High Performance Computing Storage Options

Selecting AWS Partner Network storage solutions
for High Performance Computing on Amazon Web Services

December 2018

© 2018, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Notices

This document is provided for informational purposes only. It represents AWS's current product offerings and practices as of the date of issue of this document, which are subject to change without notice. Customers are responsible for making their own independent assessment of the information in this document and any use of AWS's products or services, each of which is provided "as is" without warranty of any kind, whether express or implied. This document does not create any warranties, representations, contractual commitments, conditions or assurances from AWS, its affiliates, suppliers or licensors. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

Contents

Abstract

High Performance Computing (HPC) is an important and complicated workload for many customers. Many HPC workloads incorporate large compute clusters and need to process a lot of data. HPC workloads require storage systems that can keep up with performance and capacity demands. This paper explores storage options when running HPC workloads on Amazon Web Services (AWS).

Introduction

HPC can take many forms across many industries. Technical computing is present in everything from electronics design and manufacturing to genomics and film rendering. HPC often means massively parallel compute clusters used to analyze existing data or generate new data. In either case, data needs to be acted upon and persisted by each node in a cluster as fast as possible. Many HPC workloads run for a long time and take checkpoints. These checkpoints create large amounts of intermediate data, but enable snapshotting and restarting in case of interruption. This is particularly important on the cloud, where specialized offerings like Amazon Elastic Compute Cloud (EC2) Spot Instances may incorporate temporary resource shortages as a tradeoff to being extremely cost-effective.

There are many benefits of running HPC on AWS, and for more information about the benefits see the HPC section of the AWS website[1]. In this document, we explore HPC from a storage perspective. Cluster management and other compute aspects are out of scope for this paper, but addressed in the introduction to HPC on AWS whitepaper[2] for those interested in learning about schedulers and solvers on AWS.

HPC Storage

Programs that run in HPC clusters usually require shared access to storage. Any program on any compute node can read from and write to any file or parts of any file. Because the read or write operation could come from anywhere, storage architectures for HPC workloads aim to avoid potential performance bottlenecks. Distributed locking systems and parallel access to metadata and data help keep things working at high speed. Add to that low latency networking and high-performance media, and it's a formula for success. AWS componentry is an easy way to get started with HPC storage on the cloud, and can be used in conjunction with integrated solutions from AWS Partner Network (APN) Technology Partners. For background, let's briefly examine the anatomy of a distributed parallel cluster.

A file system understands how to map a file hierarchy and its contents to physical media, such as a drive. Only a single host can request a file. Even if the host is a Network File System (NFS) server, the NFS server software arbitrates remote requests from clients to become local requests. The media could be a

physical device or an aggregate of devices presented as a logical device, like in a Storage Area Network (SAN).

A distributed file system spans more than one set of logical media, but requests are still terminating in one point. NFS servers are a good example of this, because an NFS client only knows how to access the NFS server without understanding the underlying physical topology or having any way to access it.

A clustered file system allows multiple hosts to access the same data at the same time, fully supporting concurrent reads and writes and arbitrating any contention. In this model, every participating host can make requests, but hosts find responses on the same shared media. In clustered environments, participating hosts often export the shared namespace as a networked resource. Requests coming from remote clients always terminate on the limited number of participating hosts, like in the simple case of a single host acting as an NFS server.

A clustered distributed file system builds on these concepts by breaking data into pieces and spreading file system data across multiple hosts and multiple media. This does a better job of equalizing capabilities, because each participating host doesn't have to compete with other hosts for access to storage media. Requests are routed through specific hosts depending on which piece of data is needed. A clustered distributed system still might have a request-handling bottleneck if the data being requested doesn't vary. This is referred to as a hot spot. In a scenario where participating hosts re-export their file system to the network, remote clients asking for the same piece of data would cause all requests to be routed through the same host irrespective of which of host terminates the client connection.

To solve for this, parallel file systems support a many-to-many relationship between requests and responses. Every participating host can initiate or service requests, accessing data through many independent pathways. They work together to provide a single file system hierarchy, or namespace. This grid or mesh includes a special client software to decipher the file system namespace and inform even remote clients as to the topology of the data layout. By avoiding any form of request serialization, parallel file systems historically have provided the best performance for HPC workloads. Before the cloud, it was common to see parallel file systems talk to clients via a front-end network, and utilize a segregated back-end network for behind-the-scenes data access. Reliability was generally handled by underlying storage systems, often complex and expensive SAN topologies.

The advent of the cloud and storage solutions designed for the cloud greatly simplifies the deployment and configuration process for HPC workloads. In addition to scaling performance and capacity, distributed parallel file system clusters in the cloud also take fault tolerance into consideration. Participating hosts are fully aware of device health and have mechanisms in place to minimize interruptions in the event of component failure, thanks to the client's understanding of what is happening behind the scenes.

AWS Storage Services

AWS provides several storage services useful for HPC. Some can be used on their own, while others are building blocks for APN Partner solutions. What you choose depends largely on your workload attributes. The best way to validate your storage choice is to try your workload. Benchmarks and other synthetic measurements are rarely adequate in the tailored and ever-changing world of HPC.

Amazon S3

Companies today need the ability to simply and securely collect, store, and analyze their data at a massive scale. Amazon Simple Storage Service (S3) is object storage built to store and retrieve any amount of data from anywhere – websites and mobile applications, corporate applications, and data from Internet of Things (IoT) sensors or devices[3]. Amazon S3 delivers 99.999999999% durability, and stores data for millions of applications used by market leaders in every industry. Amazon S3 also provides comprehensive security and compliance capabilities that meet even the most stringent regulatory requirements, giving customers flexibility in the way they manage data for cost optimization, access control, and compliance. Amazon S3's query-in-place functionality allows you to run powerful analytics directly on your data at rest in Amazon S3. Lastly, Amazon S3 is the most supported cloud storage service available, with solutions available from a large community of third-party solution providers, systems integrator partners, and other AWS services.

Amazon S3 is a highly scalable and high durable storage platform for HPC applications that support an object interface. Amazon S3 includes many features, such as lifecycle management, that allow you to move less frequently accessed down to lower storage tiers for more cost effective solution[4]. Even if your HPC application does not directly support Amazon S3, you can use Amazon S3 as a data repository to hold your dataset, which can be ingested into

your processing file system. There also are APN Partner solutions that provide traditional filesystem access via SMB, NFS, and POSIX clients while using Amazon S3 as part of their solution for data storage and/or data protection.

Amazon EBS

Amazon Elastic Block Store (EBS) provides persistent block storage volumes for use with Amazon EC2 instances in the AWS Cloud. Each Amazon EBS volume is automatically replicated within its Availability Zone (AZ) to protect you from component failure, offering high availability and durability. Amazon EBS volumes offer the consistent and low-latency performance needed to run your workloads. With Amazon EBS, you can scale your usage up or down within minutes – all while paying a low price for only what you provision.

If you want to either build your own local or network file system or utilize an APN Partner solution, Amazon EBS provides block storage to your Amazon EC2 compute instances. Amazon EBS includes different performance characteristics, such as IOPS optimized Provisioned IOPS SSD (io1) volumes where customers can pre-provision the IO they need, and Throughput Optimized HDD (st1) volumes that are designed to be used with workflows that are driven more by throughput than IOPS.

Pro Tip: Aside from the IOPS and/or throughput the volumes provide, when using multiple volumes, it is important to consider the maximum performance the Amazon EC2 instance can provide. Amazon EBS optimized instances have published maximums that should be considered when choosing the instance on which to run your storage solution[5].

Amazon EC2 Instance Store

An instance store provides temporary block-level storage for your instance. This storage is located on disks that are physically attached to the host computer. Instance store is ideal for temporary storage of information that changes frequently, such as buffers, caches, scratch data, and other temporary content, or for data that is replicated across a fleet of instances, such as a load-balanced pool of web servers.

Amazon EC2 Instance Store, like Amazon EBS, provides non-persistent block storage local to instances. An instance store provides high-speed local disk storage that can serve custom-built solutions for processing data that does not

need to be persisted. Instance stores also are used by some APN Partner solutions that stripe data across multiple instances.

Amazon EFS

Amazon Elastic File System (EFS) provides simple, scalable, elastic file storage for use with AWS Cloud services and on-premises resources. It is easy to use and offers a simple interface that allows you to configure file systems and get started in mere moments. Amazon EFS is built to elastically scale based on usage. Like Amazon S3, you just use Amazon EFS—no provisioning required. The high performance sweet spot of Amazon EFS is aggregate throughput, catering to parallelized, read-heavy workloads. Behind the scenes, Amazon EFS enhances its capabilities without disrupting applications to handle additional capacity and load requirements. As a regional service, Amazon EFS is designed for high availability and durability storing data redundantly across multiple AZs.

Amazon EFS presents an NFSv4 export that can be used as working storage, as a repository, or in conjunction with other services to provide a fully managed storage solution. Amazon EFS recently announced provisioned throughput, which allows you to increase throughput disproportionately to capacity, so even smaller storage capacity workloads can take advantage of Amazon EFS' scale and ease of use.

AWS Partner Solutions

APN Technology Partners provide different solutions for supporting HPC workloads, including the Partner solutions listed below. Check out the APN site to learn more about APN partners[6].

WekaIO Matrix

WekaIO has a product called Matrix, which is a scale-out NAS platform. Matrix was designed specifically for the AWS Cloud, and was built with optimization in mind. Matrix can run on AWS as well as on premises. MatrixFS, which is the underlying file system of the Matrix NAS, is a distributed parallel file system. Matrix includes many features that customers would expect from a scale-out NAS system, including a Global Name Space, support for NAS protocols like NFS and SMB, and ability to scale both performance and capacity. Some key differentiators for Matrix include linear performance scalability, seamless tiering with Amazon S3, snapshotting to Amazon S3 with the ability to snapshot

the whole cluster, and use of a native client to provide true parallel processing across multiple nodes (currently only available for Linux).

Architecture

From an architectural perspective on AWS, Matrix runs "I" instance families for the storage components while the client can run on any instance family, allowing for a wide range of workloads. Matrix requires a minimum of six nodes to have a fully formed cluster. Under the covers, Matrix software runs inside a container, which allows it to discreetly mange its resources so it can either be installed on instances by itself or alongside other applications in a hyperconverged deployment. Matrix has configurable redundancy from N+2 to N+4 and can scale from the base six nodes up to 1,000s of nodes. MatrixFS distributes all data and metadata across the nodes in the cluster. Matrix is designed to have an active tier of data on instance storage and an inactive tier on Amazon S3. While tiering is optional and a cluster can be run with only an active tier on SSD, it is recommend that most implementations use the Amazon S3 tier, which can enabled any time after the cluster is running.

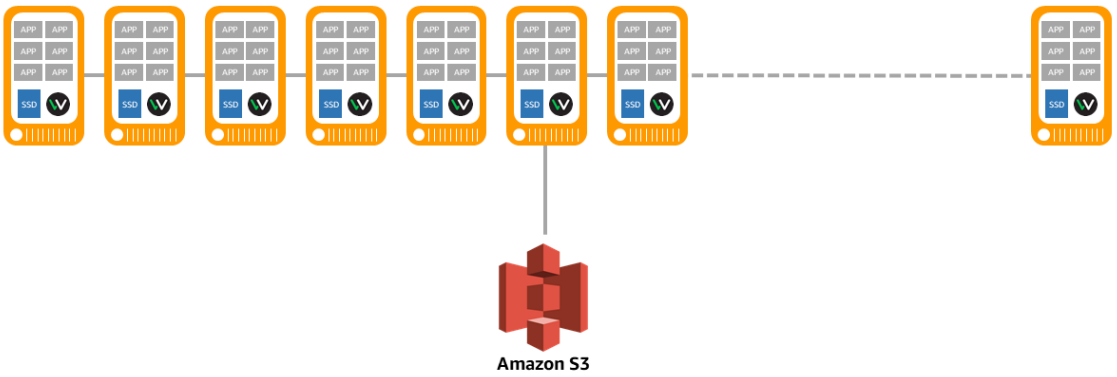

Figure 1: Weka IO MatrixFS Cluster

When to use: WekaIO provides distributed parallel file system performance and Amazon S3 persistent without the need for tuning. While you can achieve the maximum performance by utilizing the WekaIO client for Linux, this solution supports high-speed workloads over NFS and SMB, just not parallel. This makes it a good solution for most HPC workloads.

For more information on Matrix, visit the WekaIO Website[7]. You also can visit start.weka.io to quickly size a cluster based on capacity or performance needs[8]. WekaIO can be deployed via the AWS Marketplace[9].

Qumulo QF2

Qumulo File Fabric (QF2) is a software-defined, scale-out NAS system. It offers a balance of performance, scalability and ease of use. QF2 has a simple graphical user interface (GUI) as well as full REST API, making it easy to manage. QF2 can run on premises and on AWS and it offers a continuous replication feature, which allows a QF2 cluster running on premises to be in sync with a QF2 cluster running on AWS and/or between QF2 clusters running in multiple AWS locations. QF2 also includes built-in file level analytics so that users can quickly query information about files and visualize performance metrics down to the client, user, and file level.

Architecture

A base QF2 cluster requires either four, six, or 10 nodes. You can add additional nodes at any time as an online operation. On AWS, these nodes are Amazon EC2 instances running the QF2 software. Each node in the cluster has a certain amount of Amazon EBS storage, which is used to persist the data. QF2 instances use a combination of SSD- and HDD-based Amazon EBS volumes to balance performance and cost. Clients access the cluster through NAS protocols, such as SMB or NFS. Clients can access any data through any node regardless of where the data resides. QF2 does not have a custom client at this time, so when managing client distribution, it is helpful to use a domain name system (DNS) such as Amazon Route 53 or go through a load balancer[10].

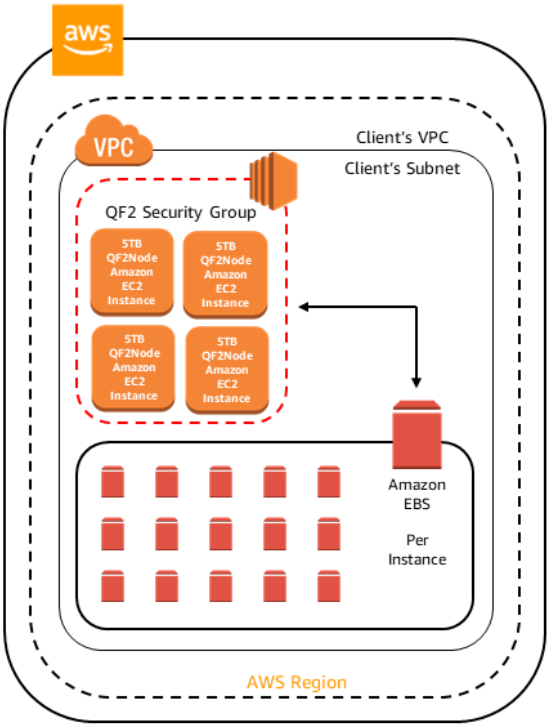

Figure 2: Qumulo QF2 20-Node Deployment Topology

When to use: Qumulo is an ideal solution for any HPC cluster that requires data processing by heterogeneous compute systems. Qumulo provides SMB and NFS protocols, so data can be natively accessed by both Windows and Linux nodes. An additional consideration would be balancing traffic between the nodes. DNS with network protocols like NFS and SMB do not provide the same parallelism as a POSIX-based parallel client. QF2 does allow traffic to be balanced between nodes and should be used with HPC clusters that can be distributed and handle balancing requests between nodes.

For more information about QF2, see the QF2 Overview[11] on Qumulo's website or download the technical whitepaper[12]. QF2 also is available on AWS Marketplace as a free single node trial[13], 5TB per node version[14], and 20TB per node version[15].

Spectrum Scale

IBM Spectrum Scale is a scalable clustered file system based on IBM's General Parallel File System (GPFS) and associated management software and clients.

IBM Spectrum Scale de-coupled architecture allows configuration options that can scale on different vectors, including: performance (bandwidth and IOPS) of

the data and metadata layers, capacity, and number of nodes that can mount the file system.

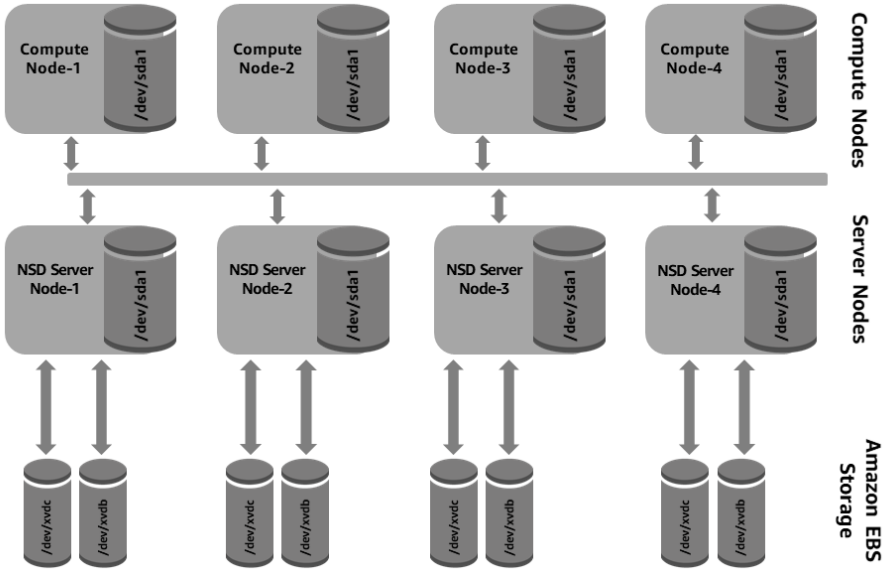

Figure 3: General Architecture Components of IBM Spectrum Scale

The diagram above depicts the general architecture components of the IBM Spectrum Scale cluster as it would be deployed in the available AWS Quick Start implementation on AWS.

One of the key GPFS functions within Spectrum Scale is the ability to ship IO requests to compute nodes using the Network Shared Disk (NSD) protocol. NSD is a client-server protocol, with NSD servers providing access to storage that is visible on servers as block storage. A typical GPFS cluster comprises of a smaller number of NSD servers and a large number of NSD clients.

IBM Spectrum Scale provides various configuration options and access methods through the client. These include traditional POSIX-based file access with features such as snapshots, compression, and encryption.

Architecture

When deploying a cluster on AWS using the AWS Quick Start, a full architecture is deployed, as shown in Figure 4.

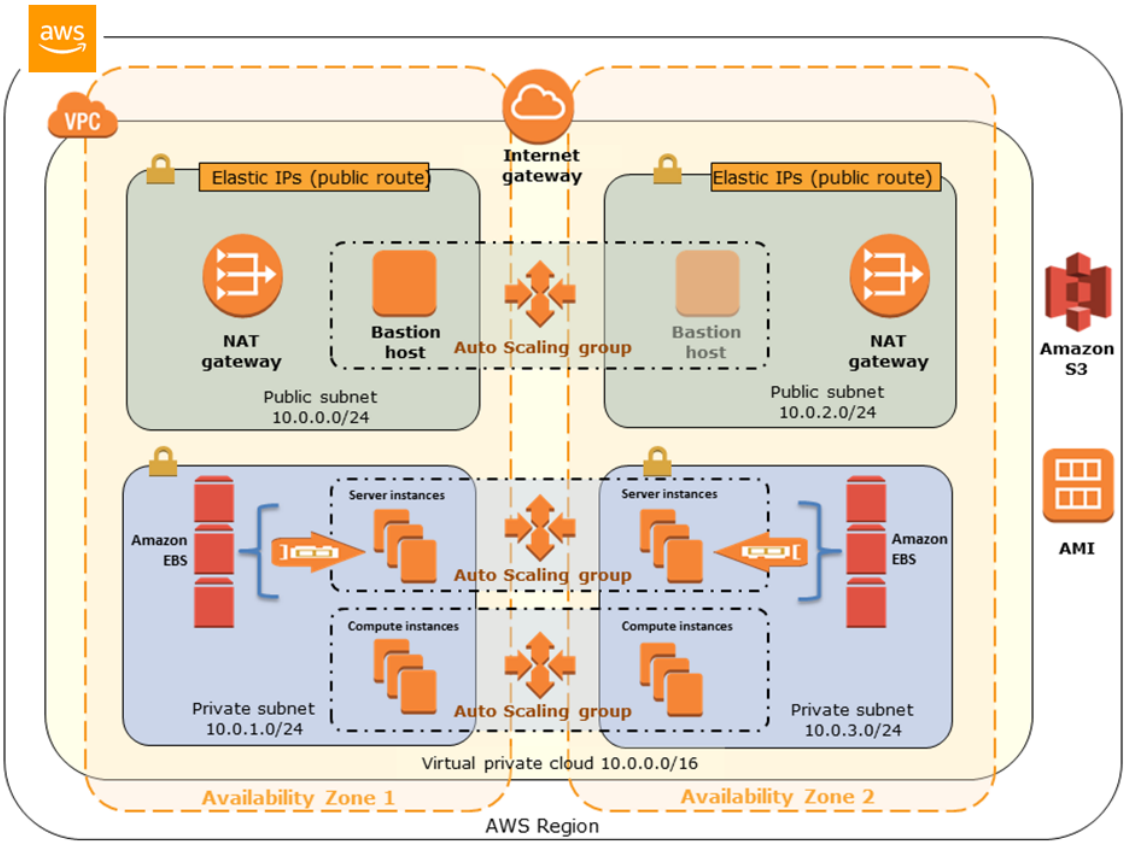

Figure 4: IBM Spectrum Scale Reference Architecture

This includes automatic deployment of AWS services and instance roles in the IBM Spectrum Scale cluster architecture. For more information on the AWS Quick Start deployment, visit the Quick Start site.

Pro Tip: For data protection purposes with the AWS solution, it is imperative that your VPC has two private subnets in different AWS AZs. Also, the data replica parameter is set to two (default) or more and should not be lowered if data protection is a concern. It also is important for data protection purposes that the cluster not run in a degraded state for any length of time if data is lost in a single AZ. Prioritizing the health of the cluster is critical to running Spectrum Scale on the AWS Cloud.

For HPC workloads, Spectrum Scale fits when many different compute hosts must have access to the same data. This is the fundamental benefit provided by a single namespace offered by Spectrum Scale. With a single namespace, a file can be addressed by a uniform file path, regardless of where the file is located, or from where the file is accessed in the cluster.

The single namespace also can be extended, including from on premises, with optional add-on features such as the data tiering feature Transparent Cloud Tiering (TCT) and universal client support to the filesystem. With Universal Client support, Spectrum Scale runs on Linux, AIX, and Windows clients to allow native file system access. Also, when export services enabled for a Spectrum Scale cluster, clients can access data using NFS, SMB, and Amazon S3 API. Spectrum Scale also provides POSIX and HDFS.

When to use: Spectrum Scale should be used in hybrid workloads where Spectrum Scale is being used on premises. Utilizing the cloud tiering feature as part of a single namespace makes bursting of HPC clusters simple. Additionally, because Spectrum Scale requires advanced configuration to set up and maintain, it is ideal for existing customers with expertise in this configuration. Spectrum Scale also is good when customers need flexibility of configuration, because it can scale on different dimensions.

For more information and to get started with spectrum scale, see the IBM Spectrum Scale on AWS Quick Start.[16]

Lustre

Lustre is an open source file system originally developed by Carnegie Mellon University and funded by the United States Department of Energy. Lead development for the platform has changed multiple times over the years. Most recently, Lustre was owned by Intel and was acquired by DataDirect Networks (DDN) in June 2018. Lustre often is used as part of HPC workloads because it is a parallel distributed file system that can provide a POSIX compliant file system with scalable performance and capacity. Commercial technical support for Lustre is often is bundled along with the computing system or storage hardware.

Architecture

Lustre uses external metadata, but it is object based not block based. A Lustre file system comprises several components, which include Metadata Servers (MDS), Metadata Targets (MDT), Object Storage Servers (OSS), Object Storage Targets (OST), and the Lustre clients. The MDS and OSS provide the access layer for the metadata and data, respectively. In kind, the MDT and OST provide the actual data storage. Because Lustre provides a file system to the client but is object based, files are ultimately stored in one or more objects, and the objects are stored on an OST. The Lustre client software plays an important role as well. The file system is accessed via the client software and not via a

network protocol like NFS/SMB. The client handles providing a single namespace for data spread across multiple OSTs, and the file system can handle multiple clients reading or writing to different parts of the same file at the same time.

On AWS, Lustre is generally deployed on Amazon EC2 instances and uses Amazon EBS for the underlying disk storage. The size of instances, as well as the type of Amazon EBS volumes, should be taken into account when designing a solution to meet performance requirements for the HPC cluster. Amazon EBS has four different volume types (details on the EBS website), which include a variety of performance options. You should take into account AZs when building a Lustre solution on AWS to meet your performance and availability requirements.

When to use: Lustre is a good solution for high-speed parallel HPC workloads. Since Lustre has no built-in data protection mechanism, precautions should be taken to persist data in Amazon S3. Monitoring for and addressing failed componentry are important. Lustre requires advanced configuration and design for optimal performance. Lustre is good for existing Lustre customers, who also have experience with AWS compute and storage services to optimize the cluster properly. For more information about Lustre visit lustre.org[17]

BeeGFS

BeeGFS, developed by ThinkParQ, is a parallel cluster file system that runs on Linux. BeeGFS is an open source, software defined storage system that runs on Linux distributions. BeeGFS provides a Linux-based client to allow parallel access to the file system.

Architecture

BeeGFS, similar to other parallel file systems, uses external metadata. There are four services that make up BeeGFS: the management service, which holds information for and monitors the other services; the storage service that hosts the data; the metadata service that stores access permissions and striping information; and the client service, which handles mounting and accessing the data. The storage service works on any local POSIX file system. BeeGFS, shown below in Figure 5, uses all available random access memory (RAM) for cache, so it benefits from additional memory for read access to hot data.

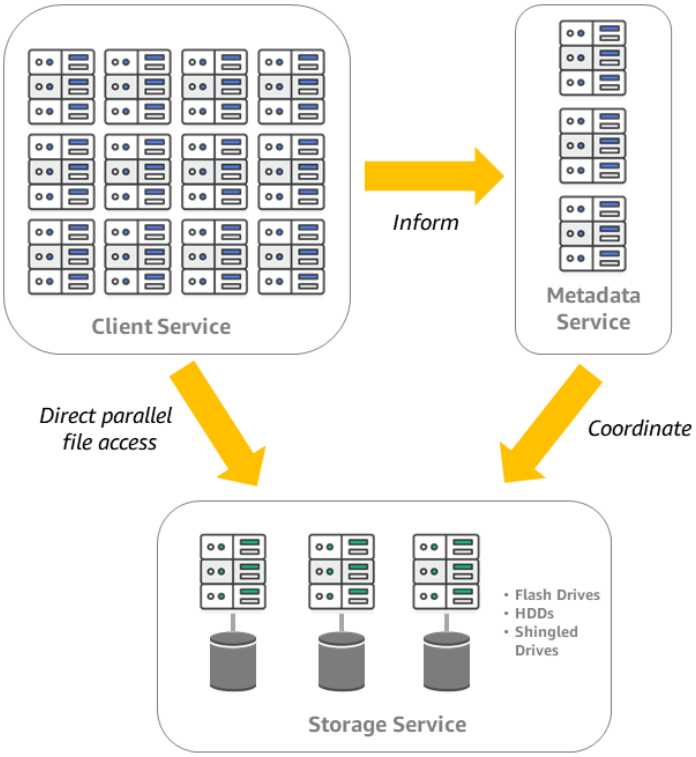

Figure 5: BeeGFS Architecture

When to use: BeeGFS, like several parallel file systems, offers a POSIX-compliant file system client. Since BeeGFS is free, open source software it can provide a cost-effective solution for customers who require high speed parallel file system solutions on AWS who typically utilize open source software as part of their HPC solutions.

For more information on BeeGFS see beegfs.io[18]. You also can find details on a building a scalable deep learning workload with BeeGFS on the AWS Blog[19].

Conclusion

There are many options for deploying HPC storage on AWS to support virtually any HPC workload. Each offering uses different AWS services to provide highly performant, scalable, and cost-optimized solutions. Native AWS services, as well as APN partner solutions, allow you to take advantage of the agility of AWS and deploy an HPC workload quickly in geographic locations around the world, including locations specifically designed for the public sector. Receive benefits from bursting your HPC workloads from on-premises to running whole workloads on AWS.

Contributors

The following individuals and organizations contributed to this document:

- Henry Axelrod, Storage Partner Solutions Architect, AWS
- Jeff Bartley, Storage Solutions Architect, AWS
- Peter Kisich, Storage Partner Solutions Architect, AWS
- Geert Wenes, HPC Partner Solutions Architect, AWS
- Isaiah Weiner, Senior Manager, Solutions Architecture, AWS

Further Reading

For additional information, see the following:

- HPC on AWS[20]

Document Revisions

Date	Description
December 2018	First publication

Notes

[1] https://aws.amazon.com/hpc

[2] https://do.awsstatic.com/whitepapers/Intro_to_HPC_on_AWS.pdf

[3] https://aws.amazon.com/what-is-cloud-object-storage/

[4] https://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html

[5] https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSOptimized.html#ebs-optimization-support

[6] https://aws.amazon.com/partners/

[7] https://www.weka.io/

[8] https://start.weka.io

[9] https://aws.amazon.com/marketplace/pp/B074XFQH6F

[10] https://aws.amazon.com/route53/

[11] https://qumulo.com/discover/qf2-overview/

[12] https://qumulo.com/wp-content/uploads/2018/07/WP-Q152-QF2-Technical-Overview.pdf

[13] https://aws.amazon.com/marketplace/pp/B07CQCLTFJ

[14] https://aws.amazon.com/marketplace/pp/B07G2VS5WT

[15] https://aws.amazon.com/marketplace/pp/B07GWP6XM2

[16] https://aws.amazon.com/quickstart/architecture/ibm-spectrum-scale/

[17] http://lustre.org/

[18] https://www.beegfs.io

[19] https://aws.amazon.com/blogs/machine-learning/scalable-multi-node-deep-learning-training-using-gpus-in-the-aws-cloud/

[20] https://aws.amazon.com/hpc