HPS

Zoya Masih

On demand file systems with BeeGFS

BeeGFS

Table of contents

Outline

Introduction

- BeeGFS is a hardware-independent POSIX parallel file system
- specifically designed for HPC
- Aggregates the capacity & performance of many servers in one namespace

History

- Was developed at the Fraunhofer Institute for industrial mathematics (ITWM)
- Originally released as FhGFS in 2005,later labelled as BeeGFS in 2014

The Name

- BeeGFS: Bee Global File System
- The appellation is referring to a colony of bees working together for a common goal in a hive, like a cluster of servers.

- BeeGFS is designed to work with various Linux distributions
- Apps don’t need to be rewritten or modified to take advantage of BeeGFS.

Outline

Architecture

- There are four main services in BeeGFS file system:
  - **Management service**: A registry and watchdog for all other services
  - **Storage service**: Stores the distributed user file contents
  - **Metadata service**: Stores access permissions and striping information
  - **Client service**: Mounts the file system to access the stored data

Management Service

- The management service is a “meeting point” for the other services.
- It is the first service which is set up in a newly deployed environment.
- It is very light-weight and typically not running on a dedicated machine
- It is not critical for performance and stores no user data
- It is watching all registered services and checks their state

Metadata Service

- The metadata service stores information about the data
- Usually, a metadata target is based on a RAID1 or RAID10
- One metadata file is created for each user-created file
- Metadata is small and grows linearly with the number of user-created files
- 512GB of usable metadata capacity are typically good for 150 m user files

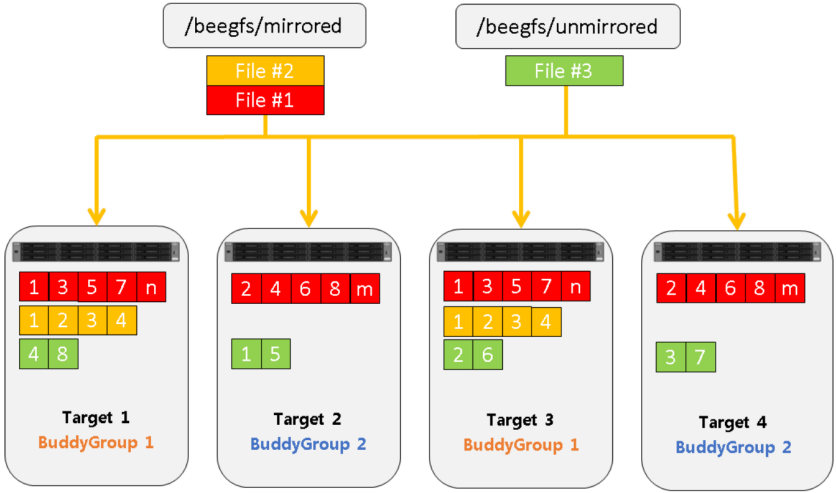
Metadata Service

- Each metadata service is responsible for its exclusive fraction of the global namespace
- Having more metadata servers improves the overall system performance.
- Adding more metadata servers later is always possible.
- Each metadata service instance has exactly one metadata target to store its data.

Storage Service

- Files get split up into chunks of fixed size
- The chunks are distributed across multiple storage targets
- Typically, a storage target is a hardware RAID6(can be directory in local FSs)
- The related metadata decides the chunksize and number of targets per file

Striping


/beegfs/mirrored
File #2
File #1
/beegfs/unmirrored
File #3
1 3 5 7 n
1 2 3 4
4 8
Target 1
BuddyGroup 1
2 4 6 8 m
1 5
Target 2
BuddyGroup 2
1 3 5 7 n
1 2 3 4
2 6
Target 3
BuddyGroup 1
2 4 6 8 m
3 7
Target 4
BuddyGroup 2

Storage Service

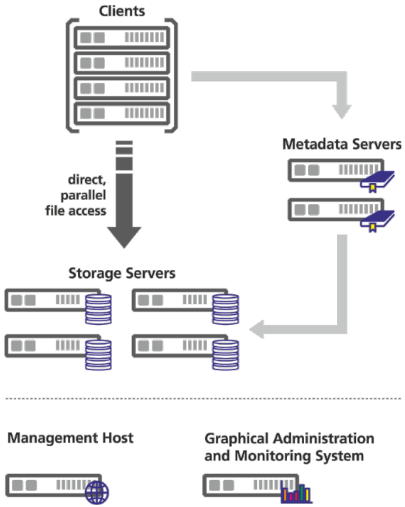
- It can aggregate small IO requests into larger blocks before writing the data out to disk
- It is also able to serve data from the cache if it has already been recently requested by another client

```
hadoop@hadoop-VirtualBox:/mnt/beegfs$ beegfs-ctl --getentryinfo /mnt/beegfs/vid
eo.mp4 --verbose
Entry type: file
EntryID: 0-6421BC57-2
Metadata node: hadoop-VirtualBox [ID: 2]
Stripe pattern details:
+ Type: RAID0
+ Chunksize: 512K
+ Number of storage targets: desired: 4; actual: 1
+ Storage targets:
  + 301 @ hadoop-VirtualBox [ID: 3]
Chunk path: u0/0/r/root/0-6421BC57-2
Dentry path: 38/51/root/
hadoop@hadoop-VirtualBox:/mnt/beegfs$
```

Client Service

- Runs on the client nodes and provides access to the BeeGFS file system
- Manages the communication between the client and server components

Architecture



Usage in HPC

- BeeGFS is widely used in HPC systems around the world
- Used in **HPC clusters** to provide fast, parallel file access to large datasets.

Usage in HPC

- BeeGFS is an important part of three European HPC projects:
- DEEP-ER, EXANODE and EXANEST.

- SuperMUC-NG cluster at the Leibniz Supercomputing Center uses BeeGFS

6,336 Thin compute nodes each with 48 cores-144 Fat compute nodes each 48 cores

BeeOND

- BeeOND: BeeGFS On Demand
- Provides on-demand access to BeeGFS file system data

BeeOND

- Traditionally, data are loaded into memory when a storage device is mounted, or a file is accessed
- With on-demand FSs, files and data are accessed only when they are requested by an app or user

BeeOND

- It reduces the amount of memory and processing resources required for storage operations
- It also provides fault-tolerance features, such as data replication and automatic failover

BeeOND

- BeeOND creates a shared parallel file system for each job across all compute nodes which are involved in the job
- It provides advantage of a single name space across multiple machines, and the flexibility and performance of a shared parallel file system.
- Combining the SSDs of multiple compute nodes, gets to high bandwidth and also gets to a system that can handle very high IOPS.

Outline

Installation

- Go to https://doc.beegfs.io
- **Get the Sources:** `git clone https://git.beegfs.io/pub/v7 beegfs-v7 cd beegfs-v7`
- **Install the dependencies**
  https://git.beegfs.io/pub/v7/-/blob/master/README.md
- **Build:**
  make

Installation

- **Add the public BeeGFS GPG key** https://www.beegfs.io/c/download/
- **Download the repository** https://www.beegfs.io/c/download/
- **Update**
- **Install the BeeGFS services**
  https:
  //doc.beegfs.io/latest/advanced_topics/manual_installation.html

Outline

- The BeeGFS architecture is designed to provide fast, scalable, and reliable FS access for HPC and big data workloads
- The distributed metadata and parallel FS enable high throughput and low latency access to file data
- The management tools provide administration and monitoring of the FS

References

- http://www.beegfs.de/docs/whitepapers/Introduction_to_BeeGFS_by_ThinkParQ.pdf
- https://doc.beegfs.io/latest/architecture/
- https://doku.lrz.de
- https://gauss-allianz.de/en/hpc-ecosystem