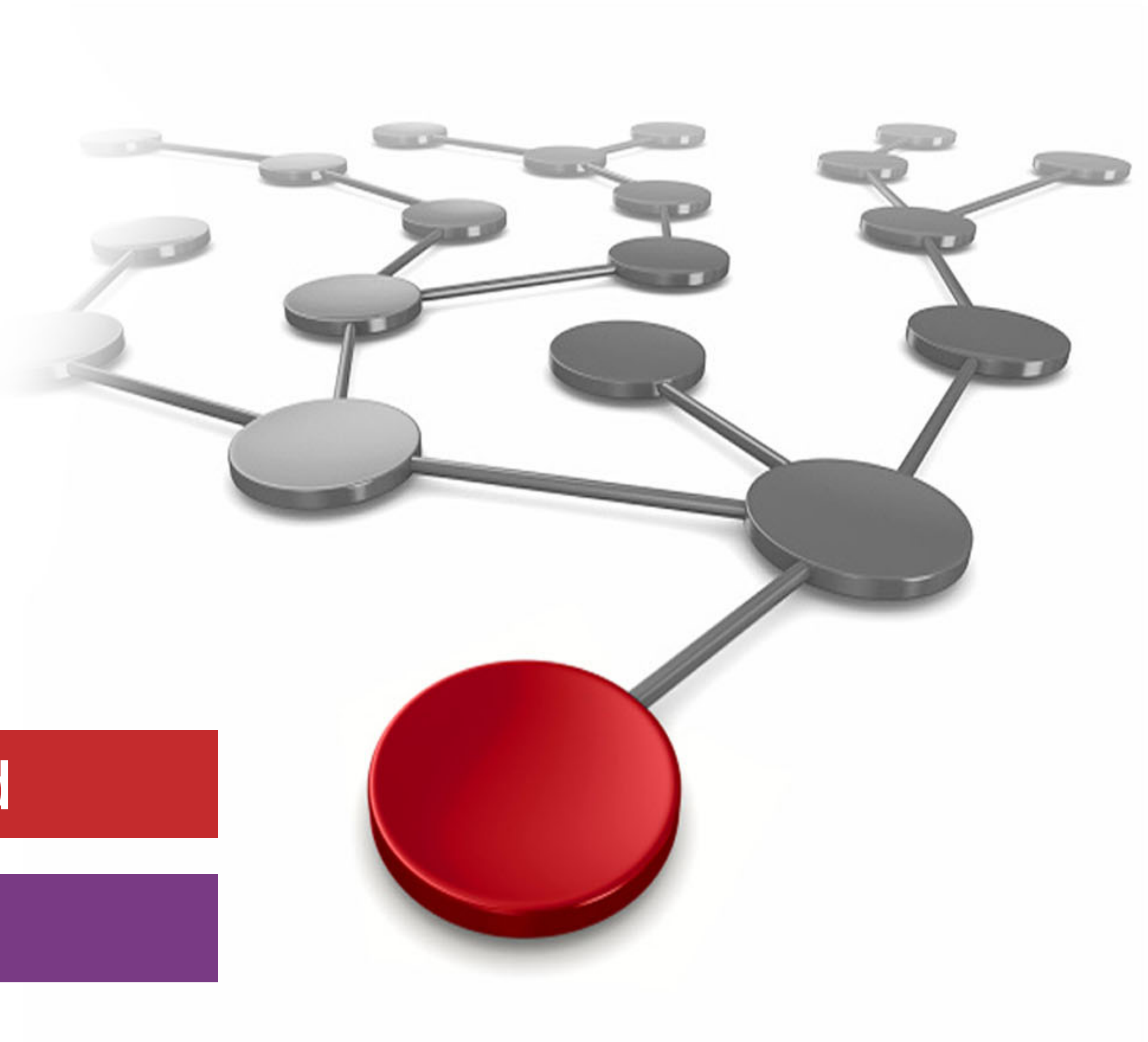
Redbooks
ibm.com/redbooks

High Performant File System for AI and HPC on Workloads on AWS using IBM Spectrum Scale

Sanjay Sudam



Cloud

Storage

IBM

Redpaper

Introduction

This IBM® Redpaper® publication is intended to facilitate the deployment and configuration of the IBM Spectrum® Scale based high-performance storage solutions for the scalable data and AI solutions on Amazon Web Services (AWS).

Configuration, testing results, and tuning guidelines for running the IBM Spectrum Scale based high-performance storage solutions for the data and AI workloads on AWS are the focus areas of the paper. The LAB Validation was conducted with the Red Hat Linux nodes to IBM Spectrum Scale by using the various Amazon Elastic Compute Cloud (EC2) instances. Simultaneous workloads are simulated across multiple Amazon EC2 nodes running with Red Hat Linux to determine scalability against the IBM Spectrum Scale clustered file system.

Solution architecture, configuration details, and performance tuning demonstrate how to maximize data and AI application performance with IBM Spectrum Scale on AWS.

Scope

Provide solutions architecture and related solutions configuration workflows with the following foundation components:

- IBM Spectrum Scale
- AWS components
- Hybrid cloud components

This paper does not replace any official manuals and documents that are issued by:

- IBM
- AWS

Prerequisites

It is assumed that users have a basic knowledge of the following topics:

- IBM Spectrum Scale
- AWS Cloud infrastructure
- Cloud technologies
- AWS Virtual Private Cloud (VPC) Network

Solution architecture and components

This solution provides guidance about building an enterprise-grade storage platform by using IBM Spectrum Scale required for data and AI workloads. It covers the benefits of the solution and provides guidance about the types of deployment models and considerations during the implementation of these different storage options on AWS.

IBM Spectrum Scale on AWS

IBM Spectrum Scale is a flexible and scalable software-defined file storage suitable for, but not limited to, analytics workloads. Enterprises around the globe deployed IBM Spectrum Scale to form large data lakes and content repositories to perform high-performance computing (HPC) and analytics workloads.

IBM Spectrum Scale architecture is based on flexible storage building blocks that allow you to easily deploy and expand your IBM Spectrum Scale environment. It can scale both performance and capacity without bottlenecks. IBM Spectrum Scale provides various configuration options and access methods through the client. These options include traditional POSIX-based file access with features such as snapshots, compression, and encryption.

IBM Spectrum Scale on AWS is a software-defined storage offering that is available through the AWS marketplace and deployed by using the Amazon EC2 instances and Elastic Block Storage (EBS) volumes.

AWS components

AWS Compute Services - Amazon EC2 instances

Amazon EC2 provides various instance types that are optimized to fit different use cases. Instance types are comprised of varying combinations of CPU, memory, storage, and networking capacity, which gives the flexibility to choose the suitable mix of resources for your applications. Each instance type includes one or more instance sizes, which allows you to scale your resources to the requirements of your target workload.

For more information about various Amazon EC2 instance configuration types available on AWS, see Amazon EC2 Instance Types.

AWS Storage Services

AWS provides several storage choices and can be configured easily per workload characteristics.

Amazon EC2 Instance Store

This storage is on disks that are physically attached to the host computer. An instance store provides temporary block-level storage for your instance, and high-speed local disk storage that can serve custom-built solutions for processing data that does not need to be persisted across the power recycles.

Amazon EBS volumes

Amazon Elastic Block Store (EBS) provides persistent block storage volumes for use with Amazon EC2 instances in the AWS Cloud. Amazon EBS volumes offer the consistent and low-latency performance needed to run your workloads.

Amazon EBS includes different performance characteristics, such as IOPS optimized Provisioned IOPSSSD (i01) volumes where customers can pre-provision the I/O they need, and Throughput Optimized HDD (st1) volumes that are designed to be used with workflows that are driven more by throughput than IOPS.

Amazon S3

Amazon S3 is a highly scalable and high durable storage platform with an object interface. Amazon S3 delivers 99.999999999% durability, and stores data for millions of applications. 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 solutions.

Security groups

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. When you launch an instance in a VPC, you can assign up to five security groups to the instance. Security groups act at the instance level, not the subnet level. Therefore, each instance in a subnet in your VPC can be assigned to a different set of security groups.

Deployment of the IBM Spectrum Scale on AWS

IBM Spectrum Scale can be deployed by using one of the following methods per user requirements:

- Marketplace deployment is available from the AWS console.

  Use this option to deploy the IBM Spectrum Scale cluster into a virtual private cloud (VPC) or a new VPC on AWS (see Figure 1).

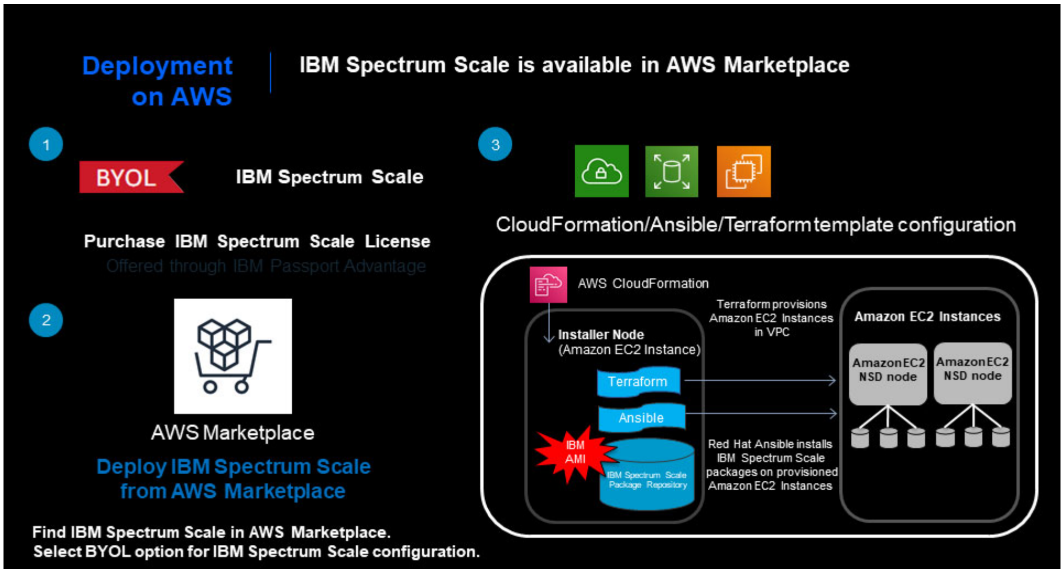

Deployment on AWS
IBM Spectrum Scale is available in AWS Marketplace
1
BYOL
IBM Spectrum Scale
Purchase IBM Spectrum Scale License
Offered through IBM Passport Advantage
2
AWS Marketplace
Deploy IBM Spectrum Scale from AWS Marketplace
Find IBM Spectrum Scale in AWS Marketplace.
Select BYOL option for IBM Spectrum Scale configuration.
3
CloudFormation/Ansible/Terraform template configuration
AWS CloudFormation
Installer Node
(Amazon EC2 Instance)
Terraform
Ansible
IBM AMI
IBM Spectrum Scale Package Repository
Terraform provisions Amazon EC2 Instances in VPC
Red Hat Ansible installs IBM Spectrum Scale packages on provisioned Amazon EC2 Instances
Amazon EC2 Instances
Amazon EC2 NSD node
Amazon EC2 NSD node

Figure 1 IBM Spectrum Scale deployment by way of AWS Marketplace

  The deployment process includes:

  - Bring Your Own License (BYOL): IBM Spectrum Scale follows the BYOL model for deployment in the AWS cloud. Customers can purchase the IBM Spectrum Scale license by using IBM Passport Advantage® or from Business Partners.
  - Automated Deployment using the AWS marketplace: IBM Spectrum Scale is available in the AWS marketplace and deployment is automated by using the Cloud Formation template leveraging the Terraform scripts and Ansible playbooks.

    For more information about deployment instructions, see IBM Spectrum Scale on AWS.

- Ansible Playbooks/Terraform Script Deployment

  IBM Spectrum Scale deployment can be automated using the Terraform scripts and Ansible playbooks. An installer node is required for this deployment method and can be configured on AWS or on-premises.

  Deployment process consists of:

  - Terraform scripts to provision the AWS cloud infrastructure resources required for the IBM Spectrum Scale deployment.
  - Ansible playbooks to automate the deployment of the IBM Spectrum Scale cluster on the AWS cloud.

    IBM open sourced the Terraform scripts and Ansible playbooks for community usage that can be accessible from the GitHub repository.

    The latest Terraform repository is available at IBM Spectrum Scale Cloud Install.

    The latest Ansible playbooks are available at IBM Spectrum Scale Install Infra.

Setting up installer node

Standard Amazon EC2 instance running with the Red Hat Linux operating system can be configured as the installer node for running the Terraform scripts and Ansible playbooks.

Complete the following steps to install and configure the prerequisites on the installer node:

1. Install the Terraform package and verify the installation:

   ```
   [ec2-user@ip-172-31-92-25 ~]$ sudo /usr/local/bin/terraform -v
   Terraform v0.13.4
   [ec2-user@ip-172-31-92-25 ~]$
   ```

2. Install Ansible and verify the package:

   ```
   $ curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
   $ python get-pip.py --user
   $ pip install ansible==2.9
   [ec2-user@ip-172-31-92-25 ~]$ ansible --version
   ansible 2.10.2
   ```

3. Install the AWS CLI on the node.
4. Configure the AWS CLI with the required credentials, so the node can perform the AWS resource provisioning.
5. Clone the GitHub Terraform repository on to the node:

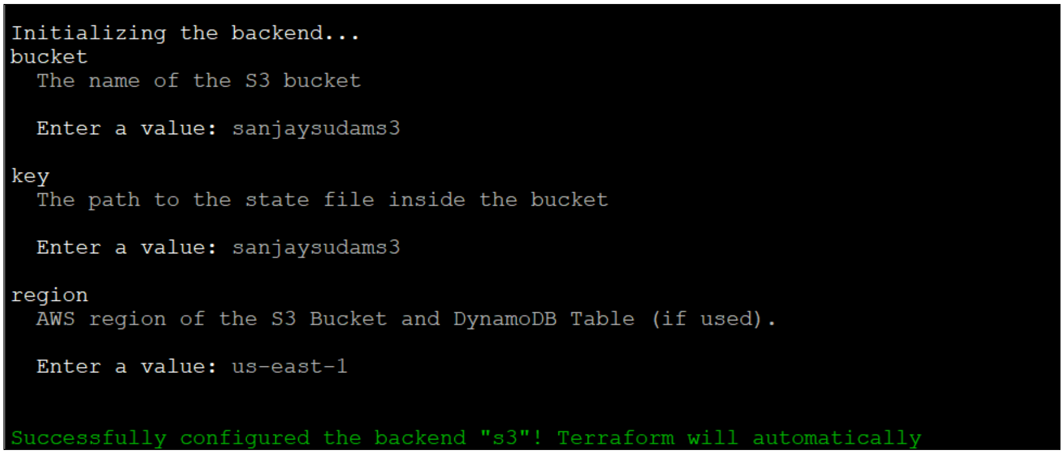
   ```
   [ec2-user@ip-172-31-92-25 ~]$ git clone https://github.com/IBM/
   Cloning into ''...
   remote: Enumerating objects: 284, done.
   remote: Counting objects: 100% (284/284), done.
   remote: Compressing objects: 100% (191/191), done.
   ```

```
remote: Total 1142 (delta 110), reused 190 (delta 62), pack-reused 858
Receiving objects: 100% (1142/1142), 6.68 MiB | 67.68 MiB/s, done.
Resolving deltas: 100% (464/464), done.
[ec2-user@ip-172-31-92-25 ~]$
```

6. Change the directory to the: `/aws_scale_templates/sub_modules/instance_template`, run the **`terraform init`** command, and provide the required details:

   ```
   [ec2-user@ip-172-31-92-25 instance_template]$ terraform init
   ```

   Figure 2 shows the initialization of the terraform environment that is required for the AWS configuration.



```
Initializing the backend...
bucket
  The name of the S3 bucket

  Enter a value: sanjaysudams3

key
  The path to the state file inside the bucket

  Enter a value: sanjaysudams3

region
  AWS region of the S3 Bucket and DynamoDB Table (if used).

  Enter a value: us-east-1


Successfully configured the backend "s3"! Terraform will automatically
```

Figure 2 terraform initialization on the deployment node

7. Clone the Ansible playbooks repository from GitHub, following IBM Spectrum Scale Install Infra:

   ```
   [ec2-user@ip-172-31-92-25 ~]$ git clone
   https://github.com/IBM/ibm-spectrum-scale-install-infra
   Cloning into 'ibm-spectrum-scale-install-infra'...
   remote: Enumerating objects: 300, done.
   remote: Counting objects: 100% (300/300), done.
   remote: Compressing objects: 100% (209/209), done.
   remote: Total 2141 (delta 147), reused 183 (delta 86), pack-reused 1841
   Receiving objects: 100% (2141/2141), 462.56 KiB | 24.34 MiB/s, done.
   Resolving deltas: 100% (1076/1076), done.
   [ec2-user@ip-172-31-92-25 ~]$ ll
   ```

8. Configure the json configuration file with the required parameters to provision AWS resources and configure the IBM Spectrum Scale Storage Cluster.

   The following sample configuration file includes four nodes that are configured as the storage nodes and another four nodes as the compute nodes:

   ```
   [root@ip-172-31-88-142 instance_template]# cat
   aws_existing_vpc_scale_inputs.auto.tfvars.json
   {
       "region": "us-east-1",
       "availability_zones": ["us-east-1a"],
       "bastion_sec_group_id": "sg-070dedf530db4f409",
       "private_instance_subnet_ids": ["subnet-de178ab9"],
       "vpc_id": "vpc-94d37aee",
       "key_name": "sanjay_US_key",
       "create_scale_cluster": "true",
   ```

```
    "compute_ami_id": "ami-0d4e69506cfe84566",
    "storage_ami_id": "ami-0d4e69506cfe84566",
    "compute_instance_type": "c5n.4xlarge",
    "storage_instance_type": "i3en.24xlarge",
    "ebs_volume_size": "10",
    "ebs_volume_type": "gp2",
    "ebs_volumes_per_instance": "0",
    "total_compute_instances": "4",
    "total_storage_instances": "4",
    "operator_email": "abc@xyz.com"}
[root@ip-172-31-88-142 instance_template]#
```

For more information about Terraform variable definitions and syntax, see Github.

Customers with the valid license and support contract can download the IBM Spectrum software from Fix Central.

9. Copy the Spectrum Scale software to the repository location on the admin node.
10. Run the Terraform script to provision the AWS resources and deployment of the IBM Spectrum Scale cluster:

```
root@ip-172-31-88-142 instance_template]# terraform apply -auto-approve
var.bucket_name
  s3 bucket name to be used for backing up ansible inventory file.
Enter a value: sanjaysudams3
```

The IBM Spectrum Scale cluster is then created and deployed as per the json configuration file.

Configuring AWS CloudWatch

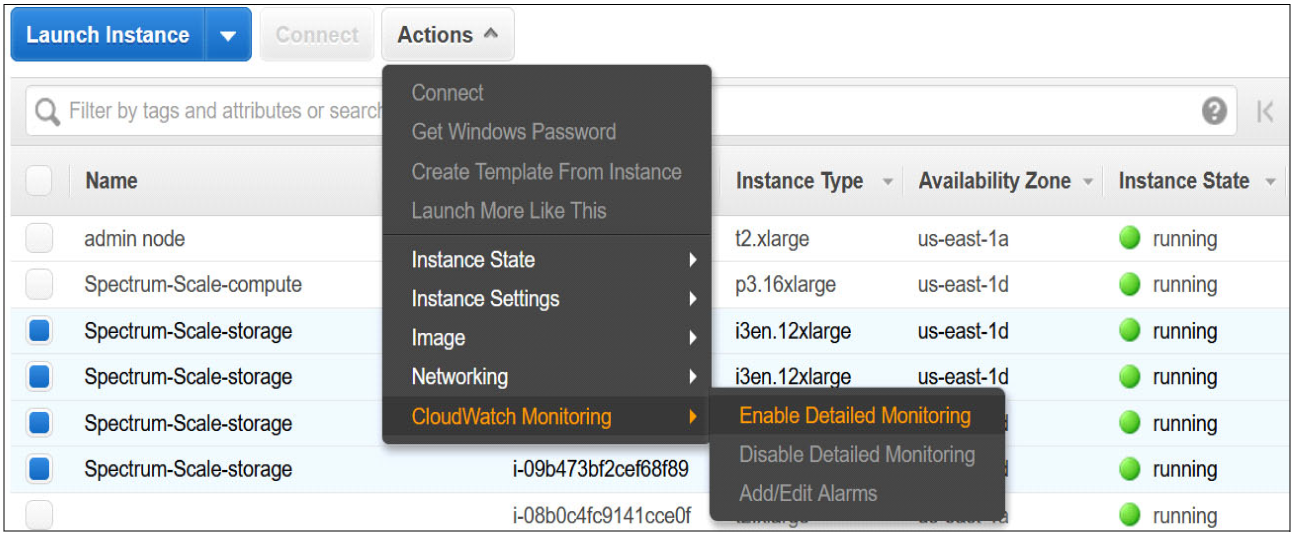
CloudWatch provides you with data and actionable insights to monitor your applications, respond to system-wide performance changes, optimize resource utilization, and get a unified view of operational health. CloudWatch collects monitoring and operational data in the form of logs, metrics, and events, which provide a unified view of AWS resources, applications, and services that run on AWS and on-premises servers. You can use CloudWatch to detect anomalous behavior in your environments, set alarms, visualize logs and metrics side by side, take automated actions, troubleshoot issues, and discover insights to keep your applications running smoothly.

Enabling AWS CloudWatch:

You can enable detailed monitoring on an instance as you start it or after the instance is running or stopped. Enabling detailed monitoring on an instance does not affect the monitoring of the EBS volumes that are attached to the instance.

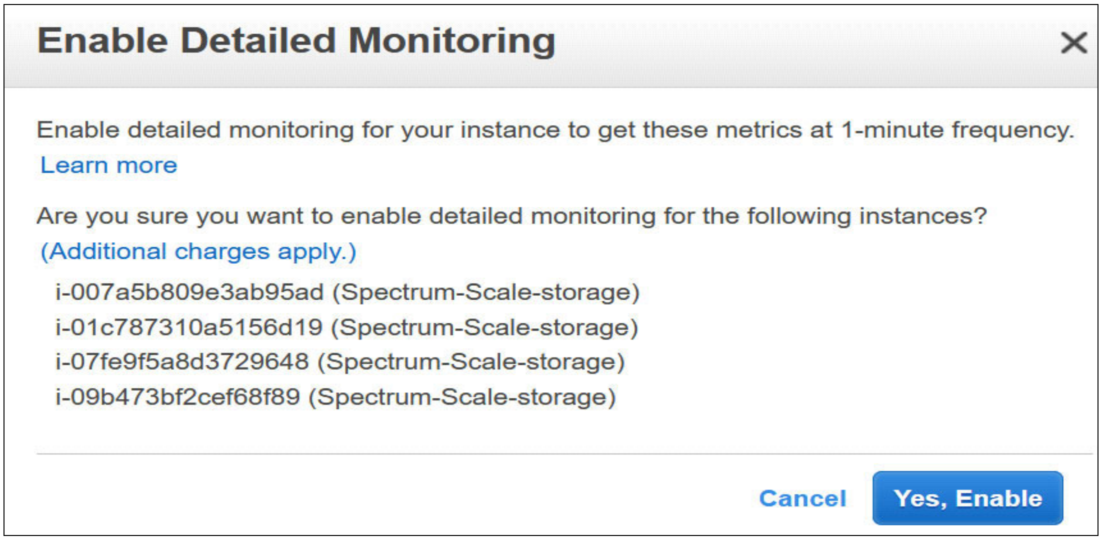
Complete the following steps to enable AWS CloudWatch:

1. Open the Amazon EC2 console.
2. In the navigation pane, choose **Instances**.
3. Select the IBM Spectrum Scale Storage **instance** → **Actions** → **Monitoring** → **Manage detailed monitoring**. See Figure 3 on page 8, Figure 4 on page 8, and Figure 5 on page 8 for selections and expected results.



	Name		Instance Type	Availability Zone	Instance State
☐	admin node		t2.xlarge	us-east-1a	● running
☐	Spectrum-Scale-compute		p3.16xlarge	us-east-1d	● running
☑	Spectrum-Scale-storage		i3en.12xlarge	us-east-1d	● running
☑	Spectrum-Scale-storage		i3en.12xlarge	us-east-1d	● running
☑	Spectrum-Scale-storage				● running
☑	Spectrum-Scale-storage	i-09b473bf2cef68f89			● running
☐		i-08b0c4fc9141cce0f			● running

Figure 3 AWS CloudWatch enablement



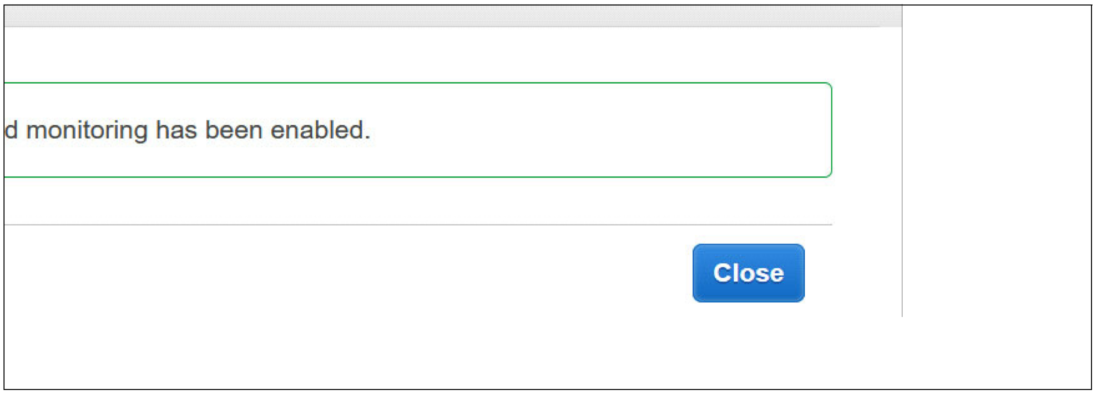
Enable Detailed Monitoring

Enable detailed monitoring for your instance to get these metrics at 1-minute frequency. Learn more

Are you sure you want to enable detailed monitoring for the following instances? (Additional charges apply.)

i-007a5b809e3ab95ad (Spectrum-Scale-storage)
i-01c787310a5156d19 (Spectrum-Scale-storage)
i-07fe9f5a8d3729648 (Spectrum-Scale-storage)
i-09b473bf2cef68f89 (Spectrum-Scale-storage)

Figure 4 AWS CloudWatch enablement for the IBM Spectrum Scale storage nodes



d monitoring has been enabled.

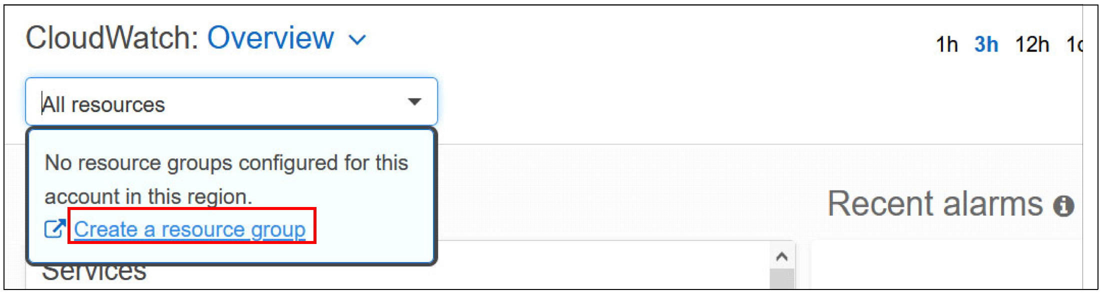
Figure 5 AWS CloudWatch enablement confirmation

For more information, see Enable or turn off detailed monitoring for your instances.

Creating resource groups

Complete the following steps to create a resource group:

1. Log into the CloudWatch dashboard console and select **Create a resource group** option to create a new resource group for the IBM Spectrum Scale storage nodes (see Figure 6).

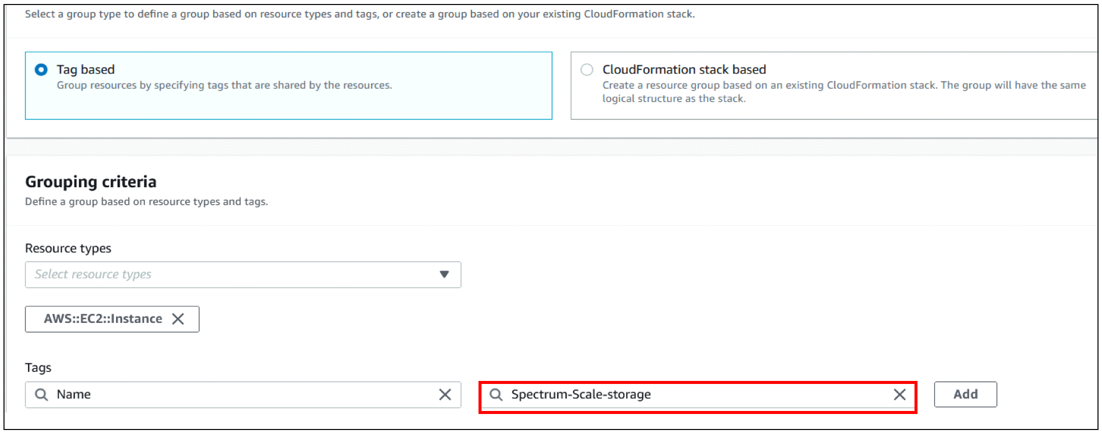


No resource groups configured for this account in this region.

Create a resource group

Figure 6 Creating a resource group option on CloudWatch dashboard console

2. Select the Amazon EC2 instance name tag **Spectrum-scale-storage** to create the Network Storage Disk (NSD) storage nodes that are configured at the IBM Spectrum Scale cluster (see Figure 7).



Select a group type to define a group based on resource types and tags, or create a group based on your existing CloudFormation stack.

● Tag based
Group resources by specifying tags that are shared by the resources.

○ CloudFormation stack based
Create a resource group based on an existing CloudFormation stack. The group will have the same logical structure as the stack.

Grouping criteria
Define a group based on resource types and tags.

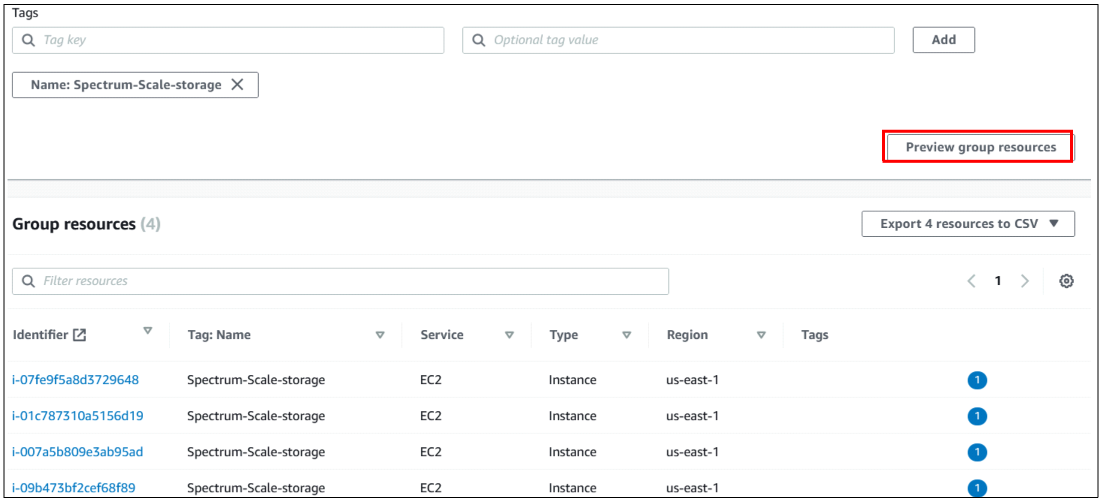
Resource types: Select resource types

AWS::EC2::Instance

Tags: Name — Spectrum-Scale-storage

Figure 7 Creating AWS resource group for the IBM Spectrum Scale storage nodes

3. Click **Preview group resources** to verify the filers (see Figure 8).



Tags: Tag key — Optional tag value

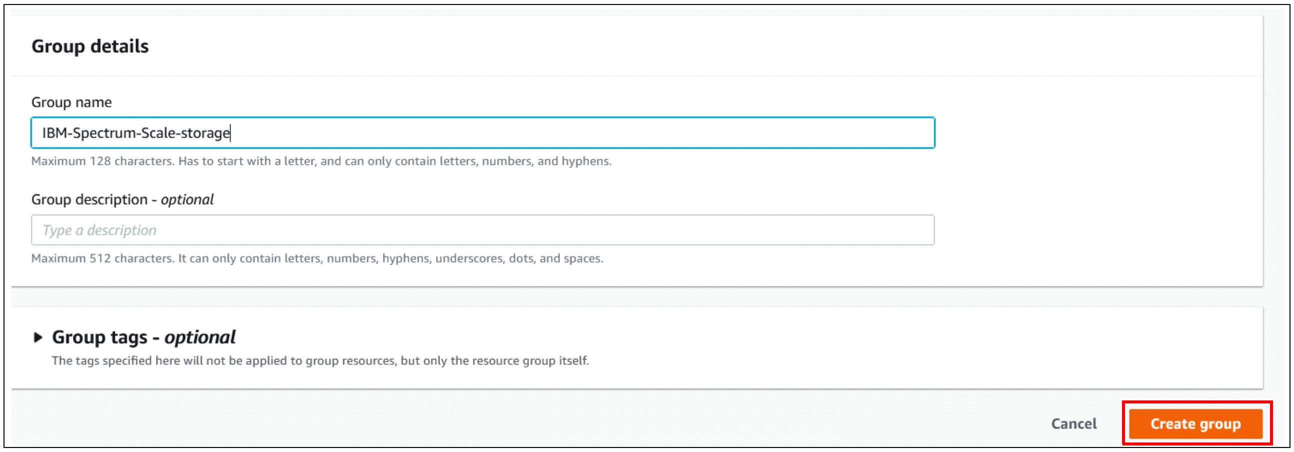
Name: Spectrum-Scale-storage

Group resources (4)

Identifier	Tag: Name	Service	Type	Region	Tags
i-07fe9f5a8d3729648	Spectrum-Scale-storage	EC2	Instance	us-east-1	1
i-01c787310a5156d19	Spectrum-Scale-storage	EC2	Instance	us-east-1	1
i-007a5b809e3ab95ad	Spectrum-Scale-storage	EC2	Instance	us-east-1	1
i-09b473bf2cef68f89	Spectrum-Scale-storage	EC2	Instance	us-east-1	1

Figure 8 Verifying the AWS resource groups

4. Select **Create group** to create the IBM-Spectrum-Scale-storage AWS resource group (see Figure 9).



Group details

Group name

IBM-Spectrum-Scale-storage

Maximum 128 characters. Has to start with a letter, and can only contain letters, numbers, and hyphens.

Group description - *optional*

Type a description

Maximum 512 characters. It can only contain letters, numbers, hyphens, underscores, dots, and spaces.

▸ **Group tags - *optional***

The tags specified here will not be applied to group resources, but only the resource group itself.

Figure 9 Creating the AWS resource groups

This allows filtering and monitoring IBM Spectrum Scale storage (NSD) nodes usage metrics from the CloudWatch dashboard (see Figure 10).

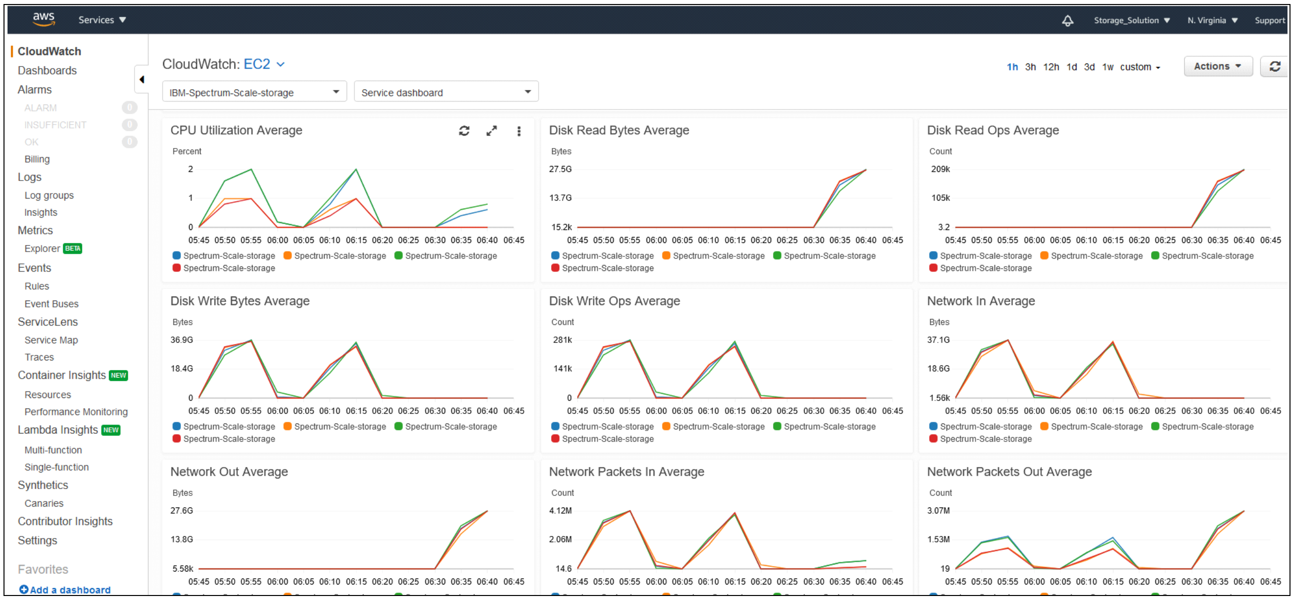


Figure 10 CloudWatch monitoring of the IBM Spectrum Scale resource group

LAB validation configuration

During the validation phase, the following types of instances are configured for the IBM Spectrum Scale storage nodes usage:

- Storage Optimized Instances

  In this configuration, I3 Amazon EC2 instance types with the local attached NVMe based instance storage volumes are configured as the IBM Spectrum Scale storage nodes. Local attached NVMe instance drives are configured as NSD drives and used for the IBM Spectrum Scale file system. This instance storage provides the high-performance throughput but will not be persistent.

  For more information, see Storage optimized instances.

- Compute Optimized instances

  In this configuration, compute optimize instances (c5/C5n) with the persistent storage volumes used for the IBM Spectrum Scale Storage (NSD) nodes. Compute optimized instances offers up to 100 Gbps network and EBS bandwidth of up to 19000 Mbps. These instances provide good storage throughput required for high-computing applications usage. These instances can be configured as the Compute nodes at the IBM Spectrum Scale cluster level in addition to the storage (NSD) nodes.

  For more information, see Amazon EC2 Instance Types.

- Accelerated computing instances

  These GPU-based high-performance instances are used for running accelerated workloads, such as Machine Learning, high-performance computing, computational fluid dynamics, where P3 instances are configured as the compute nodes (clients) at the IBM Spectrum Scale cluster level running the AI workloads. P3.16x large instances offers eight GPUs and 25 Gbps network throughput for running the Deep Learning and High-Performance Computing applications using the IBM Spectrum Scale file system.

  For more information, see Amazon EC2 Instance Types.

Performance file storage solution configuration based on instance storage

For the LAB validation purpose, high-performance storage optimized instances are configured as the storage nodes (see Figure 11).

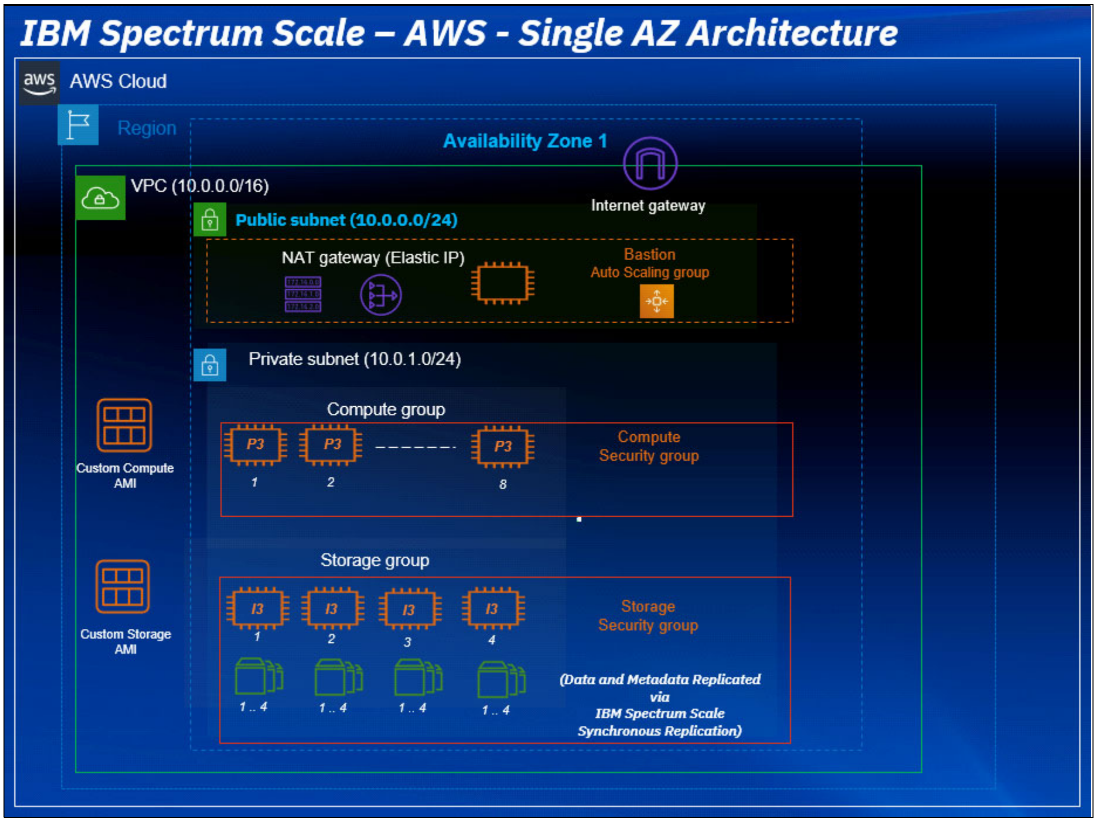


Figure 11 IBM Spectrum Scale Architecture based on I3 instances

IBM Spectrum Scale NSD (storage) nodes

i3en.12xlarge instance types are configured as the storage nodes (NSD) at the IBM Spectrum Scale cluster level. i3en instances are optimized for applications that require high random I/O access to large amounts of data and designed for data intensive workloads.

For more information, see Amazon EC2 I3en Instance.

Compute nodes

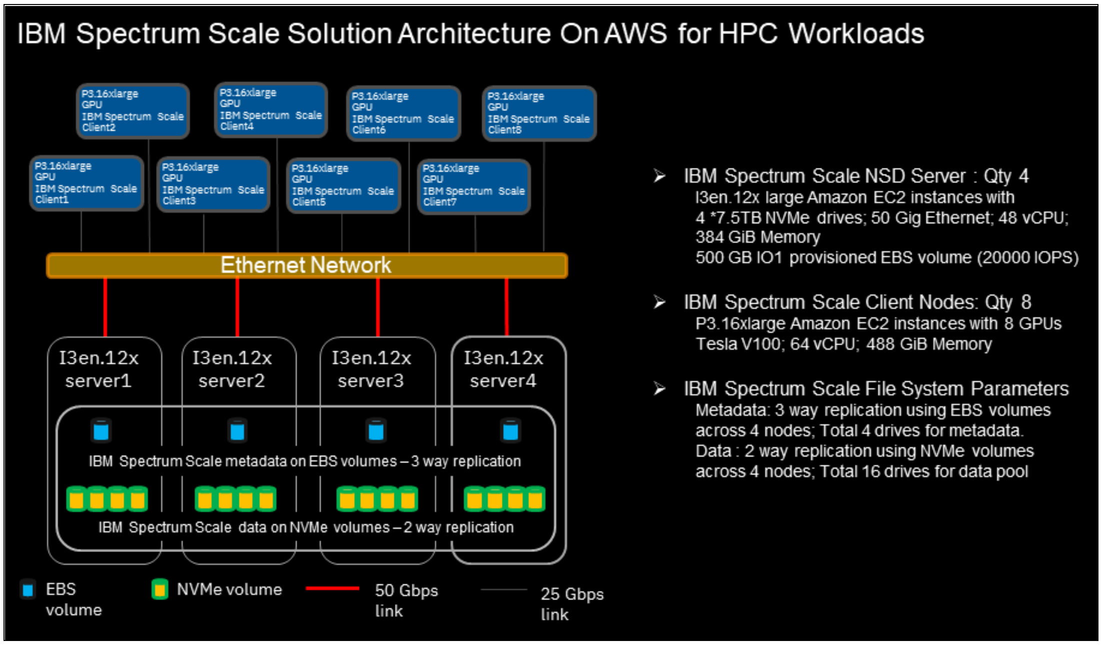
GPU-based P3 instances are configured as compute nodes.

Amazon EC2 P3 instances deliver high-performance computing in the cloud with up to eight NVIDIA V100 Tensor Core GPUs and up to 100 Gbps of networking throughput for machine learning and HPC applications.

For more information, see Amazon EC2 P3 Instance.

Each i3en.12xlarge is configured with 48 vCPU, 384 GB Memory, 4 x 7500 GB NVMe drives and 50 Gbps bandwidth. Each node is configured with an additional IO1 provisioned EBS volumes for the metadata usage at the IBM Spectrum Scale level. Three-way replication for the metadata that uses EBS volumes and two-way data replication that uses NVMe drives are configured at the IBM Spectrum Scale file system.

IBM Spectrum Scale architecture on AWS is based on the building block concept that allows to easily deploy and expand your IBM Spectrum Scale environment per performance and capacity requirements (see Figure 12).



IBM Spectrum Scale Solution Architecture On AWS for HPC Workloads

- IBM Spectrum Scale NSD Server : Qty 4
  I3en.12x large Amazon EC2 instances with
  4 *7.5TB NVMe drives; 50 Gig Ethernet; 48 vCPU;
  384 GiB Memory
  500 GB IO1 provisioned EBS volume (20000 IOPS)
- IBM Spectrum Scale Client Nodes: Qty 8
  P3.16xlarge Amazon EC2 instances with 8 GPUs
  Tesla V100; 64 vCPU; 488 GiB Memory
- IBM Spectrum Scale File System Parameters
  Metadata: 3 way replication using EBS volumes
  across 4 nodes; Total 4 drives for metadata.
  Data : 2 way replication using NVMe volumes
  across 4 nodes; Total 16 drives for data pool

P3.16xlarge GPU IBM Spectrum Scale Client2
P3.16xlarge GPU IBM Spectrum Scale Client4
P3.16xlarge GPU IBM Spectrum Scale Client6
P3.16xlarge GPU IBM Spectrum Scale Client8
P3.16xlarge GPU IBM Spectrum Scale Client1
P3.16xlarge GPU IBM Spectrum Scale Client3
P3.16xlarge GPU IBM Spectrum Scale Client5
P3.16xlarge GPU IBM Spectrum Scale Client7
Ethernet Network
I3en.12x server1
I3en.12x server2
I3en.12x server3
I3en.12x server4
IBM Spectrum Scale metadata on EBS volumes – 3 way replication
IBM Spectrum Scale data on NVMe volumes – 2 way replication
EBS volume
NVMe volume
50 Gbps link
25 Gbps link

Figure 12 I3 instance-based storage node building block on AWS

It is recommended to configure four nodes of i3en.12xlarge instance as the building block for the storage nodes (see Figure 12 on page 12), so that it can provide a reliable and performance-oriented solution that is based on the three-way metadata replication at the IBM Spectrum Scale cluster level. Each NSD node is configured as the separate failure group so that its data is replicated to different NSD nodes.

This architecture is flexible and provides an ability to expand the cluster in the block of four storage nodes, per performance requirements. Each additional building block provides a linear incremental in the data throughput and provides the flexibility to scale the solution per user requirements.

During the LAB validation, four storage nodes (i3en.12xlarge) and eight compute nodes (p3.16xlarge) are configured at the IBM Spectrum Scale cluster level for performance validation purpose. Industry-standard data simulation tools IOR and mpirun are used for the workload simulation on the IBM Spectrum Scale file system. Multiple tests with the different parameters are simulated to determine the IBM Spectrum Scale cluster performance characteristics on the AWS cloud.

Table 1 shows the Bill of Material that is used for the validation process on the AWS Cloud.

Table 1 Bill of Material used for the validation on the AWS Cloud

Role	Instance type	Configuration	Quantity

Storage nodes	I3en.12xlarge	48 vCPU; 384GB Memory;
		4x 7.5TB NVMe drives;
		50Gbps Ethernet
			4

Compute nodes	P3.16xlarge	64vCPU;488GB Memory;
		8 x Tesla V100 GPU;
		25Gbps Ethernet
			8

System throughput results with I3en.12xlarge storage nodes

The total IBM Spectrum Scale throughput configured with the four storage nodes shows the performance scales linearly for the one to eight P3 GPU compute systems (see Figure 13). The results demonstrate that the solution maximizes the potential throughput of the data infrastructure around 18 GBps read performance. Each building block of four storage nodes augments the read throughput by 18 GBps and performance scales linearly with each additional building storage unit.

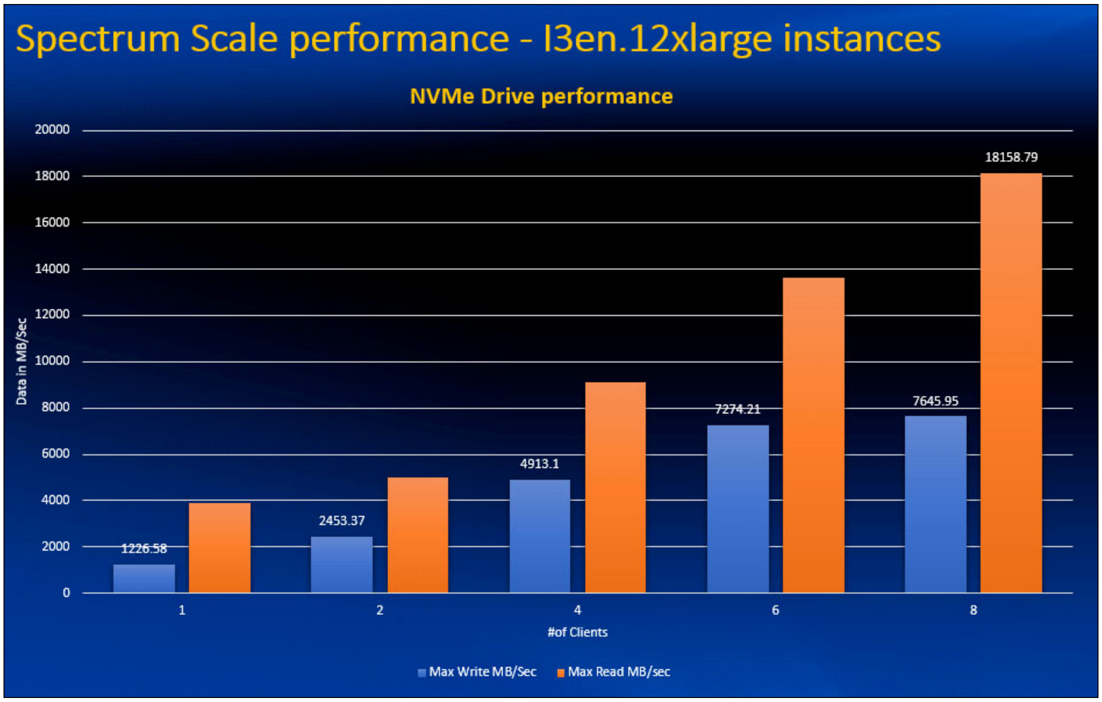


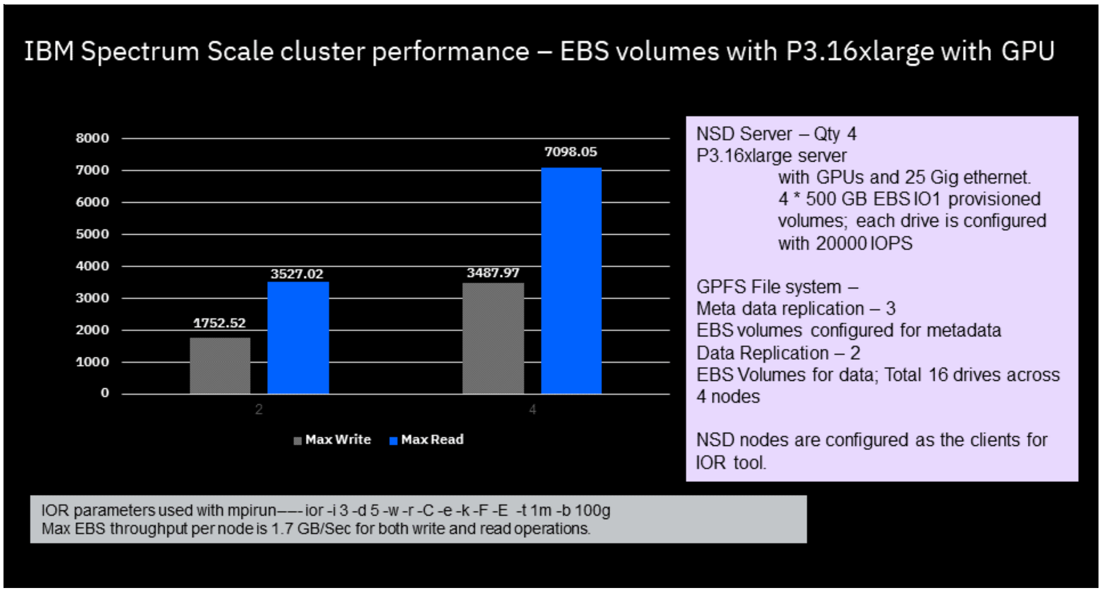
Figure 13 IBM Spectrum Scale performance with the I3 instances

EBS volumes for attached storage

GPU based instances with the EBS volumes are configured as storage and compute nodes during this validation. The primary objective of this configuration is to provide an ability to run the workloads directly on the storage nodes and provide an optimized solution in the AWS cloud.

During the LAB validation four P3.16xlarge instances are configured as storage and compute nodes to validate the solution. The total IBM Spectrum Scale throughput configured with the four storage nodes shows the performance scales linearly for the 1 - 4 P3 GPU compute systems (see Figure 14). The results demonstrate that the solution maximizes the potential throughput of the data infrastructure around 7 GBps read performance.

Each additional building block of four storage nodes augments the read throughput by 7 GBps and performance scales linearly with each additional building storage unit.


IBM Spectrum Scale cluster performance – EBS volumes with P3.16xlarge with GPU

NSD Server – Qty 4
P3.16xlarge server
with GPUs and 25 Gig ethernet.
4 * 500 GB EBS IO1 provisioned volumes; each drive is configured with 20000 IOPS

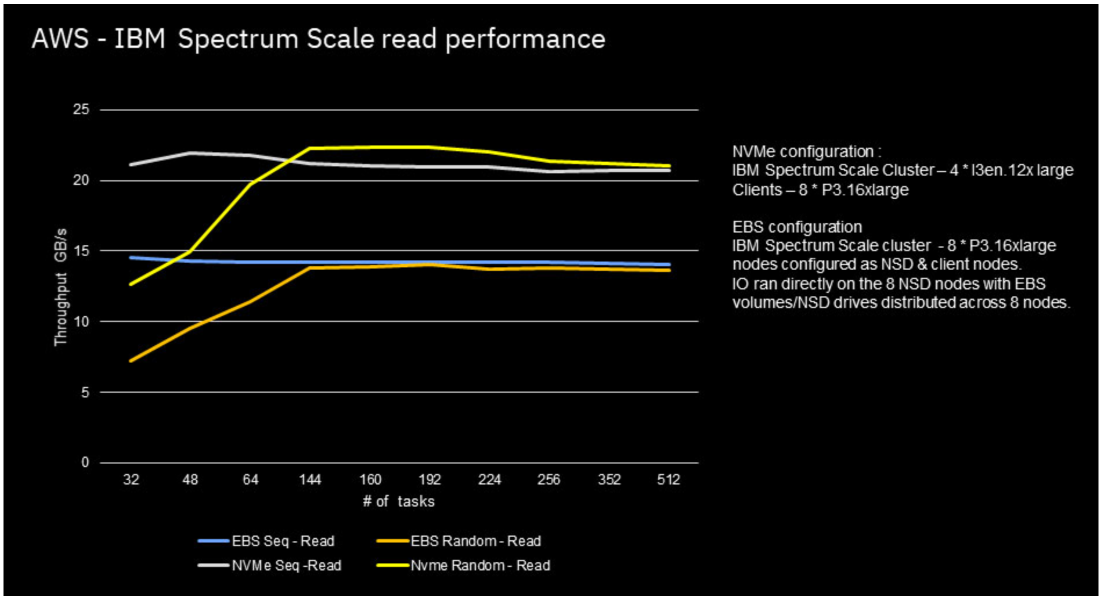
GPFS File system –
Meta data replication – 3
EBS volumes configured for metadata
Data Replication – 2
EBS Volumes for data; Total 16 drives across 4 nodes

NSD nodes are configured as the clients for IOR tool.

IOR parameters used with mpirun—-- ior -i 3 -d 5 -w -r -C -e -k -F -E -t 1m -b 100g
Max EBS throughput per node is 1.7 GB/Sec for both write and read operations.

Figure 14 IBM Spectrum Scale performance with the EBS volumes

To demonstrate the flexibility of the IBM Spectrum Scale storage solution, additional throughput tests were run for sequential versus random IO access patterns. Sequential read performance versus random read performance shows some prefetch advantage fades when the number of job threads increases. IBM Spectrum Scale shows robust throughput capabilities regardless of the I/O type (see Figure 15 on page 15).



NVMe configuration :
IBM Spectrum Scale Cluster – 4 * I3en.12x large
Clients – 8 * P3.16xlarge

EBS configuration
IBM Spectrum Scale cluster - 8 * P3.16xlarge
nodes configured as NSD & client nodes.
IO ran directly on the 8 NSD nodes with EBS
volumes/NSD drives distributed across 8 nodes.

Figure 15 IBM Spectrum Scale performance: Random Versus Sequential

Overall, the IBM Spectrum Scale throughput results show that the system makes full use of the high-performance NVMe drives and provides the scale solution for the AI workloads. This performance capability provides development teams the ability to add compute resources when needed knowing that the storage system performance can accommodate their increasing workload demands and, if needed, seamlessly add IBM Spectrum Scale storage units into the AI storage cluster.

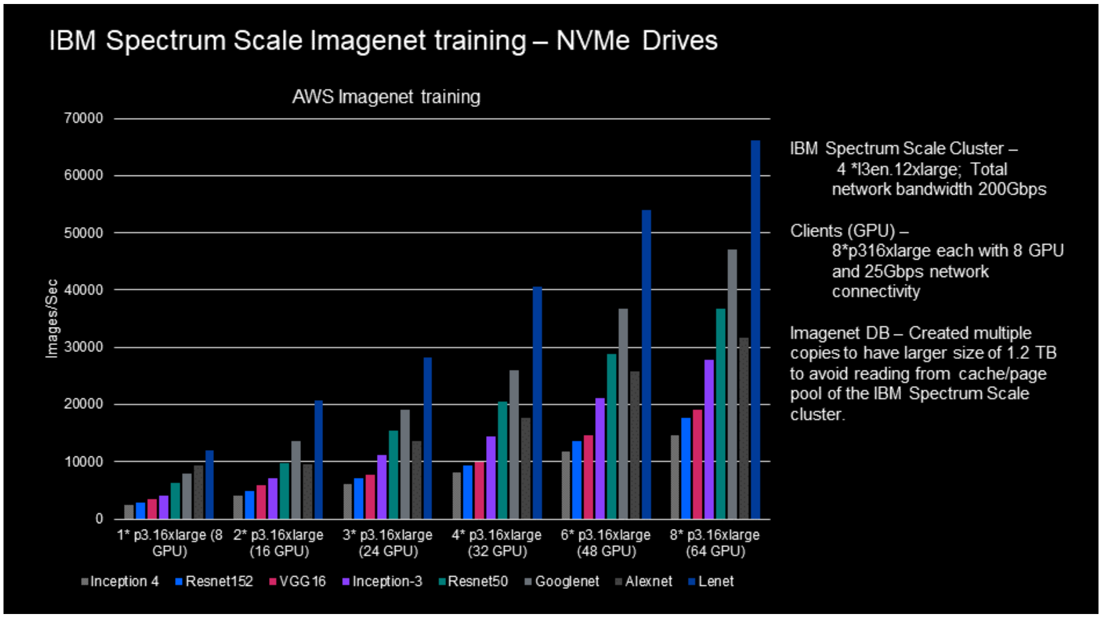
Training results: Instance-based NVMe storage volumes

Various convolutional neural network (CNN) models were tested with the IBM Spectrum Scale storage to study the effects of each model on performance during the training and inference. We tested each model with the ImageNet reference dataset. The ImageNet dataset is available at ImageNet, which includes detailed instructions about how to obtain the latest images. The dataset is approximately 140 GB and multiple copies of the data were created to simulate the dataset that is greater than 1 TB.

Figure 16 on page 16 shows the images per second training throughput with different CNN models using different numbers of GPUs configured across multiple P3.16xlarge instances on AWS cloud.

As shown, IBM Spectrum Scale system effectively feeds the system GPUs in the IBM Spectrum Scale cluster, keeping the GPU systems fully saturated with data for maximum training capabilities for all models.

Some models scale up with linearity as the number of GPUs increase while others present a consistent non-linear scale up pattern. This non-linear scalability in these cases is not constrained by storage I/O, but rather by a pattern of the DL model scalability within the compute infrastructure.


IBM Spectrum Scale Imagenet training – NVMe Drives

IBM Spectrum Scale Cluster –
4 *I3en.12xlarge; Total network bandwidth 200Gbps

Clients (GPU) –
8*p316xlarge each with 8 GPU and 25Gbps network connectivity

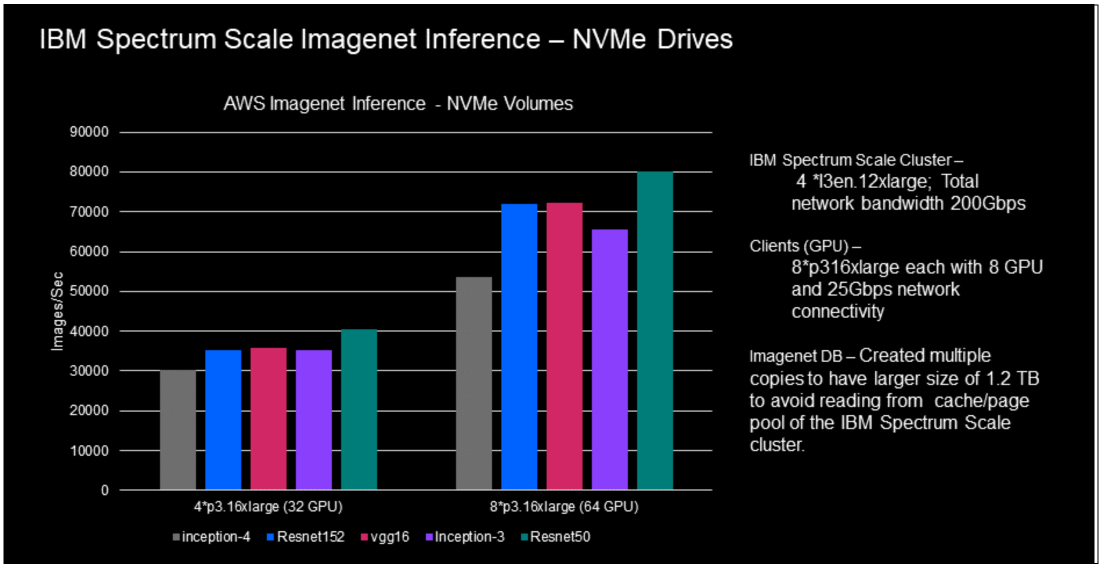
Imagenet DB – Created multiple copies to have larger size of 1.2 TB to avoid reading from cache/page pool of the IBM Spectrum Scale cluster.

Figure 16 IBM Spectrum Scale performance: ImageNet training with I3 instances

Inference results: Instance-based NVMe storage volumes

Inferencing is the process of deploying the trained model to assess a new set of objects and making predictions with a level of accuracy like that observed during the training phases. IBM Spectrum Scale storage service based that is on the NVMe instance storage was used to demonstrate the inferencing by using an ImageNet dataset on AWS Cloud by using the P3.16xlarge instance types.

Figure 17 shows the number of images that can be processed per second during inferencing. As tested, inference image processing rates scales with the additional GPU instances.


IBM Spectrum Scale Imagenet Inference – NVMe Drives

IBM Spectrum Scale Cluster –
4 *I3en.12xlarge; Total network bandwidth 200Gbps

Clients (GPU) –
8*p316xlarge each with 8 GPU and 25Gbps network connectivity

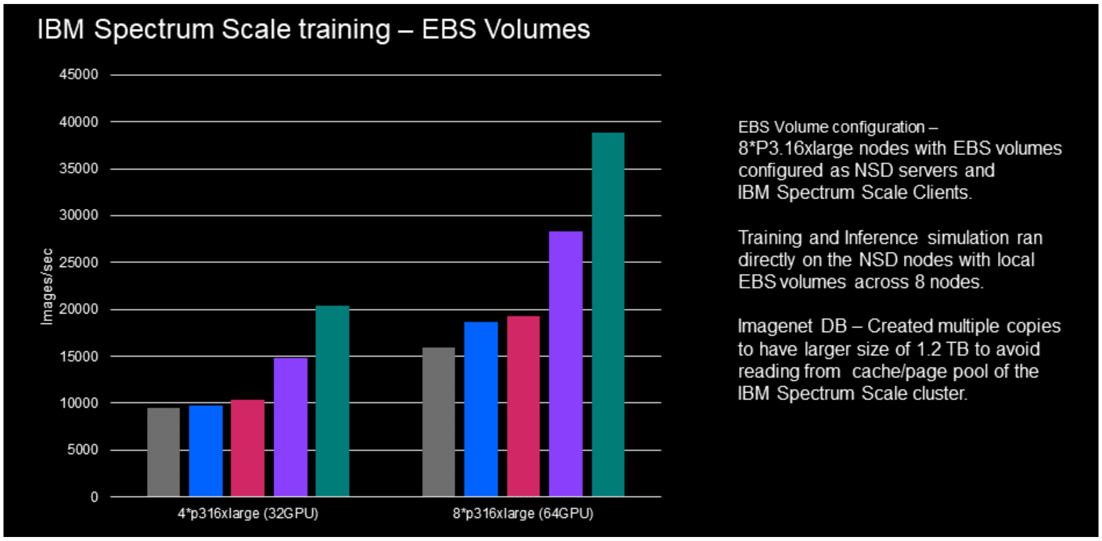
Imagenet DB – Created multiple copies to have larger size of 1.2 TB to avoid reading from cache/page pool of the IBM Spectrum Scale cluster.

Figure 17 IBM Spectrum Scale performance: ImageNet inference with I3 instances

Training results: EBS volumes based P3.16xlarge instances

Additional tests were simulated on the IBM Spectrum Scale cluster, configured with the EBS storage volumes. In this configuration, GPU instances are configured for both roles: storage and compute nodes. The same Amazon EC2 nodes were used for storage nodes, in addition to the AI workload simulation usage.

IBM Spectrum Scale storage service based on the EBS storage volumes was used to demonstrate the training models by using ImageNet dataset on AWS cloud. Figure 18 shows the images per second training throughput with different CNN models that uses the EBS storage volumes configured under the IBM Spectrum Scale cluster.


IBM Spectrum Scale training – EBS Volumes

EBS Volume configuration –
8*P3.16xlarge nodes with EBS volumes configured as NSD servers and IBM Spectrum Scale Clients.

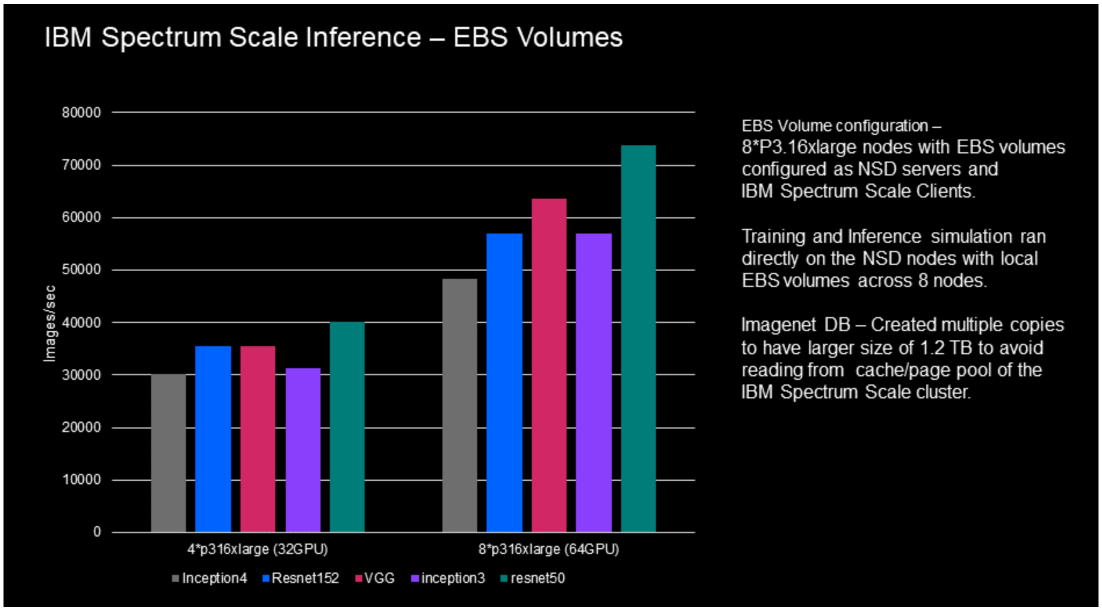
Training and Inference simulation ran directly on the NSD nodes with local EBS volumes across 8 nodes.

Imagenet DB – Created multiple copies to have larger size of 1.2 TB to avoid reading from cache/page pool of the IBM Spectrum Scale cluster.

Figure 18 IBM Spectrum Scale performance: ImageNet training with EBS volumes

Inference results: EBS volumes based P3.16xlarge instances

IBM Spectrum Scale storage service based on the EBS storage volumes was used to demonstrate the inferencing using ImageNet dataset on AWS cloud. Figure 19 on page 18, shows the images per second training throughput with different CNN models using the EBS storage volumes configured under the IBM Spectrum Scale cluster.



EBS Volume configuration –
8*P3.16xlarge nodes with EBS volumes configured as NSD servers and IBM Spectrum Scale Clients.

Training and Inference simulation ran directly on the NSD nodes with local EBS volumes across 8 nodes.

Imagenet DB – Created multiple copies to have larger size of 1.2 TB to avoid reading from cache/page pool of the IBM Spectrum Scale cluster.

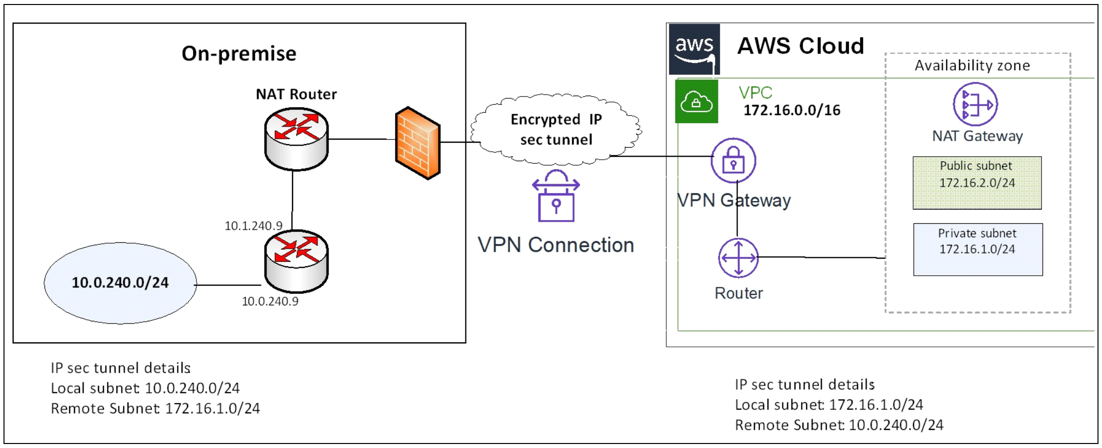
Figure 19 IBM Spectrum Scale performance - ImageNet inference with EBS volumes

Hybrid cloud configuration

Hybrid cloud is an environment that combines private and public cloud resources by allowing data and applications to be shared. This sharing requires configuring the network configuration and establishing connectivity between on-premises private cloud and public cloud resources. The most used solution is the public Internet for data transfer. However, this solution includes some privacy and security concerns about data flowing over the public internet. Users can make a secure connection by using an IPsec based Virtual Private Network (VPN) between private and public cloud networks. This connection encrypts the data and provides a point-to-point tunnel between private and public networking devices. The performance is dependent on the network link speed of the connection.

Site-to-site VPN connection between on-premises and AWS Cloud

Figure 20 shows the site-to-site VPN connectivity between the on-premises private cloud and the AWS hybrid cloud that was used for solution validation in the LAB.



IP sec tunnel details
Local subnet: 10.0.240.0/24
Remote Subnet: 172.16.1.0/24

IP sec tunnel details
Local subnet: 172.16.1.0/24
Remote Subnet: 10.0.240.0/24

Figure 20 Hybrid Cloud connectivity

IBM Spectrum Scale hybrid cloud architecture

IBM Spectrum Scale Active File Management (AFM) is used for data movement and caching between the on-premises and AWS cloud. AWS cloud end points are supported as cache sites only.

IBM AFM is a scalable, high-performance, intelligent file system caching layer integrated into the IBM Spectrum Scale file system. This integration allows implementation of a single global name space across various sites, including the public cloud offerings. Consider the following additional opportunities:

- Enables data mobility and sharing of data across various clusters.
- Offer an asynchronous data, cross-cluster caching utility.
- Configures on-premises as home and acts as a primary storage.
- Provides public cloud endpoints (AWS) that are cache only.

IBM Spectrum Scale architecture provides the flexibility of extending the single global name space from the high-performance, on-premises data lake to the AWS Cloud (see Figure 21). NFS protocol is used for communication and data movement between home (on-premises) data lake to the AWS Cloud.

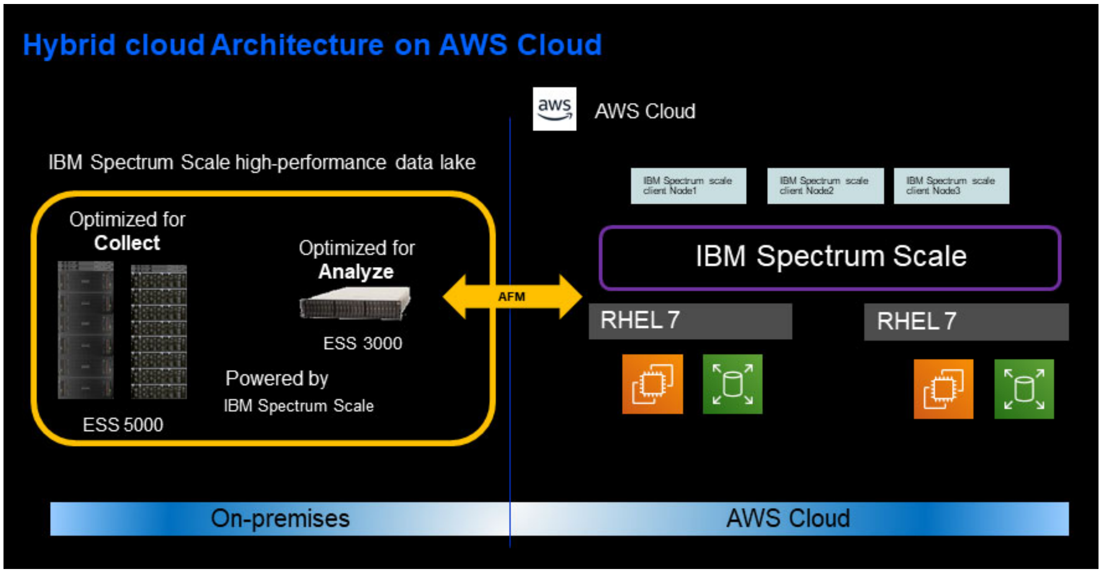

Hybrid cloud Architecture on AWS Cloud
aws
AWS Cloud
IBM Spectrum Scale high-performance data lake
Optimized for Collect
Optimized for Analyze
ESS 3000
Powered by IBM Spectrum Scale
ESS 5000
AFM
IBM Spectrum scale client Node1
IBM Spectrum scale client Node2
IBM Spectrum scale client Node3
IBM Spectrum Scale
RHEL 7
RHEL 7
On-premises
AWS Cloud

Figure 21 IBM Spectrum Scale hybrid cloud architecture

Implementing enterprise data pipeline in hybrid cloud environments

Enterprise Analytics and AI environments involve multiple AI and Analytics applications, however they share a common asset in the form of the enterprise data. These environments are designed and implemented as workflows that represent stages from data ingest to final insights. These implementations are generally referred to as *enterprise data pipelines*. Data flows through multiple stages of these data pipelines in which various applications access it for analysis/processing.

IBM has the distinctive architecture for the AI journey and involves integrating various components that are required for the AI journey in the private, public, and hybrid cloud environments. Now, with the wider adoption of hybrid cloud within enterprises, it is often a requirement to implement such an enterprise data pipeline in hybrid cloud setups where some of the stages are implemented on-premises and some are implemented on public cloud.

For example, an enterprise with an on-premises data lake in which Ingest, Organize, and Analyze stages are performed on-premises and want to enhance their analytics capabilities by augmenting it with AI. These additional AI capabilities require GPU-enabled compute to run model training on selected data sets from the data lake. AWS Cloud provides GPU-based instances and can be used for simulating the training models in AWS Cloud.

IBM Spectrum Scale provides a single global name space across the hybrid cloud and allows moving selective data sets from the on-premises data lake to AWS cloud per workload requirements. Results (trained model) can be shared back to the on-premises Hadoop data lake for further analysis usage.

The hybrid cloud data solution that uses IBM Spectrum Scale is an ideal option to implement the hybrid cloud data pipeline AI workflows. IBM Spectrum Scale AFM capabilities easily enables extending this implementation in hybrid cloud environments (see Figure 22).

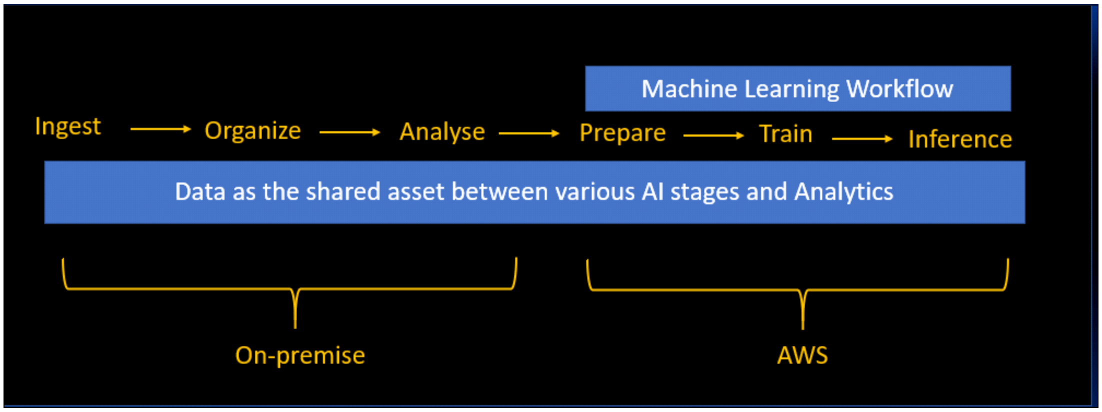

Machine Learning Workflow
Ingest
Organize
Analyse
Prepare
Train
Inference
Data as the shared asset between various AI stages and Analytics
On-premise
AWS

Figure 22 Hybrid cloud AI workflow

This solution enables on-demand caching of the required data set for machine learning workload, avoids creating multiple copies of the same data, and helps to reduce the total cost of the solution.

Additional Hybrid Cloud use-cases

In addition to the GPU based Analytics, IBM Spectrum Scale on AWS can be used for other HPC applications. These applications in the hybrid cloud requires an ability to move the data to the AWS cloud and provide the environment for data sharing and data simulation that is required in the HPC simulation.

IBM Spectrum Scale AFM has different configuration modes, which controls the data flow direction and helps in building the high-performance hybrid cloud data management platform required for the AI and HPC environments.

Summary

AWS cloud provides various high-performance computing and storage instance types and helps organizations to build a rich ecosystem to develop, deploy, and test AI/DL applications. GPUs are priced resources maintained at higher usage to reduce the cost of usage.

IBM Spectrum Scale provides high-performance parallel processing with high bandwidth and low latency for the full use of GPUs when running on multiple GPU systems. IBM Spectrum Scale prevents GPUs from waiting for data and helps in building the optimized data and AI solutions with the reduced costs.

The IBM technology feature enables to move the data in a secure and optimized way between on-premises and AWS cloud. The hybrid cloud data solution that uses IBM Spectrum Scale is the perfect option to implement the AI enterprise data pipeline in hybrid cloud setups, where some of the stages are implemented on-premises and others on public cloud.

Get more Information

For more information about the IBM and AWS products and capabilities, contact your IBM representative or IBM Business Partner, or see the following web pages:

- IBM Spectrum Scale on AWS
- AWS marketplace

Appendix

This section lists the sample configurations files and scripts in building the solution.

IBM Spectrum Scale deployment and configuration on the AWS cloud can be automated via Terraform scripts and Ansible playbooks per the parameters in the json configuration file.

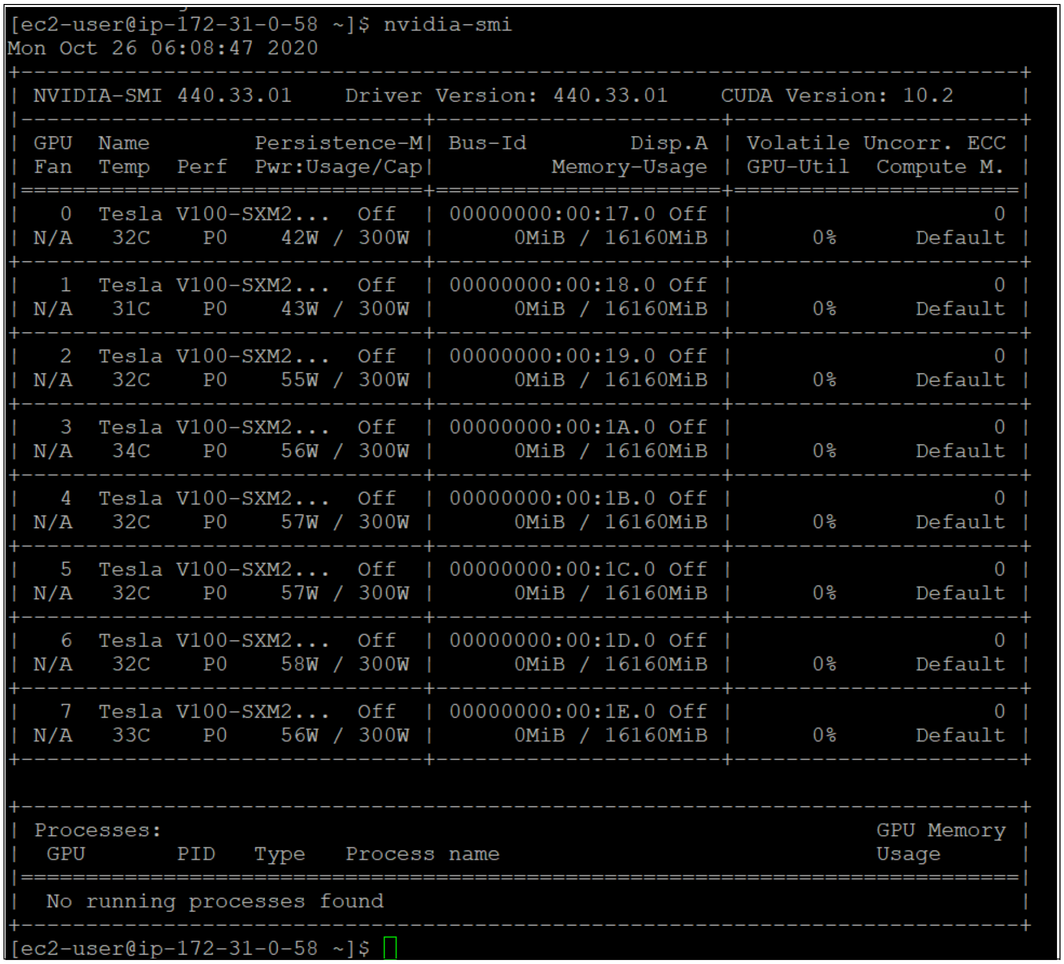
Here is the sample json configuration file for deploying the IBM Spectrum Scale in single Availability Zone (AZ) on AWS by using Terraform scripts and Ansible playbooks:

```
[root@ip-172-31-88-142 instance_template]# cat aws_existing_vpc_scale_inputs.auto.tfvars.json
{
    "region": "us-east-1",
    "availability_zones": ["us-east-1a"],
    "bastion_sec_group_id": "sg-070dedf530db4f409",
    "private_instance_subnet_ids": ["subnet-de178ab9"],
    "vpc_id": "vpc-94d37aee",
    "key_name": "sanjay_US_key",
    "create_scale_cluster": "true",
    "compute_ami_id": "ami-0d4e69506cfe84566",
    "storage_ami_id": "ami-0d4e69506cfe84566",
    "compute_instance_type": "c5n.4xlarge",
    "storage_instance_type": "i3en.24xlarge",
    "ebs_volume_size": "10",
    "ebs_volume_type": "gp2",
    "ebs_volumes_per_instance": "0",
    "total_compute_instances": "4",
    "total_storage_instances": "4",
    "operator_email": "abc@xyz.com"
}
```

Preparing the GPU Compute node

Complete the following steps to prepare the GPU Compute node for running Deep Learning training and inference analysis. P3.16xlarge GPU instances running with the Red Hat Enterprise Linux are used for simulating the training and inference.

1. Install and configure the NVIDIA CUDA tool kit for RHEL distribution, following NVIDIA CUDA Installation Guide for Linux instructions.
2. Verify the NVIDIA GPU configured on the node via nvidia-smi (see Figure 23 on page 22).



```
[ec2-user@ip-172-31-0-58 ~]$ nvidia-smi
Mon Oct 26 06:08:47 2020
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 440.33.01    Driver Version: 440.33.01    CUDA Version: 10.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla V100-SXM2...  Off  | 00000000:00:17.0 Off |                    0 |
| N/A   32C    P0    42W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   1  Tesla V100-SXM2...  Off  | 00000000:00:18.0 Off |                    0 |
| N/A   31C    P0    43W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   2  Tesla V100-SXM2...  Off  | 00000000:00:19.0 Off |                    0 |
| N/A   32C    P0    55W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   3  Tesla V100-SXM2...  Off  | 00000000:00:1A.0 Off |                    0 |
| N/A   34C    P0    56W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   4  Tesla V100-SXM2...  Off  | 00000000:00:1B.0 Off |                    0 |
| N/A   32C    P0    57W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   5  Tesla V100-SXM2...  Off  | 00000000:00:1C.0 Off |                    0 |
| N/A   32C    P0    57W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   6  Tesla V100-SXM2...  Off  | 00000000:00:1D.0 Off |                    0 |
| N/A   32C    P0    58W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   7  Tesla V100-SXM2...  Off  | 00000000:00:1E.0 Off |                    0 |
| N/A   33C    P0    56W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
[ec2-user@ip-172-31-0-58 ~]$
```

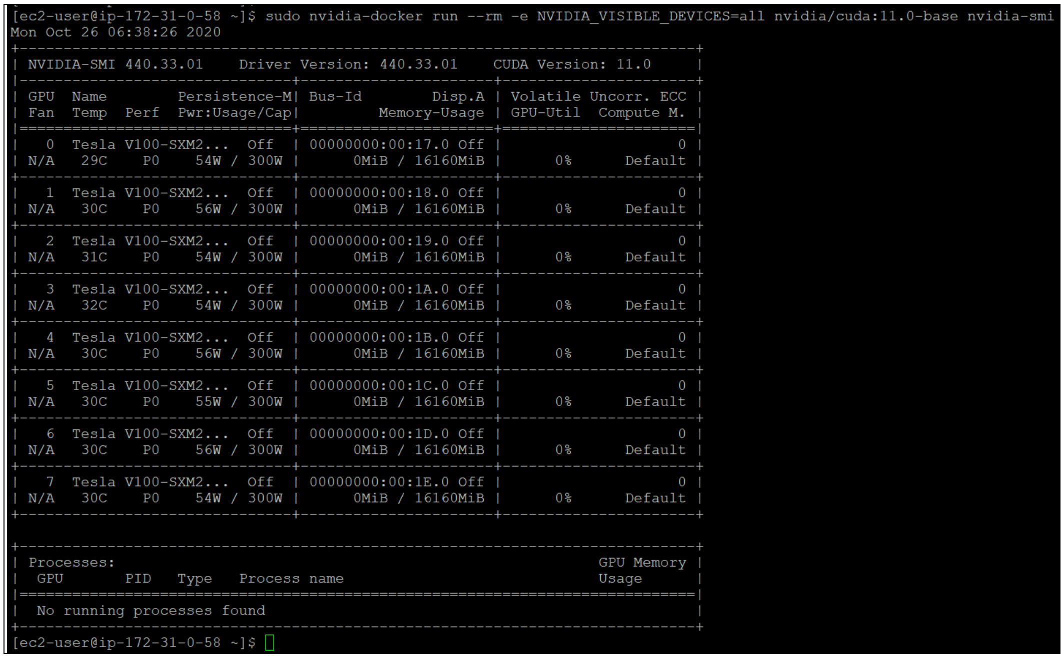
Figure 23 Verification of the GPU configuration on the nodes

3. Install the Docker engine and NVIDIA container tool kit on the RHEL, following Installing on RHEL 7 instructions.
4. Verify the Docker installation.

```
[ec2-user@ip-172-31-0-58 ~]$ sudo docker -v
Docker version 19.03.11, build 42e35e61f3
```

5. Install the nvidia-container toolkit on the RHEL node, following Installing on RHEL 7 instructions.
6. Verify the working CUDA container (see Figure 24 on page 23):

```
sudo nvidia-docker run --rm -e NVIDIA_VISIBLE_DEVICES=all nvidia/cuda:11.0-base
nvidia-smi
```



```
[ec2-user@ip-172-31-0-58 ~]$ sudo nvidia-docker run --rm -e NVIDIA_VISIBLE_DEVICES=all nvidia/cuda:11.0-base nvidia-smi
Mon Oct 26 06:38:26 2020
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 440.33.01    Driver Version: 440.33.01    CUDA Version: 11.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla V100-SXM2...  Off  | 00000000:00:17.0 Off |                    0 |
| N/A   29C    P0    54W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   1  Tesla V100-SXM2...  Off  | 00000000:00:18.0 Off |                    0 |
| N/A   30C    P0    56W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   2  Tesla V100-SXM2...  Off  | 00000000:00:19.0 Off |                    0 |
| N/A   31C    P0    54W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   3  Tesla V100-SXM2...  Off  | 00000000:00:1A.0 Off |                    0 |
| N/A   32C    P0    54W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   4  Tesla V100-SXM2...  Off  | 00000000:00:1B.0 Off |                    0 |
| N/A   30C    P0    56W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   5  Tesla V100-SXM2...  Off  | 00000000:00:1C.0 Off |                    0 |
| N/A   30C    P0    55W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   6  Tesla V100-SXM2...  Off  | 00000000:00:1D.0 Off |                    0 |
| N/A   30C    P0    56W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   7  Tesla V100-SXM2...  Off  | 00000000:00:1E.0 Off |                    0 |
| N/A   30C    P0    54W / 300W |      0MiB / 16160MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
[ec2-user@ip-172-31-0-58 ~]$
```

Figure 24 Verification of the CUDA driver on the node

7. Pull and deploy the tensorflow docker image from the NVIDIA NGC catalogue:

   ```
   [root@ip-172-31-4-241 ~]# nvidia-docker pull
   nvcr.io/nvidia/tensorflow:20.03-tf1-py3
   20.03-tf1-py3: Pulling from nvidia/tensorflow
   ```

8. Start the tensorflow Docker with an option to mount the IBM Spectrum Scale file system as the storage mount path for the container. This path is used for accessing the ImageNet datasets during the training and inference analysis:

   ```
   nvidia-docker run --shm-size=1g --ulimit memlock=-1 --ulimit stack=67108864 -v
   /ibm/gpfs0/:/ibm/gpfs0 --network=host -it
   nvcr.io/nvidia/tensorflow:20.03-tf1-py3 /bin/bash
   ```

   In this example, mount point `/ibm/gpfs0` contains the ImageNet data for the training and inference analysis.

9. The ImageNet dataset is available on the ImageNet website, which includes detailed instructions about how to obtain the latest image. The ImageNet dataset is approximately 140 GB and takes time to copy to the AWS Cloud environment. Copy this dataset to the IBM Spectrum Scale file system location for the training and inference analysis usage.

Configuring Tensor Flow

Complete the following steps to configure the Tensor Flow on the GPU node:

- Clone the tensorflow benchmark scripts from GitHub on to the node on to the IBM Spectrum Scale shared file system. Make sure that you are not in the container and run the following cloning command from the RHEL node, not from the container image:

  ```
  [root@ip-172-31-4-241 gpfs0]# pwd
  /ibm/gpfs0
  [root@ip-172-31-4-241 gpfs0]# git clone
  https://github.com/tensorflow/benchmarks
  Cloning into 'benchmarks'...
  ```

```
remote: Enumerating objects: 4806, done.
remote: Total 4806 (delta 0), reused 0 (delta 0), pack-reused 4806
Receiving objects: 100% (4806/4806), 2.39 MiB | 0 bytes/s, done.
Resolving deltas: 100% (3195/3195), done.
[root@ip-172-31-4-241 gpfs0]#
```

Test methodology

Complete the following steps:

1. Run the tensorflow container image with the IBM Spectrum Scale file system mount point.

```
[root@ip-172-31-4-241 gpfs0]# nvidia-docker images
REPOSITORY                  TAG                 IMAGE ID            CREATED             SIZE
nvidia/cuda                 11.0-base           2ec708416bb8        2 months ago        122MB
nvcr.io/nvidia/tensorflow   20.03-tf1-py3       8b2abbd886f0        7 months ago        9.51GB
nvcr.io/nvidia/tensorflow   20.03-tf2-py3       9af3e368023b        7 months ago        7.44GB
nvidia/cuda                 10.0-base           841d44dd4b3c        11 months ago       110MB
[root@ip-172-31-4-241 gpfs0]#
[root@ip-172-31-4-241 ~]# nvidia-docker run --shm-size=1g --ulimit memlock=-1 --ulimit
stack=67108864 -v /ibm/gpfs0/:/ibm/gpfs0 --network=host -it nvcr.io/nvidia/tensorflow:20.03-tf1-py3
/bin/bash
================
== TensorFlow ==
================
NVIDIA Release 20.03-tf1 (build 11025831)
TensorFlow Version 1.15.2
Container image Copyright (c) 2019, NVIDIA CORPORATION.  All rights reserved.
Copyright 2017-2019 The TensorFlow Authors.  All rights reserved.
NVIDIA Deep Learning Profiler (dlprof) Copyright (c) 2020, NVIDIA CORPORATION.  All rights
reserved.
Various files include modifications (c) NVIDIA CORPORATION.  All rights reserved.
NVIDIA modifications are covered by the license terms that apply to the underlying project or file.

MOFED driver for multi-node communication was not detected. Multi-node communication performance
may be reduced.

root@667f517eaf24:/workspace#
root@667f517eaf24:/workspace#
```

2. Change the directory to the tensorflow benchmark scripts folder in the container:

```
root@667f517eaf24:/ibm/gpfs0/benchmarks/scripts/tf_cnn_benchmarks# pwd
/ibm/gpfs0/benchmarks/scripts/tf_cnn_benchmarks
root@667f517eaf24:/ibm/gpfs0/benchmarks/scripts/tf_cnn_benchmarks#
```

3. Run tensor flow benchmark scripts on a single node.
4. Run the following command from the container.

```
root@667f517eaf24:/ibm/gpfs0/benchmarks/scripts/tf_cnn_benchmarks# python
tf_cnn_benchmarks.py --num_gpus=1 --device=gpu --use_fp16=True
--data_format=NCHW --batch_size=256 -batch_group_size=20 --num_batches=1000
--data_name=imagenet --data_dir=/ibm/gpfs0/imageDB/HPM_format/
--model=resnet50 --print_training_accuracy --train_dir=/ibm/gpfs0/results/
--nodistortions --use_datasets=True --summary_verbosity=1
--datasets_use_prefetch=True --datasets_prefetch_buffer_size=1
--variable_update=horovod --horovod_device=gpu
```

5. Run the training with the multiple GPUs on the same host. This process requires mpirun-based coordinated execution from the container:

```
root@ip-172-31-4-241:/ibm/gpfs0/benchmarks/scripts/tf_cnn_benchmarks# mpirun
--n  8 -allow-run-as-root --host host1:8 --report-bindings -bind-to none -map-by
```

```
slot -x LD_LIBRARY_PATH -x PATH -mca plm_rsh_agent ssh -mca plm_rsh_args "-p
12345" -mca pml ob1 -mca btl ^openib -mca btl_tcp_if_include eth0 -x
NCCL_DEBUG=INFO -x NCCL_SOCKET_NTHREADS=2 -x NCCL_NSOCKS_PERTHREAD=8 python
tf_cnn_benchmarks.py --num_gpus=1 --device=gpu --use_fp16=True
--data_format=NCHW --batch_size=256 -batch_group_size=20 --num_batches=1000
--data_name=imagenet --data_dir=/ibm/gpfs0/imageDB/HPM_format/
--model=resnet50 --print_training_accuracy --train_dir=/ibm/gpfs0/results/
--nodistortions --use_datasets=True --summary_verbosity=1
--datasets_use_prefetch=True --datasets_prefetch_buffer_size=1
--variable_update=horovod --horovod_device=gpu
```

6. Complete the following steps to run tensor flow benchmark scripts on multiple P3.16xlarge instances.

   a. Create a custom Docker image with the SSH capabilities enabled, so that containers running with this image allows to SSH each other for running the mpi commands:

   b. Start the custom build docker image on the other hosts - host2, host3, hosts4… :

   ```
   [root@ip-172-31-4-241 ~]#  nvidia-docker run --shm-size=1g --ulimit
   memlock=-1 --ulimit stack=67108864 -v /ibm/gpfs0/:/ibm/gpfs0 --network=host
   -it nvcr.io/nvidia/tensorflow:20.03-tf1-py3 /bin/bash -C "/usr/sbin/sshd  -p
   12345 ; sleep infinity"
   ```

   c. Start the custom build Docker image on host1:

   ```
   [root@ip-172-31-4-241 ~]#  nvidia-docker run --shm-size=1g --ulimit
   memlock=-1 --ulimit stack=67108864 -v /ibm/gpfs0/:/ibm/gpfs0 --network=host
   -it nvcr.io/nvidia/cusom-tensorflow:20.03-tf1-py3 /bin/bash
   ```

   d. After logging into the container on host1, execute the following command:

   ```
   mpirun --n  64 -allow-run-as-root --host host1:8,host2:8, host3:8, host4:8,
   host5:8, host6:8, host7:8, host8:8 --report-bindings -bind-to none -map-by
   slot -x LD_LIBRARY_PATH -x PATH -mca plm_rsh_agent ssh -mca plm_rsh_args "-p
   12345" -mca pml ob1 -mca btl ^openib -mca btl_tcp_if_include eth0 -x
   NCCL_DEBUG=INFO -x NCCL_SOCKET_NTHREADS=4 -x NCCL_NSOCKS_PERTHREAD=4  -x
   NCCL_SOCKET_IFNAME=eth0 python tf_cnn_benchmarks.py --num_gpus=1
   --device=gpu --use_fp16=True --data_format=NCHW --batch_size=128
   -batch_group_size=20 --num_batches=1000 --data_name=imagenet
   --data_dir=/ibm/gpfs0/data_copy  --model=inception4
   --print_training_accuracy --train_dir=/ibm/gpfs0/results/ --nodistortions
   --use_datasets=True --summary_verbosity=1 --datasets_use_prefetch=True
   --datasets_prefetch_buffer_size=1 --variable_update=horovod
   --horovod_device=gpu
   ```

Author

This paper was produced by a team of specialists from around the world working with IBM Redbooks, Tucson Center.

Sanjay Sudam is a Senior Solutions Architect for the data and AI solutions using IBM® Storage systems. He is responsible for creating the reference architectures and solution blueprints for the IBM storage portfolio and external ISV partners solutions. Sanjay has created end-to-end reference architectures for AI, analytics, cloud data, Data Protection, and Digital Video Surveillance and Media Entertainment solutions with the IBM portfolio.

Thanks to the following people for their contributions to this project:

Larry Coyne
IBM Redbooks®, Tucson Center

Udayasuryan Kodoly
Muthuannamalai Muthiah
Sasikanth Eda
Rajan Mishra
Simon Lorenz
Aaron S Palazzolo
IBM Systems

Girish Chanchlani
Kenneth Chang
Jeanna James
Baris Guler
Amazon Web Services

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an IBM Redbooks residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

IBM Redbooks Residencies

Stay connected to IBM Redbooks

- Find us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

Notices

This information was developed for products and services offered in the US. This material might be available from IBM in other languages. However, you may be required to own a copy of the product or product version in that language in order to access it.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, MD-NC119, Armonk, NY 10504-1785, US

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some jurisdictions do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you provide in any way it believes appropriate without incurring any obligation to you.

The performance data and client examples cited are presented for illustrative purposes only. Actual performance results may vary depending on specific configurations and operating conditions.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to actual people or business enterprises is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at “Copyright and trademark information” at http://www.ibm.com/legal/copytrade.shtml

The following terms are trademarks or registered trademarks of International Business Machines Corporation, and might also be trademarks or registered trademarks in other countries.

Redbooks (logo) ®
IBM®
IBM Spectrum®
Passport Advantage®
Redbooks®

The following terms are trademarks of other companies:

The registered trademark Linux® is used pursuant to a sublicense from the Linux Foundation, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis.

Ansible, Red Hat, are trademarks or registered trademarks of Red Hat, Inc. or its subsidiaries in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

REDP-5626-00

ISBN 0738459550

Printed in U.S.A.