PANACHE: A PARALLEL FILE SYSTEM CACHE FOR GLOBAL FILE ACCESS

Agenda

- Why Panache…
- An Overview
- Components of Panache
- pNFS Architecture
- pNFS-GPFS
- Panache Architecture
- Consistency
- Synchronous Operations
- Asynchronous Operations
- Evaluation
- Conclusion

Why Panache?

- Need for reliability, consistency and performance of a cluster file system despite the physical distance.
- Panache is a scalable, high performance clustered file system cache for parallel data intensive applications that require wide area access.
- Withstands WAN latencies.

Overview

- Large cluster file systems(ex GPFS) can scale in capacity and support many clients but cannot mask latency and fluctuations across WAN.
- Panache is a fully parallelizable design that provides parallelism in every aspect.
- All data and metadata updates made to the cache are asynchronous.
- Features: Disconnected operations,persistence across failures,consistency,conflict handling and resolution.

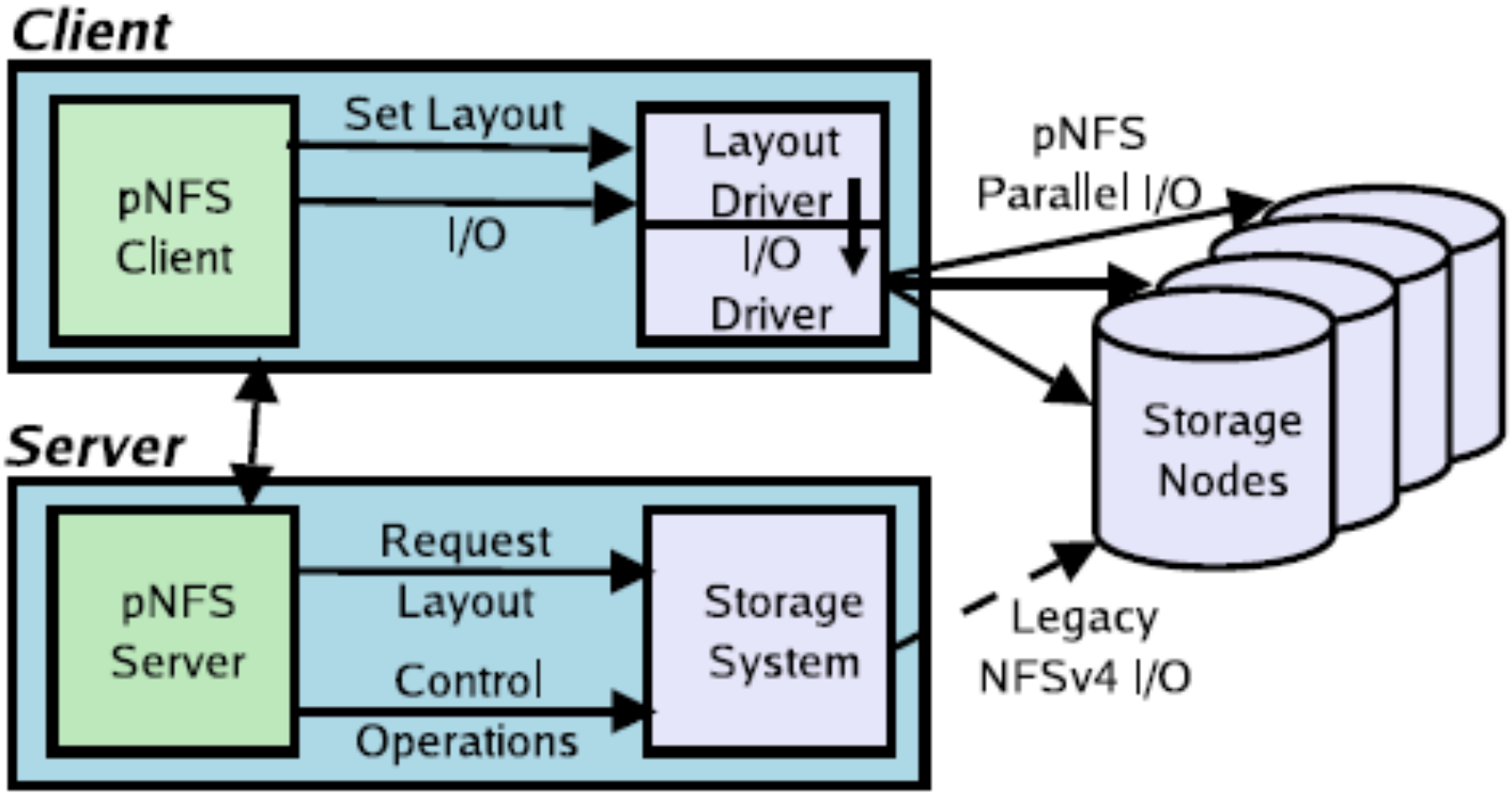
Components of Panache

- GPFS AND pNFS.
- *GPFS RECAP:*
  - ❖Shared disk Architecture
  - ❖File striped across disks-throughput
  - ❖Switching Fabric that connects file system nodes to disks.
  - ❖Distributed locking

pNFS

- Allows clients for direct and parallel access to storage while preserving OS,hardware and file system independence.
- pNFS clients and servers are responsible for control and file management operations and I/O to a layout driver on client.

pNFS Architecture



pNFS-GPFS

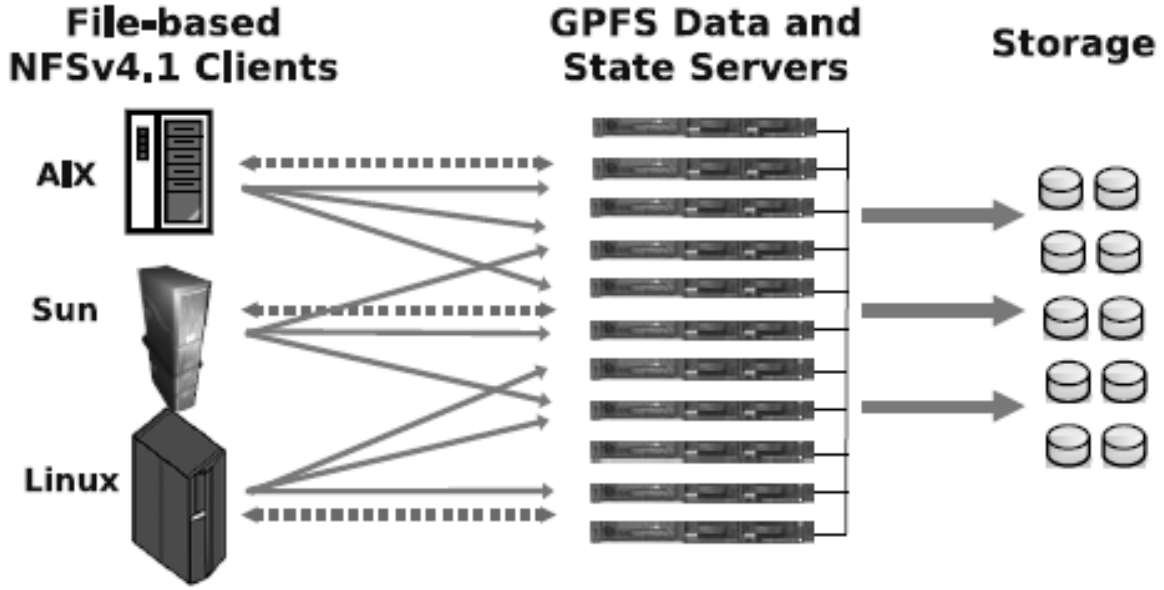


Figure 2: **pNFS-GPFS Architecture.** *Servers are divided into (possibly overlapping) groups of state and data servers. pNFS/NFSv4.1 clients use the state servers for metadata operations and use the file-based layout to perform parallel I/O to the data servers.*

Panache Architecture

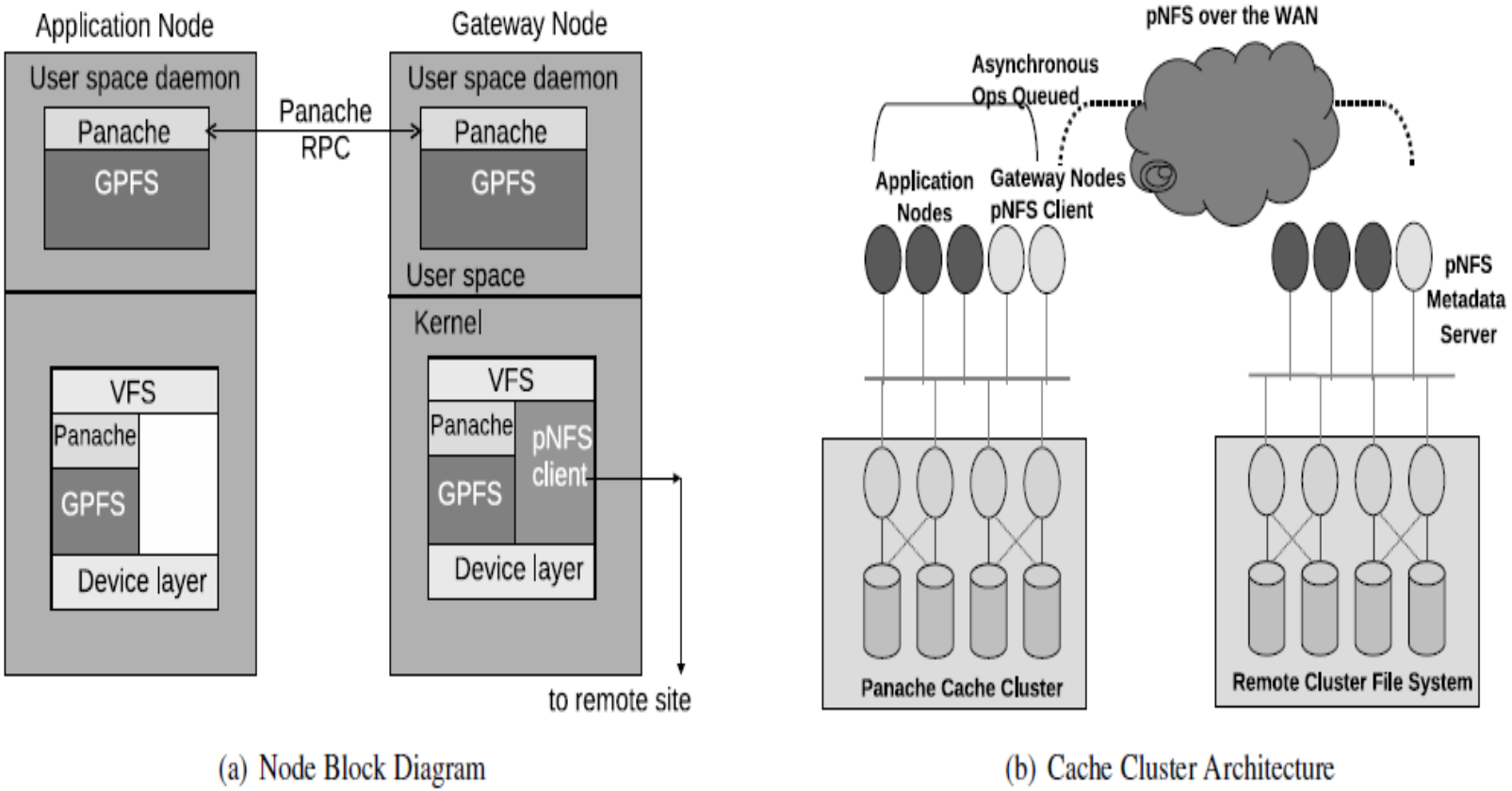


Figure 3: **Panache Caching Architecture.** *(a) Block diagram of an application and gateway node. On tje gateway node, Panache communicates with the pNFS client kernel module through the VFS layer. The application and gateway nodes communicate via custom RPCs through the user-space daemon. (b) The cache cluster architecture. The gateway nodes of the cache cluster act as pNFS/NFS clients to access the data from the remote cluster. The application nodes access data from the cache cluster.*

How Does Panache Achieve Consistency…

- Distributed Locking enables data to be locally consistent all the times for updates at the cache cluster.
- Serialized Access by electing one of the nodes as token manager for read and writes.
- Cross cluster consistency is achieved through adjusting the validity and synchronization lag.
- NFS timeout value - to check if file attributes have changed.

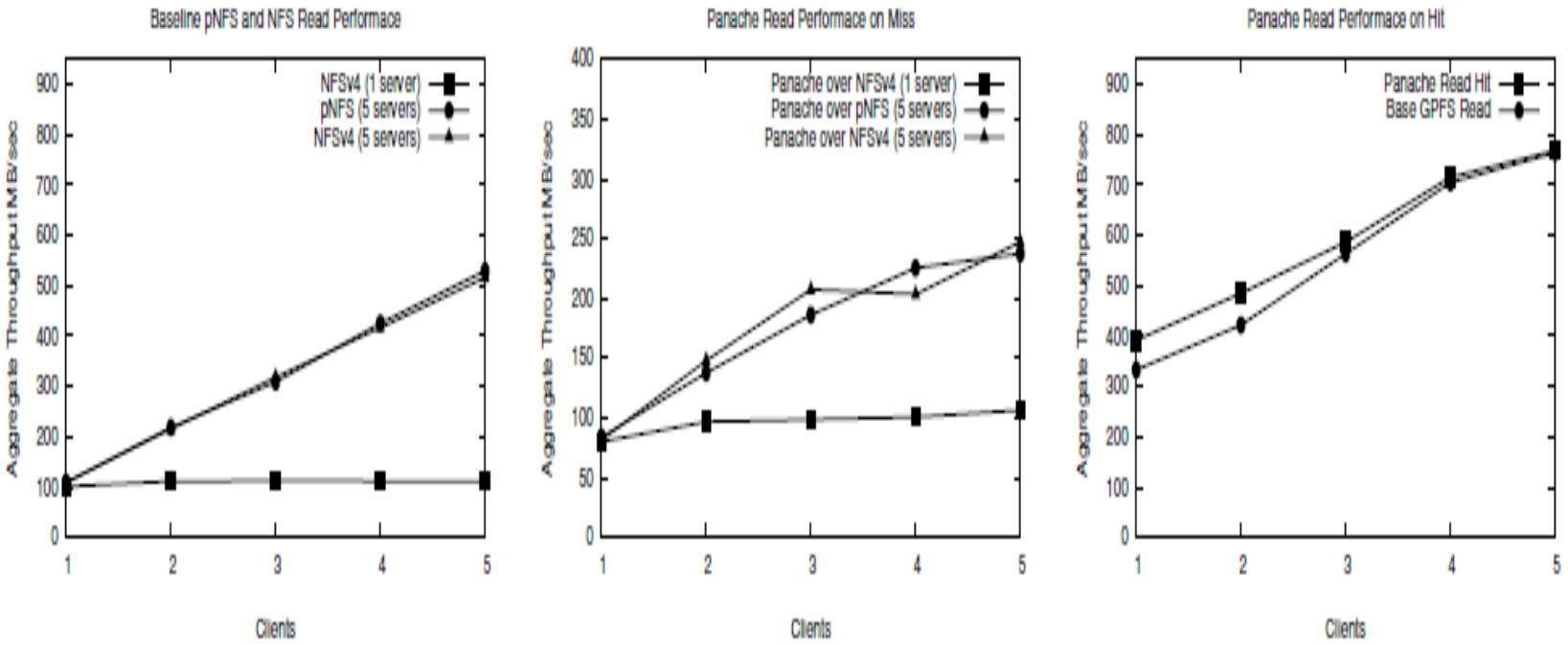
Synchronous Operations

- Synchronous operations block until the remote operation completes.
- Metadata Reads
- Parallel Data Reads
- Namespace Caching
- Data and Attribute Validation

Asynchronous Operations

- Mainly to negate the WAN latencies – Sync lag.
- Dependent Metadata Operations
- Data Write operations
- Conflict Handling
- Access Control and Authentication
- Recovery on Failure

Evaluation



(a) pNFS and NFSv4 (b) Panache Cache Miss (c) Panache Cache Hit vs. Standard GPFS

Figure 5: **Aggregate Read Throughput.** *(a) pNFS and NFSv4 scale with available remote bandwidth. (b) Panache using pNFS and NFSv4 scales with available local bandwidth. (c) Panache local read performance matches standard GPFS.*

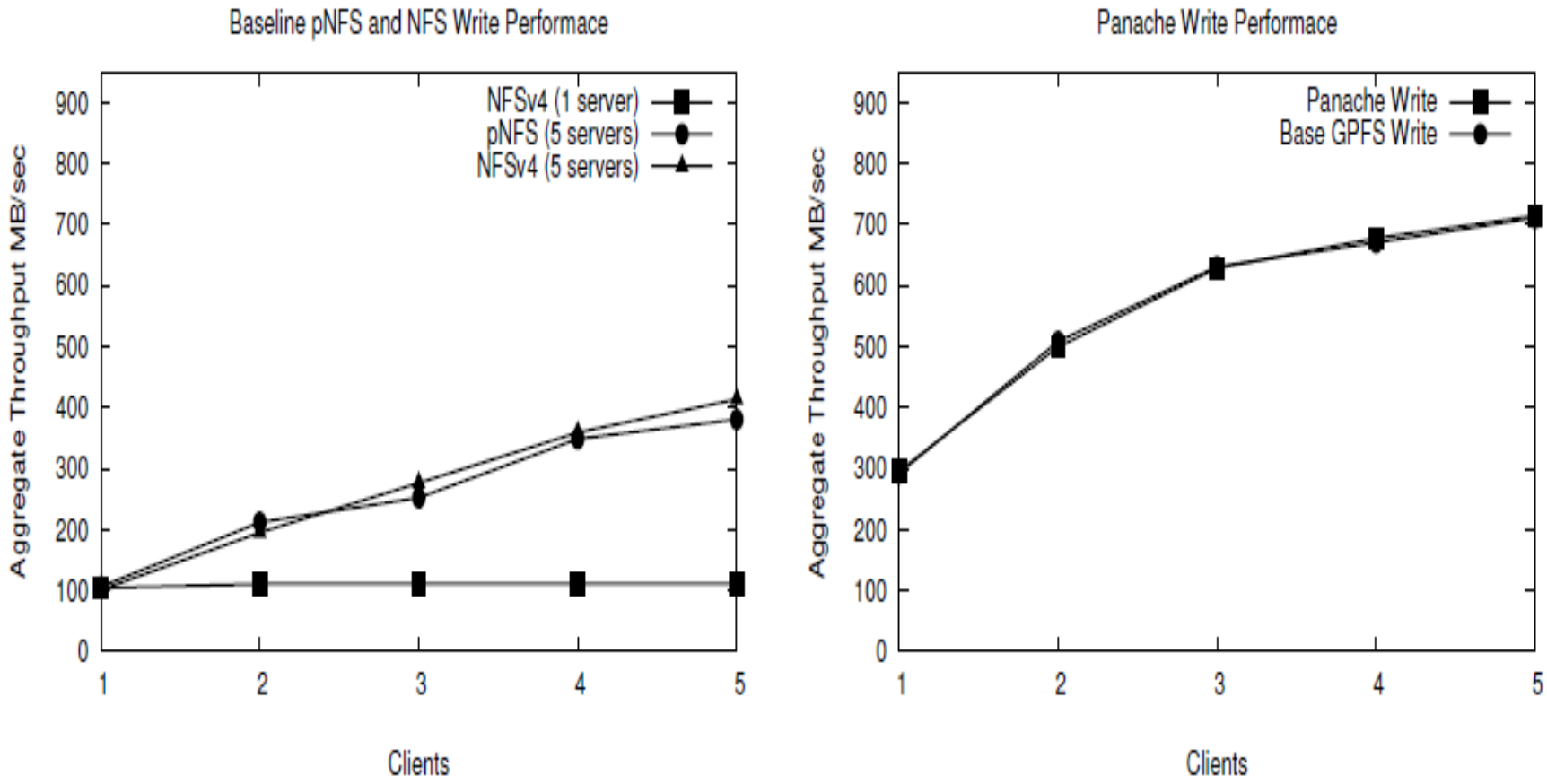


Figure 6: **Aggregate Write Throughput.** *(a) pNFS and NFSv4 scale with available disk bandwidth. (b) Panache local write performance matches standard GPFS, demonstrating the negligible overhead of queuing write messages on the gateway nodes.*

Conclusion

- Panache provides access to massive and remote datasets.
- Panache offers a fully parallelizable design
- One of the main advantages of Panache is it's ability to mask WAN latencies and outages.
- The scalability of Panache was demonstrated using the various benchmarks.