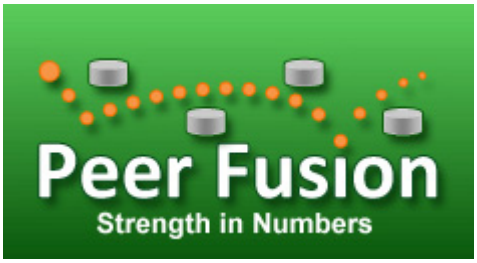


PEER FUSION PFFS BASIC ARCHITECTURE

A WHITE PAPER

Richard Levy
September 2016

ABSTRACT

Data continues to grow exponentially and the resiliency and scalability of storage systems are challenges for enterprise IT organizations. With rapidly changing requirements, planning and the containment of costs and complexities are difficult.

Peer Fusion's storage cluster technology provides flexible resiliency without replication and scalability through cluster growth. The cluster distributes the workload equally across the peers for greater performance, capacity and resiliency.

This document describes the concepts, architecture and major components of a PFFS cluster, and presents use cases.

To learn more about Peer Fusion products, services, and solutions, contact your local representative or authorized reseller, or visit www.peerfusion.com.

Peer Fusion believes the information in this publication is accurate as of its publication date. The information is subject to change without notice.

The products described in this publication may be protected by one or more U.S. patents, foreign patents, or pending applications.

The information in this publication is provided by Peer Fusion “as is.” Peer Fusion makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose.

Use, copying, and distribution of any Peer Fusion software described in this publication requires an applicable software license.

Peer Fusion reserves the right to change any products described herein at any time, and without notice. Peer Fusion assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by Peer Fusion. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of Peer Fusion.

Table of Contents

EXECUTIVE SUMMARY

This document introduces the architecture of the Peer Fusion storage cluster and the PFFS.

The PFFS is a software product for the control of a Peer Fusion storage cluster. The cluster consists of a group of peer servers that cooperate to provide data storage and resiliency. The cluster is able to operate with no data loss even with over 70 percent peer failures. The cluster can scale by adding storage capacity to the peers as well as by adding peers to the cluster. Gateways are servers that export POSIX file system semantics to client computers and control the peers in the cluster. The system is hardware agnostic and runs under most commodity Linux servers. The cluster is accessed through NFS and SMB as well as directly from the gateway.

The cluster is extremely simple to configure and administer. The plug-and-play capability of the peers allows them to perform network discovery and automatically join running clusters. Ingest performance can exceed 700 MB/s even with multiple peer failures. Read performance can exceed 400MB/s on low-end hardware.

AUDIENCE

This white paper is intended for customers, partners and IT professionals interested in understanding the architecture and major components of a Peer Fusion storage cluster.

TERMINOLOGY TABLE

Term	Definition
CLI	Command Language Interpreter. The protocol used for namespace and administration commands in a PFFS cluster.
FEC	A method for controlling errors by including extra information along with the data, which can be used to check and correct the data.
Gateway	A special peer that controls the cluster storage peers and interfaces with the client computers via NFS or SMB.
HDD	Hard disk drives. Traditional magnetic devices that store digitally encoded data.
LAN	A Local-Area Network. A computer network that spans a relatively small area. PFFS supports all network speeds including 1Gb and 10Gb and allows multiple LANs for added speed and resiliency.
MBP	The Multicast Burst Protocol. The protocol used to transfer data within a PFFS cluster.
Multicast	The simultaneous transmission of messages to multiple targets. This very efficient networking protocol is used by the MBP.
NFS	The Network File System is a distributed file system protocol originally developed by Sun Microsystems in 1984, allowing a user on a client computer to access files over a computer network much like local storage is accessed.
NIC	Network Interface Card. Used for communication between host computers (e.g. peers, gateways and client computers).
P2P	Peer-to-peer computing or networking is a distributed application architecture that partitions tasks or work loads between peers. Peers are equally privileged, equipotent participants in the application. They are said to form a peer-to-peer network of nodes.
PCIe	Peripheral Component Interconnect Express is a high-speed serial computer expansion bus.
Peers	A cluster node that participates in the storing, retrieving, and repairing of data.
PFFS	The Peer Fusion File System. The Software system that manages a cluster.
SMB	The Server Message Block (SMB) Protocol is a network file sharing protocol, and as implemented in Microsoft Windows is known as Microsoft SMB Protocol.
SSD	Solid state disk that has no moving parts and uses flash memory to store data persistently.

ZFS	ZFS is a combined file system and logical volume manager designed by Sun Microsystems. The features of ZFS include protection against data corruption, support for high storage capacities, efficient data compression, integration of the concepts of file system and volume management, snapshots and copy-on-write clones, continuous integrity checking and automatic repair, RAID-Z and native NFSv4 ACLs.

INTRODUCTION

The requirement to store ever-growing quantities of data drives several additional requirements from storage systems. The data must be protected against hardware failures and replication is very expensive for big data and can be difficult to manage. As data is ingested it must be replicated before it is deemed safe. Each data replication can double the cost of storage:

- **Hardware/software costs** – including servers, storage media (HDDs/SDDs), networking gear, software licensing, maintenance agreements, etc.
- **Datacenter costs** – including space, power and cooling, etc.
- **Additional IT costs** – doubling the equipment *always* means more work, which could require more people.

Duplicating data is done in the hope that the right two failures in the storage systems that would cause data loss will never occur. In some IT organizations this risk warrants one more data replication.

A Peer Fusion cluster overcomes these challenges by allowing IT to configure the resiliency level they require. The PFFS can be configured so that no data is lost, and applications are not disrupted, despite multiple peer failures or even the failure of over half the cluster. In practical terms, a 40-peer cluster can be configured to lose *any* 4 peers with just 10 percent overhead. On the extreme end, it is possible to configure a 40-peer cluster to lose a maximum of 38 peers and still incur no data loss. Peer Fusion gives IT organizations the power to define their requirements and configure the storage cluster accordingly.

ARCHITECTURE

The PFFS is architected to deliver scalable capacity, performance, and resiliency. The main goal of the PFFS is to distribute the workload equally among the cluster peers in order to achieve greater resiliency and performance. Parallelism is a fundamental aspect of the design and it has been emphasized throughout the implementation, starting with the design decision to develop a P2P cluster, to the use of multicast for data transmission, the ability to utilize multiple NICs, to the heavy use of threads, etc. The result is a system that can ingest data at wire-speed as it simultaneously encodes the data for resiliency.

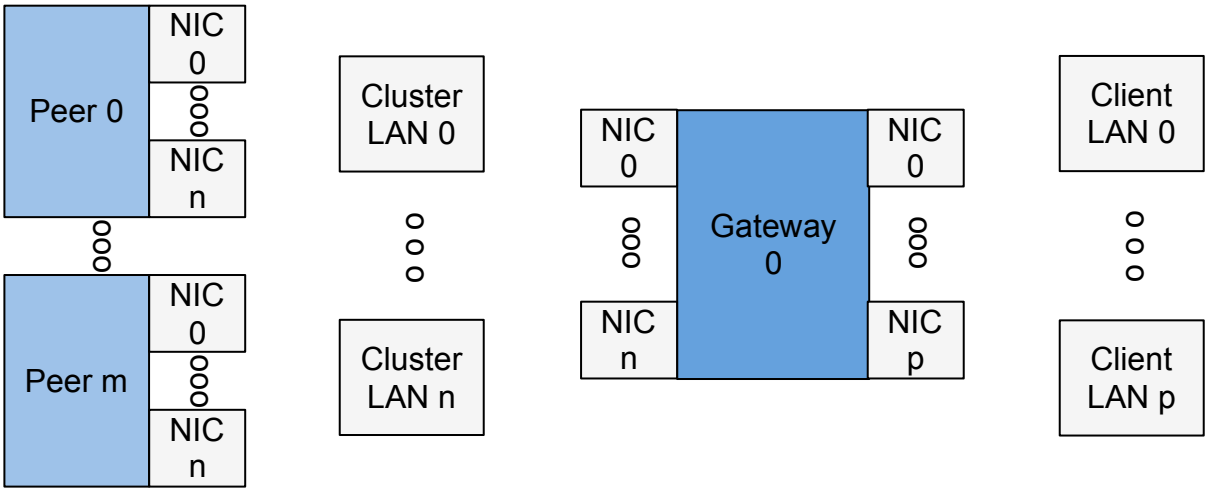


Figure 1 - PFFS Network Topology

PFFS Cluster
A PFFS cluster consists of a group of peers that store, retrieve and regenerate data. The cluster also includes one or more gateways to provide access to the cluster to client computers. The hardware for the cluster can be any commodity servers that boot a recent Linux CentOS release. It is recommended that the peers be identical, as the slowest peer will determine the overall cluster performance. The choice of hardware components will have an impact on performance and it is left up to the IT organization to make their selections based upon their requirements.

PFFS Peer
The peers have local storage capacity (e.g. HDDs, SDDs) that they contribute to the overall cluster capacity. They use the local file system (e.g. XFS, etc.) to boot. The ZFS file system is recommended for cluster data storage, though most POSIX compliant file systems should be compatible.

The peers receive commands from the gateways and cooperate among themselves to fulfill these commands. It is important to note that cooperation between the peers seldom requires that they communicate requests. The initial gateway command is used to compute by each peer the expected actions of all the peers. This includes the actions required by the gateway as well as derived actions needed to help peers fulfill the commands. For example, a gateway command to overwrite a block would automatically trigger the peer storing the block to provide the old block data to the peers generating checksums for this block.

PFFS Gateway
The gateways export POSIX file system semantics so the cluster can be mounted and accessed locally as a VFS and remotely through NFS and SMB. They interface to the cluster dashboard and various administration tools. Gateways only store the PFFS software, the configuration files and the logs. Specifically there is no user data stored in a gateway, so when a gateway fails and is replaced or rebooted, no data is lost.

The gateways run node.js to support the browser-based PFFS dashboard. The dashboard receives status updates and performance metrics for all nodes in the cluster from a gateway. Alternatively, Admintool is a command-line utility to administer the cluster.

THE PROTOCOLS

The PFFS has two main protocols: CLI and MBP. Both protocols rely on multicast for message passing. The protocols distribute messages equally across all the NIC configured. They are designed to overcome multicast artifacts to ensure the reliable and efficient exchange of messages.

CLI Protocol
The CLI protocol is designed for unstructured commands mostly related to the namespace, but also for cluster administration, healing, and heartbeats. The namespace commands include `mkdir, rmdir, link, symlink, unlink,` extended attributes (e.g. for ACLs), etc. Cluster administration commands include shutting down some peers, changing configuration parameters, etc. Healing commands are initiated by the gateway to gauge and ensure the resiliency of directory entries. Heartbeats are used for network discovery as achieving a quorum, obtaining statistics and health information about the peers.

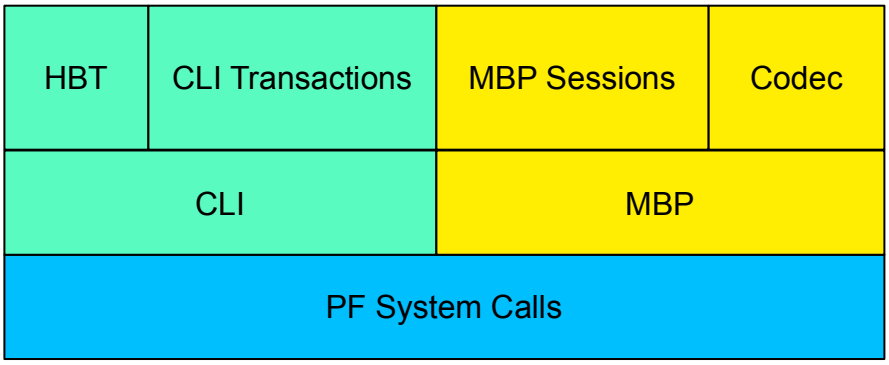


Figure 2 - CLI Protocol Stack

MBP Protocol
The MBP protocol manages the transfer of user data into and out of the cluster. MBP is a high volume and high performance protocol that ensures reliable many-to-many multicast communication. Multicast artifacts are managed, time-outs are vastly reduced and chatter is eliminated through the use of inference. Upon receiving a command from a gateway the peers can compute the expected actions of the rest of the cluster. Inference is the ability of each node that missed the gateway command to derive this information from the replies of other peers. The MBP commands include `pread, pwrite, close, stat, trunc,` etc.

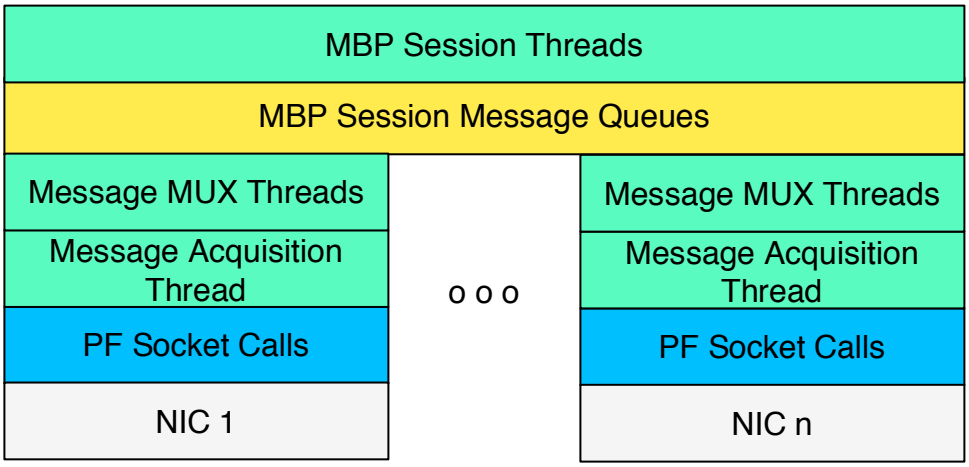


Figure 3 - MBP Protocol Stack

CONFIGURATIONS

The configuration of a cluster must specify the resources each host is allowed to use, and the multicast address of this cluster. It is possible for a host to be part of multiple clusters by either sharing or segregating the hardware resources. There are two configuration file types: one for the gateways and one for the peers. Both are very simple, and consist of a small set of parameters. The gateway will inform the peers of all the configuration parameters as well as notify the peers in real-time of any changes effected by IT.

Peer Configuration
The peer configuration files will be identical across the cluster. The peer configuration file specifies:

1. The multicast address of the cluster.
2. The size of the CLI threads pool (number of threads).
3. The network interfaces to use.
4. The root directory for the user data repository.

Gateway Configuration
The gateway configuration files have a little more information. The gateway configuration file specifies:

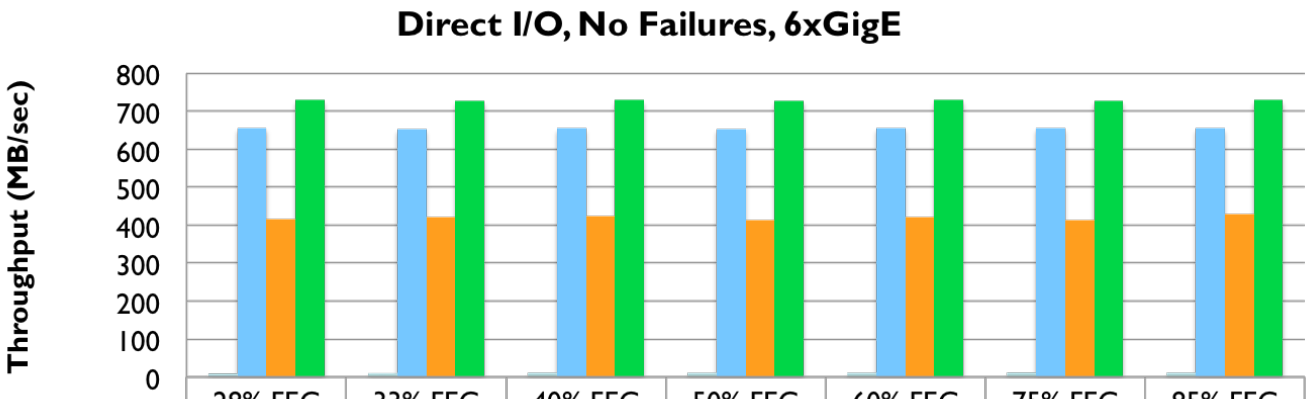
1. The multicast address of the cluster.
2. The size of the CLI threads pool (number of threads).
3. The network interfaces to use.
4. The root directory for the directory cache tree.
5. The count of pre-allocated MBP burst buffers.
6. The data staging throttle (maximum count of memory pages to use).
7. The maximum peer count (defines resiliency).
8. The minimum peer count (defines resiliency).

PERFORMANCE

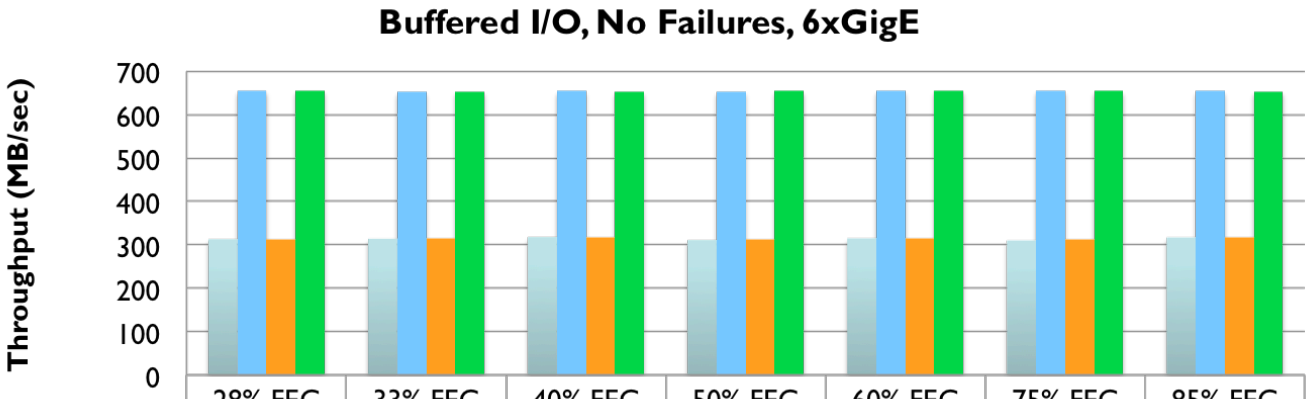
The highly parallel architecture of the PFFS ensures that there are no bottlenecks in the movement of data. Each NIC has a thread dedicated to continuously draining the NIC and queuing up messages. A pool of threads shares the load of taking the messages from the NIC's central queue and multiplexing them to the appropriate processing queue (e.g. MBP read/write operation). Similarly, transmitting blocks of data is done by distributing the blocks across all the NICs configured. The more NICs are configured, the faster the transmission of data.

The performance of a PFFS cluster is controlled by several factors:

- The hardware of the nodes: the CPU, NIC, RAM, bus, HDD, etc.
- The NIC bandwidth available: multiple NICs are aggregated to deliver near wire-speed performance.
- The count of peers: the more the better, but higher peer counts require more network bandwidth.
- The FEC level configured: drives how much encoding the peers perform to meet the desired resiliency.
- The count of peer failures: peer failures degrade performance linearly and gracefully.
- The I/O size: small I/O performs vastly better when the cluster is mounted in buffered I/O mode. Large I/O performs better when the cluster is mounted in direct I/O mode.



	28% FEC	33% FEC	40% FEC	50% FEC	60% FEC	75% FEC	85% FEC
Read 4KB	9.2	9.9	11.2	11	11.1	11.6	11.2
Write 4KB	656.2	653	655	654	656	656	655
Read 1MB	417	420	424	414	421	413	428
Write 1MB	729	727	729	728	730	728	729



	28% FEC	33% FEC	40% FEC	50% FEC	60% FEC	75% FEC	85% FEC
Read 4KB	313	314	318	311	315	310	317
Write 4KB	656.2	653	655	654	656	656	655
Read 1MB	313	314	317	312	314	312	317
Write 1MB	656	654	654	655	656	655	654

RESILIENCY

The smallest cluster configuration is 3 peers, of which only one can fail at a time. Any lesser configuration would amount to replication, which is anathema to this architecture. The level of resiliency of a PFFS cluster is configured by IT. It is possible to configure a 20-peer cluster to withstand the loss of 1 peer, or the cluster can be configured to withstand the loss of up to 18 peers, or any number in between. Resiliency has a cost: the higher the resiliency level, the more cluster capacity is occupied by checksum blocks.

When peer failures occur within the resiliency level configured, the cluster will continue to operate normally (albeit with degraded performance). Commands to read blocks from failed peers will trigger the cluster to perform on-the-

fly repairs to regenerate the data. Commands to write blocks to failed peers beyond the EOF will have no impact on performance. Commands to write blocks to failed peers below the EOF will trigger the cluster to regenerate data as needed to complete the transaction.

Healing the cluster is the process of performing repairs of missing data on previously failed peers. All the peers in parallel perform the work of healing. Multiple failed peers can be healed simultaneously with a comparable performance to healing a single peer. Healing does not repair all blocks in a peer, only the blocks storing user data. Healing does not heal all files, only those having missing or stale data. Healing can perform data repair at a rate of hundreds of megabytes per second.

USE CASES

Most commands require a quorum of peers to succeed. The quorum count is the minimum active peer count as defined in the cluster configuration. For example, opening an existing file requires that the gateway compare all the replies of the peers to make sure that, at minimum, a quorum of peers agree as to the file's version. Any peer not part of the quorum is rejected, will close the file and not participate in any I/O on the file. When the file is opened all the peers know which peers are participating and which peers are failed.

Nominal condition
When there are no peer failures all the peers agree and the minimum quorum count is exceeded. When a file is opened all the peers know that there are no failed peers and that no repairs will be necessary. This is the most common condition and provides the highest performance.

Failure condition
When there are peer failures some peers are rejected but the minimum quorum count is met. When a file is opened all the peers know which peers have failed and can independently compute for every I/O what repairs will be necessary and which peer will repair each block of data. This condition results in a graceful degradation of performance commensurate with the count of failed peers. When a failed peer rejoins the cluster it will automatically rejoin MBP sessions that have not modified their files since the peer failed. This means that unmodified files will immediately regain their resiliency and full performance. Files that were modified must be healed before the failed peers can participate in their MBP sessions.

Healing
Healing is the process of regenerating data missing from previously failed peers. The healing process traverses a directory tree and heals all the directory entries as necessary. A previously discussed, when a file is opened and there are peer failures, some peers are rejected but the minimum quorum count is met. When the file is opened for healing, the rejected peers are enlisted. All the peers know which peers were rejected and which are being healed, and can independently compute for every I/O what repairs will be necessary and which peer will repair each block of data. Healing of a file consists of the gateway reading the file in large chunks whose missing blocks are repaired by the cluster and stored by the healed peers.

CONCLUSIONS

The PFFS is a software-defined storage cluster that provides high performance, resiliency, and simple administration. The PFFS uses commodity servers that are built using off-the-shelf components. The PFFS runs under Linux and FreeBSD.

The PFFS provides significant cost savings through its flexible levels of resiliency without replication. No longer must half your storage capacity and servers be idle pending a failure. With the PFFS all your hardware is utilized at all times. The PFFS provides relief from expensive and complex storage solutions.

For more information please visit http://www.peerfusion.com