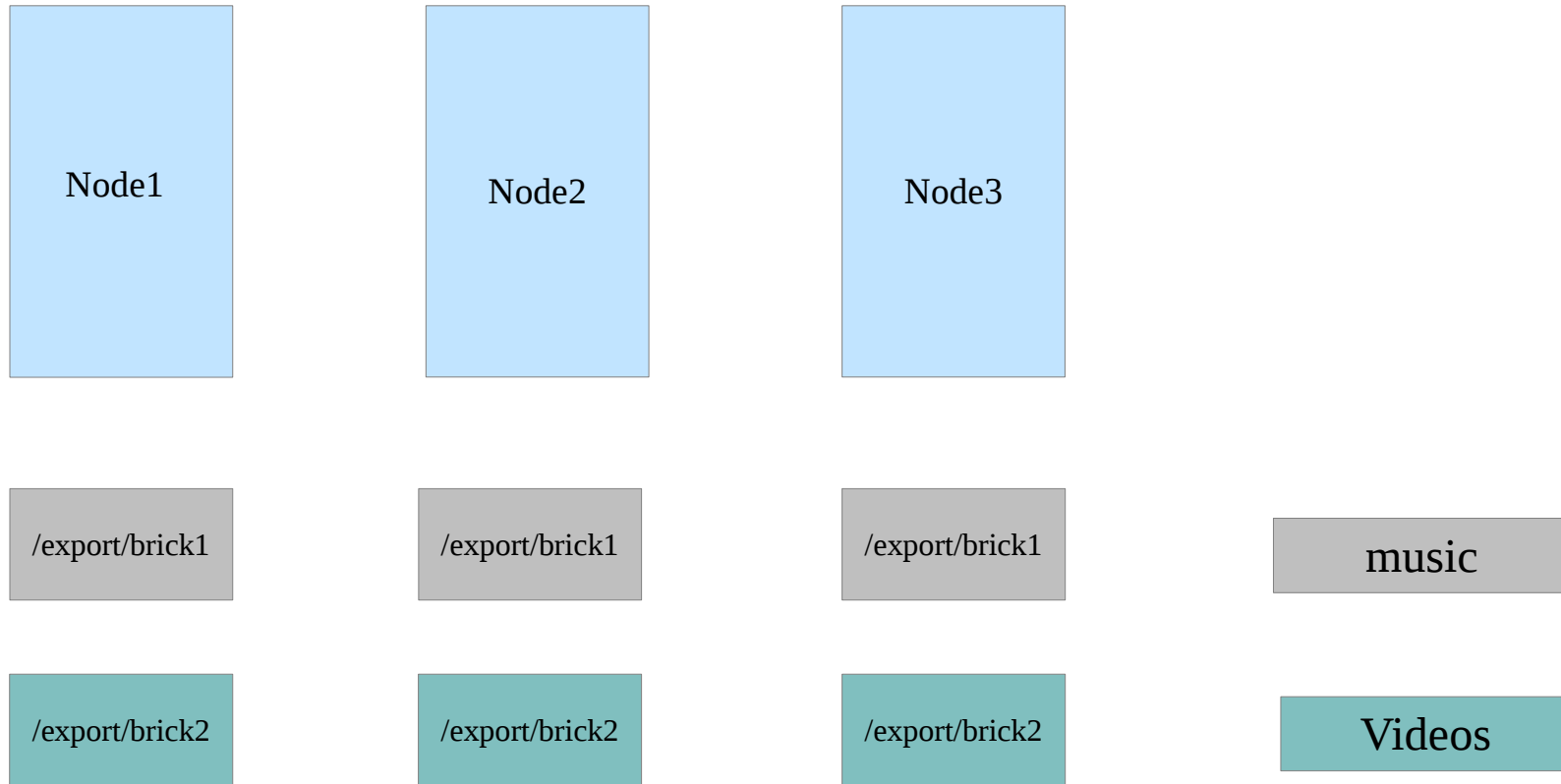


GlusterFS concepts - Bricks

- A brick is the combination of a node and an export directory – for e.g. hostname:/dir
- Each brick inherits limits of the underlying filesystem
- Ideally, each brick in a cluster should be of the same size



GlusterFS concepts - Volumes



Access Mechanisms

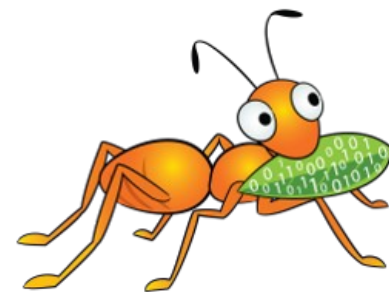
Gluster volumes can be accessed via the following mechanisms:

- FUSE based Native protocol
- NFS - V3, V4, V4.1/pNFS, V4.2
- SMB
- libgfapi
- ReST/HTTP
- HDFS



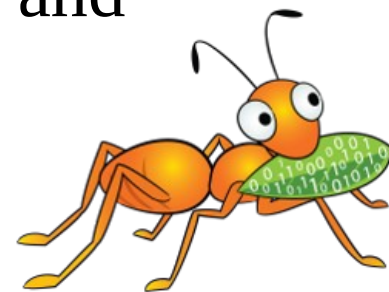
libgfapi

- Exposes APIs for accessing Gluster volumes.
- Reduces context switches.
- qemu, samba integrated with libgfapi.
- Both sync and async interfaces available.
- Emerging bindings for various languages.



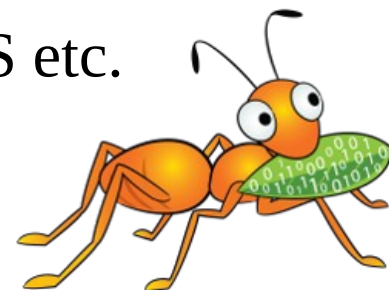
GlusterFS and NFS

- Gluster had its own NFS v3 server for access to the filesystem/storage volumes
- NFSv4 as a protocol has developed
 - Been ironed out for almost a decade
 - Slow at first but increased interest and adoption across the industry
 - Firewall friendly, UNIX and Windows friendliness, better security
 - Paves the way for the interesting stuff in 4.1/4.2 and beyond



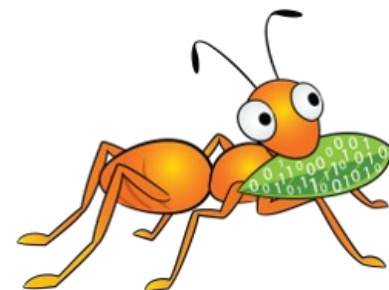
nfs-ganesha

- User-space NFS server; developed by Daniel Philippe in FR
- Supports V2, V3, V4, v4.1/pNFS, v4.2
- Can manage huge meta-data and data caches
- Provision to exploit FS specific features
- Can serve multiple types of File Systems at the same time
- Can serve multiple protocols at the same time
- Can act as a Proxy server
- Cluster Manager agnostic
- Small but active and growing community
- Active participants - IBM, Panasas, Redhat, LinuxBox, CES etc.

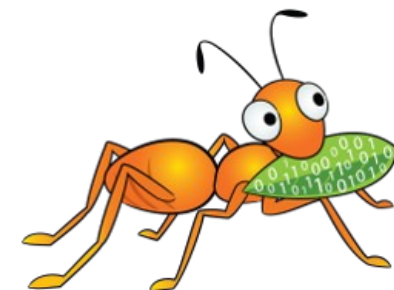
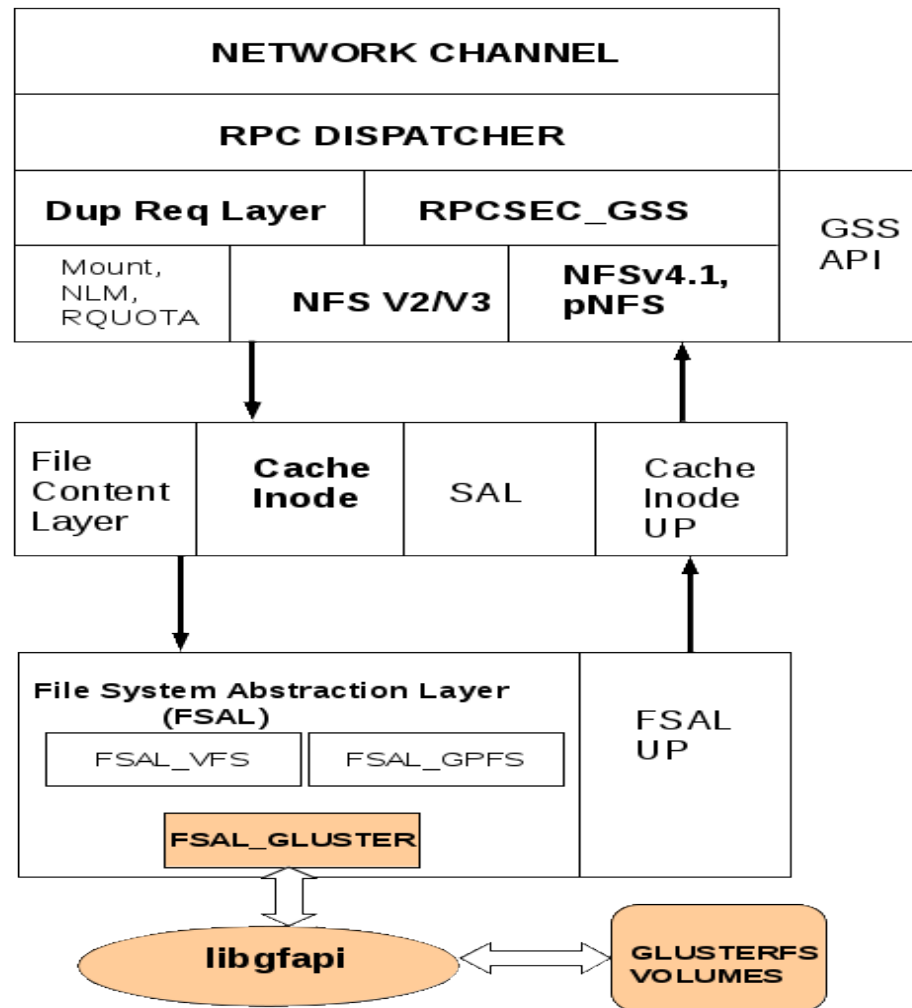


So, why user space?

- **What is so great about user-space?**
 - User Space is more flexible than kernel
 - Easy restarts, failover, failback implementation
 - Clustering becomes natural and easy
 - Targeted and aggressive caching capabilities
 - Flexible and Plug-able FSAL : FS Specific features can be exploited
 - Multi-Protocol easier to implement
 - Easier to achieve multi-tenancy
 - Easy to monitor and control resource consumption and even extend to other features like enforcing QoS
 - Manageability and debug-ability are major plus
 - GlusterFS is a user-space FS

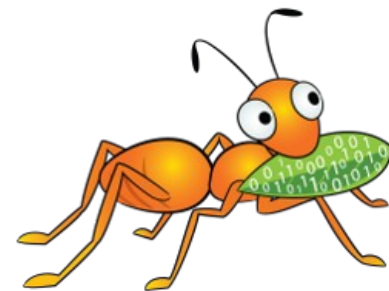


nfs-ganesha : Architecture



Other Enhancements

- **Dynamic Exports**
- **V4 R-O and W-O Delegations**
- **FSAL enhancements**
- **NFSv4 ACLs support.**
- **Open upgrade support.**
- **Stackable FSALs.**
- **PseudoFS support**
- **LTTng Integration.**
- **RDMA support through libmooshika.**

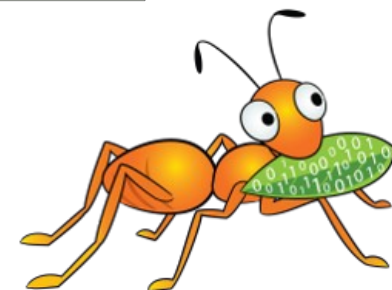
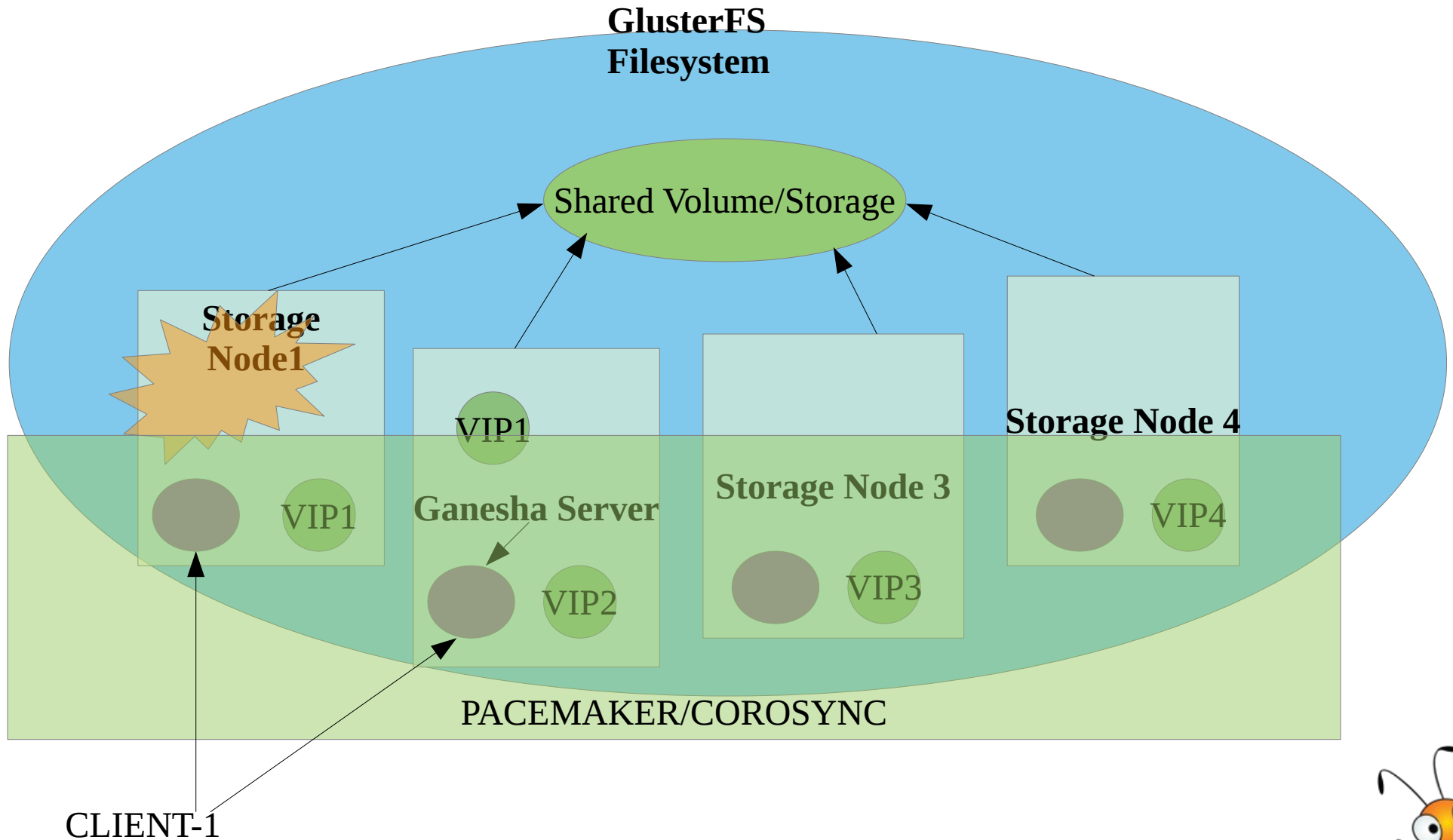


CMAL (Cluster Manager Abstraction Layer)

- Proposed Cluster Manager Abstraction Layer (CMAL) (under design and prototyping in the community):
 - makes it cluster agnostic. Works with any backend CM with a pluggable library
 - Fail-over and IP move handling
 - Enables handling of DLM across cluster or any cluster state
 - Cluster wide Grace period can be easily managed
 - Enables creation of a cluster of ganesha heads or can plugin into any existing cluster in the backend

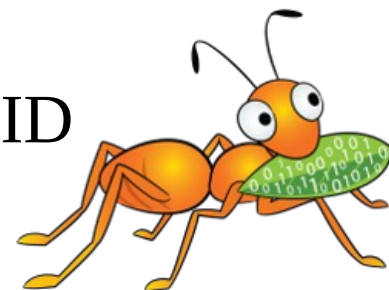


Clustered NAS



V4 Multi-head

- Pacemaker/Corosync the underlying clustering infra managing the ganesha heads
- Virtual IP associated with each ganesha head
- Persistent shared storage (another gluster vol) shared by the ganesha heads to store cluster-state (lock state /var/lib/nfs)
- ON FO client gets redirected to a pre-designated head (specified in a config file)
- Entire cluster of ganesha heads put in NFS CLUSTER_GRACE
- V3 and V4 lock recovery proceeds via SM_NOTIFY calls or via V4 lock reclaim process during grace period
 - We rely on STALE CLIENTID and STALE STATEID



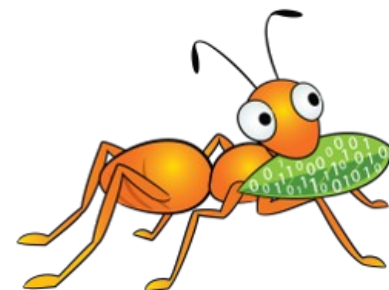
V4 Multi-head (Cont'd...)

- Upcall infrastructure added to GlusterFS for
 - Cache Inode invalidation in nfs-ganesha
 - LEASE LOCK requests conflicts (DELEGATION recall)
 - Support for share reservations
 - etc.
- Is this “true” NFSv4 active-active?
 - “True” active-active is when clients may not even get a stale stateid and are completely oblivious to changes on server side
 - Downside – too much cluster-aware state needed



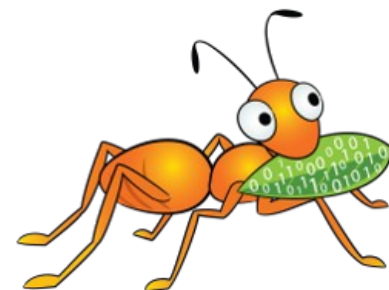
Scaling out

- Add one or more nodes (each with one or more bricks) to the GlusterFS trusted storage pool
 - Normal GlusterFS scale-out operation during run-time
 - Scale out to as many as 100+ nodes
- Add one or more nfs-ganesha V4 heads for added availability, scale-out, load-balancing etc.
 - Requires running a script to setup the new node
 - Currently supports 16 NFS server heads
 - Scale out to as many as 64+ ganesha heads (pacemaker-remote)
- Very low-cost scale out solution



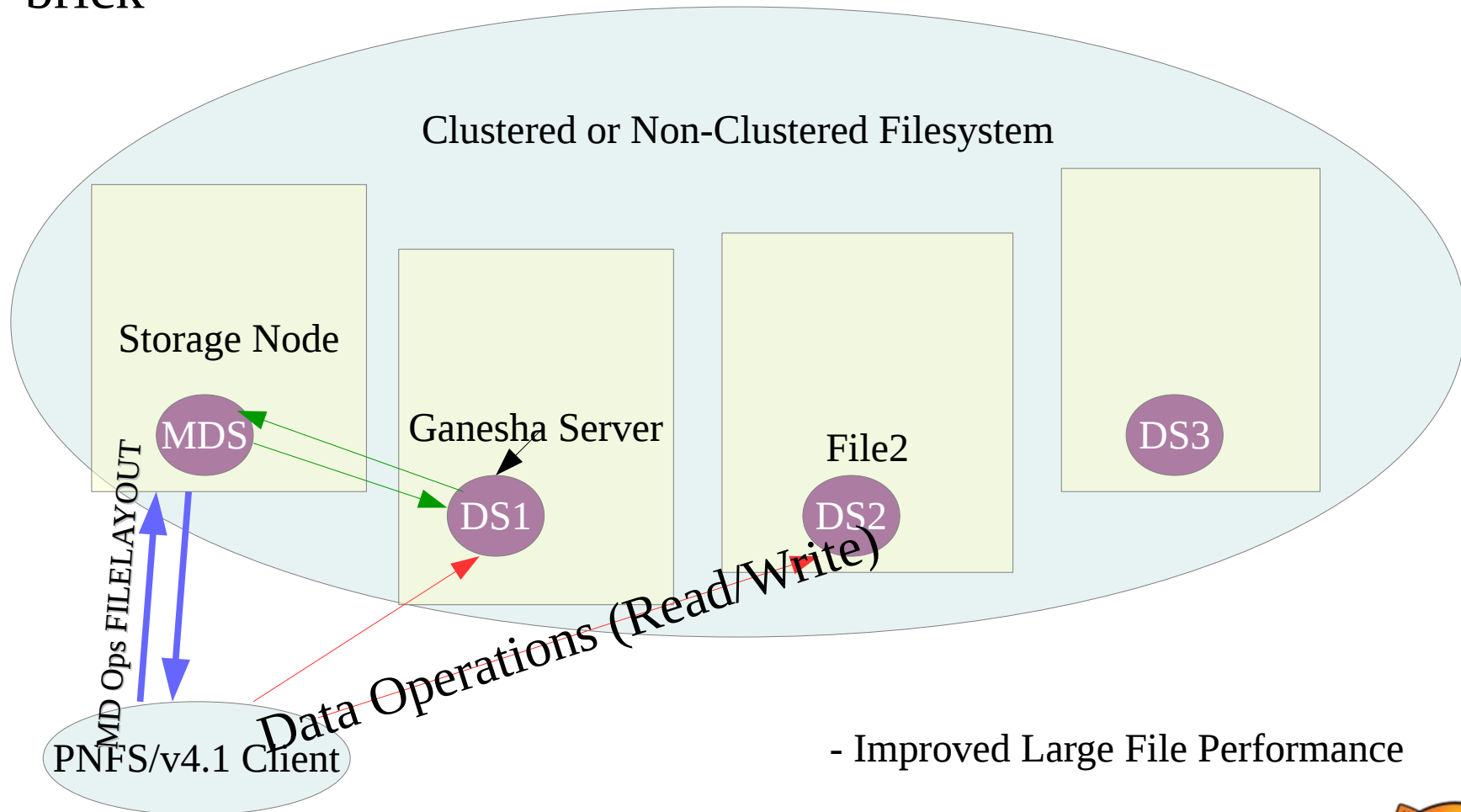
pNFS

- nfs-ganesha supports the pNFS/V4.1 protocol
- FSAL architecture enables easy plugin into core protocol support for any filesystem
- Support for pNFS protocol ops added to FSAL_GLUSTER
- nfs-ganesha and GlusterFS integration means pNFS/V4.1 support for GlusterFS Volumes

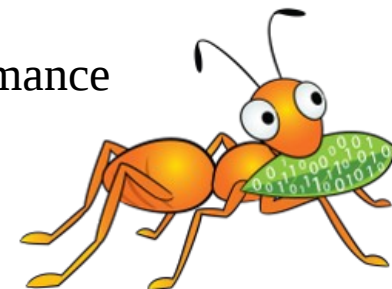


GlusterFS pNFS access using nfs-ganesha

- Supports only PNFS FILE LAYOUT – a file resides on a Gluster brick

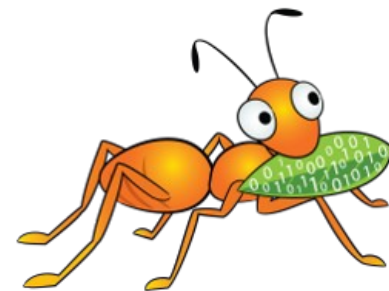


- Improved Large File Performance



pNFS Benefits

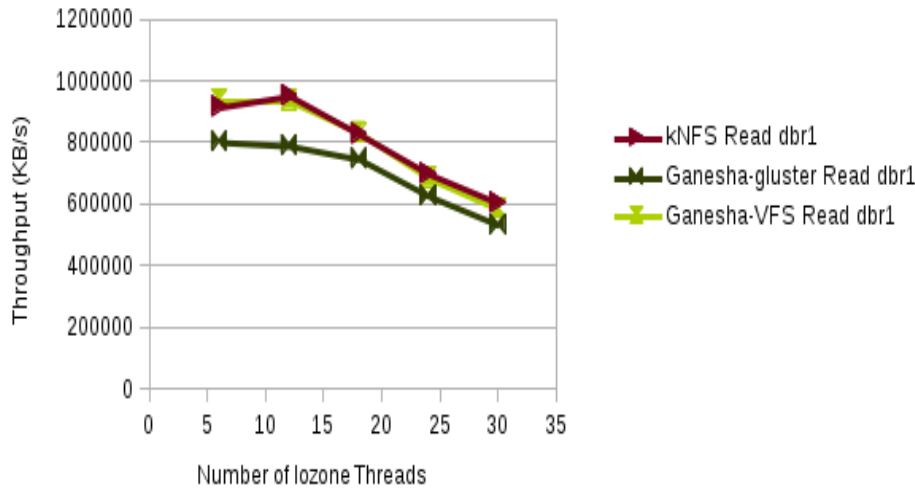
- Better bandwidth utilization
- Better load balancing across the storage pool
- Improved large file reads/writes
- More interesting configurations
 - All symmetric : every node can act as MDS as well as DS
 - Works but needs UPCALL support from GlusterFS wrt LAYOUT RECALL etc.
- Scale out by adding as many ganesha nodes as storage pool nodes depending on access requirements!
- Low cost : Probably the least expensive all open source PNFS server for FILE_LAYOUT? :)



Performance (Ganesha+Gluster in V4 versus kernel-NFS)

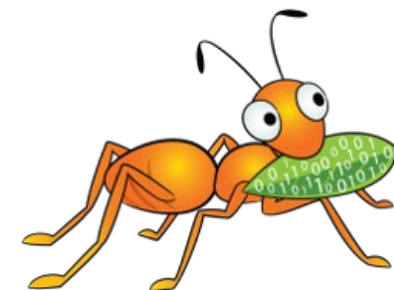
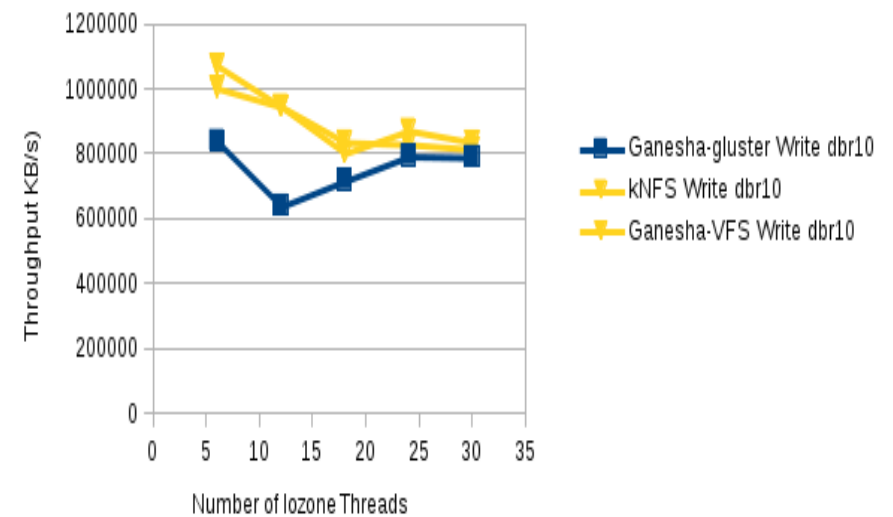
Ganesha NFS Read Throughput: Gluster vs VFS

vm.dirty_background_ratio=1



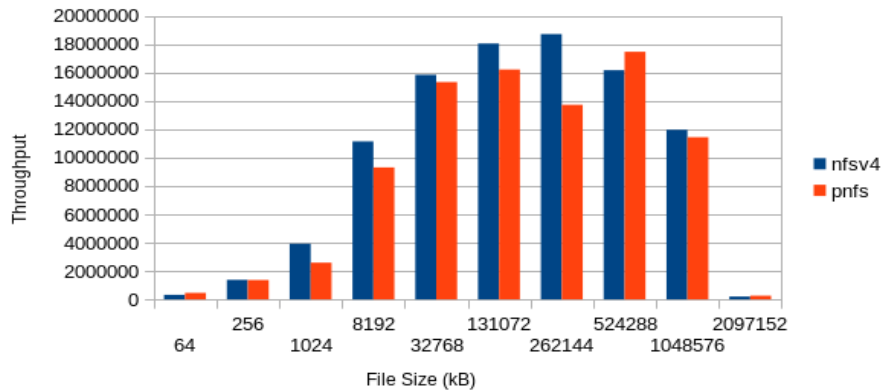
Ganesha NFS Write Throughput: Gluster vs VFS

vm.dirty_background_ratio=10

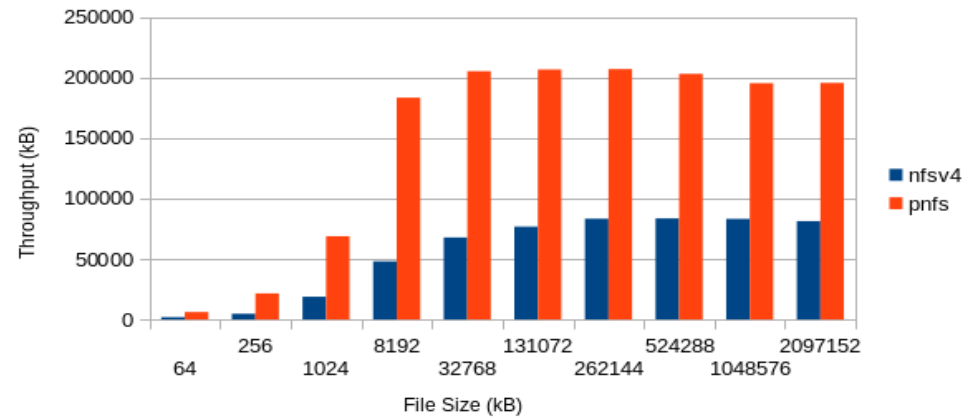


Ganesha + Gluster (V4 vs pNFS early numbers – no tuning)

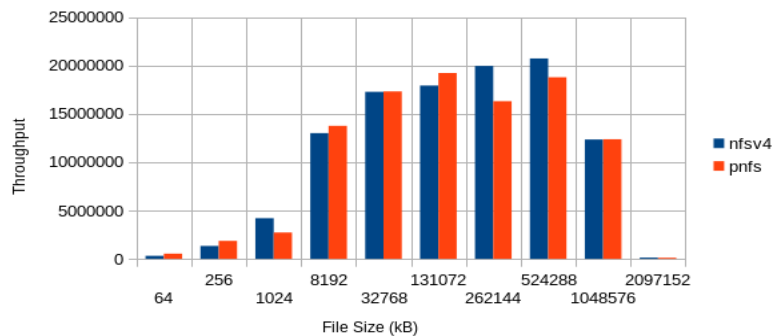
pNFS vs NFSv4 Throughput (iozone workload) - READS



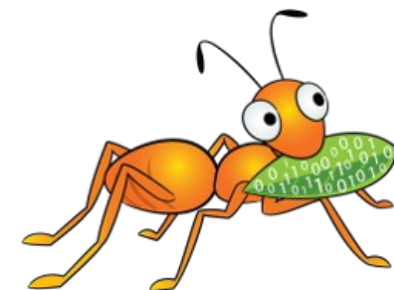
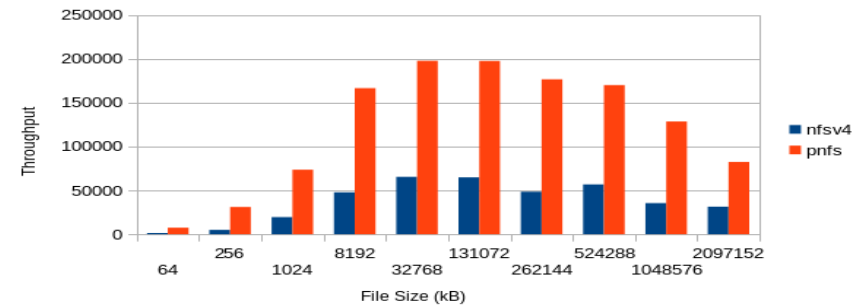
pNFS vs NFSv4 throughput comparison (iozone workload)



pNFS vs NFSv4 Throughput (iozone workload) RANDOM-READS

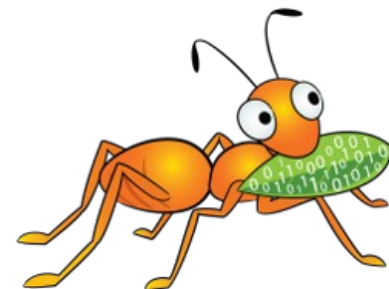


pNFS vs NFSv4 Throughput (iozone) RANDOM WRITES



Next Steps

- NFSv4 paves the way forward for interesting stuff
- Adding NFSV4.2 feature support for GlusterFS. Use case : better support for VM environments:
 - ALLOCATE/DEALLOCATE – allows VMs to do space reclamation, hole-punching
 - Server side copy support (client does not need to be involved in copying of a large file)
 - ADB – init a very large file with a pattern without transferring the entire file across the network
 - FedFS
 - Labeled-NFS
 - Flex File Layouts in pNFS



Resources

Mailing lists:

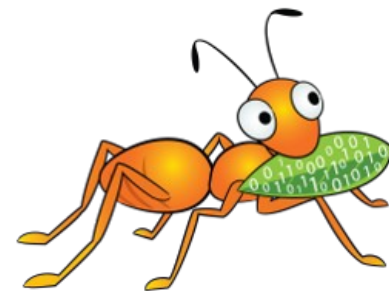
nfs-ganesha-devel@lists.sourceforge.net
gluster-users@gluster.org
gluster-devel@nongnu.org

IRC:

#ganesha on Freenode
#gluster and #gluster-dev on freenode

Links (Home Page):

<http://sourceforge.net/projects/nfs-ganesha>
<http://www.gluster.org>



Thank you!

Anand Subramanian

anands@redhat.com

