

APEX File Storage for AWS

Manual Deployment Guide

June 2024

H19556.5

Deployment Guide

Abstract

This document provides guidance for preparing APEX File Storage for AWS deployment, including instructions for deploying AWS resources for PowerScale OneFS cluster with AWS Management Console and AWS CLI manually.

Copyright

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

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

Copyright © 2023-2024 Dell Inc. or its subsidiaries. All Rights Reserved. Published in the USA June 2024 H19556.5.

Dell Inc. believes the information in this document is accurate as of its publication date. The information is subject to change without notice.

Contents

Executive summary.....	4
Introduction	5
Deployment overview	8
Plan your deployment.....	9
Prerequisites check list	13
Deploy AWS infrastructure	15
Set up the cluster	47
Expand a OneFS cluster	50
OneFS SmartConnect	52
Replace a volume for a cluster node	54
Replace a cluster node	63
Maintenance events from AWS	66
Stopping OneFS node instances	66
Appendix A: required AWS permission for deployer user	68
Appendix B: supported cluster configuration details.....	71
Appendix C: json file template	72
Appendix D: EC2 instance user data example	77

Executive summary

Overview

This deployment guide provides instructions for deploying PowerScale OneFS clusters for APEX File Storage for AWS. It also provides guidance on post-deployment, including expanding a cluster, OneFS SmartConnect, replacing a volume for a cluster, and replacing a cluster node.

After you have deployed a cluster, see the APEX File Storage for AWS Getting Started Guide on the [Dell Technologies Support](#) website for more details about OneFS. If you are looking for an auto-deployment guide with Terraform, see the document [APEX File Storage for AWS Deployment Guide with Terraform](#).

Revisions

Date	Part number/revision	Description
May 2023	H19556	Initial release
August 2023	H19556.1	Updated links in <i>AWS CLI instructions</i> and <i>AWS Management Console instructions</i> sections
November 2023	H19556.2	Updated with new instance types supported (m5d.24xlarge and i3en.12xlarge)
December 2023	H19556.3	Updated with OneFS 9.7 release new features, including new instance type, larger capacity to 1.6PiB
April 2024	H19556.4	Updated with Dell APEX Navigator deployment introduction
June 2024	H19556.5	Add more permission for the IAM policy

We value your feedback

Dell Technologies and the authors of this document welcome your feedback on this document. Contact the Dell Technologies team by [email](#).

Author: Lieven Lin

Note: For links to other documentation for this topic, see [PowerScale Info Hub](#).

Introduction

About this guide

AWS cloud and storage administrators can use this guide to deploy the AWS infrastructure resources for deploying and configuring OneFS as an Elastic Compute Cloud (EC2) instance on AWS. To use this guide effectively, administrators should be familiar with the AWS cloud services, including EC2, Elastic Block Storage (EBS), VPC, and IAM.

After you have deployed a cluster, see the APEX File Storage for AWS Getting Started Guide on the [Dell Technologies Support](#) website for more details about OneFS. If you are looking for an auto-deployment guide with Terraform, see the document [APEX File Storage for AWS Deployment Guide with Terraform](#).

The following conventions are adopted in this guide to represent the first and subsequent occurrences of frequently used terminology in a section:

Table 1. Terminology

First occurrence in the title and introductory paragraph	Usage thereafter
Dell Technologies PowerScale OneFS	OneFS
OneFS cluster	Cluster
OneFS cluster node	Node
OneFS cluster internal subnet	Internal subnet
OneFS cluster external subnet	External subnet
OneFS cluster internal security group	Internal security group
OneFS cluster external security group	External security group
OneFS cluster internal network interfaces	Internal network interfaces
OneFS cluster external network interfaces	External network interfaces

APEX File Storage for AWS deployment methods

In the realm of modern software deployment, flexibility and efficiency are important. APEX File Storage for AWS offers various deployment methods to suit diverse user needs. Choose the most suitable method from the following three methods:

Manual deployment with AWS CLI and Management Console

The manual deployment method offers users complete control and customization over the deployment process. By following detailed step-by-step instructions, users can manually configure and deploy AFS according to their specific requirements. This method is ideal for users who prefer hands-on management and want a deep understanding of their deployment architecture. This document focuses on the manual deployment approach.

Auto-deployment with Terraform

Leveraging the power of infrastructure as code (IaC), APEX File Storage for AWS provides an automated deployment option using Terraform. With Terraform's declarative syntax and the Terraform module terraform-aws-onefs, users can define their

infrastructure in code and execute it to provision AWS resources automatically. This method streamlines deployment processes, reduces human error, and ensures consistency across environments. It is well-suited for users who are seeking efficient and repeatable deployments. For details, see [APEX File Storage for AWS Deployment Guide with Terraform](#).

Auto-deployment with Dell APEX Navigator (AWS US regions only)

[Dell APEX Navigator](#) elevates your multicloud experience through one unified console that streamlines operations across public cloud environments. Dell APEX Navigator offers seamless integration with APEX File Storage for AWS for automated deployments in AWS US regions (US-East 1, US-East 2, and US-West 2). Dell APEX Navigator orchestrates the deployment process effortlessly, ensuring rapid and consistent deployments, so you can deploy APEX File Storage for AWS with minimal manual intervention. For details, see [Dell APEX Navigator documentation](#).

APEX File Storage for AWS architecture

APEX File Storage for AWS is a software-defined and customer-managed scale-out storage solution running on AWS cloud infrastructure. It brings the Dell Technologies PowerScale OneFS distributed file system into the public cloud, allowing users to have the same management experience as on their on-premises PowerScale appliances. You can run OneFS on multiple EC2 instances backed by EBS volumes, and then form a OneFS cluster with the EC2 instances' virtual nodes.

In general, cluster performance is related to your cluster size. Larger cluster size delivers higher throughput and IOPS. Therefore, before creating your cluster in AWS, consider the capacity of storage and number of nodes that you need for your business workflow.

Figure 1 shows the architecture of APEX File Storage for AWS.

- **Availability zone:** APEX File Storage for AWS is designed to run in a spread strategy placement group within a single availability zone to get the best performance.
- **VPC:** APEX File Storage for AWS requires an AWS VPC to provide network connectivity.
- **OneFS cluster internal subnet:** the cluster nodes communicate with each other through the internal subnet. The internal subnet must be isolated from instances that are not in the cluster. Therefore, a dedicated subnet is required for the internal network interfaces of cluster nodes that do not share the internal subnets with other EC2 instances.
- **OneFS cluster external subnet:** the cluster nodes communicate with clients through the external subnet by using different protocols, such as NFS, SMB, and S3.
- **OneFS cluster internal network interfaces:** network interfaces that are located in the internal subnet.
- **OneFS cluster external network interfaces:** network interfaces that are located in the external subnet.
- **OneFS cluster internal security group:** the security group applies to the cluster internal network interfaces which allows all traffic between the cluster nodes' internal network interfaces only.

- **OneFS cluster external security group:** the security group applies to cluster external network interfaces which allows specific ingress traffic from clients.
- **EC2 instance nodes:** cluster nodes which run the OneFS filesystem backed by EBS volumes and provide network bandwidth.

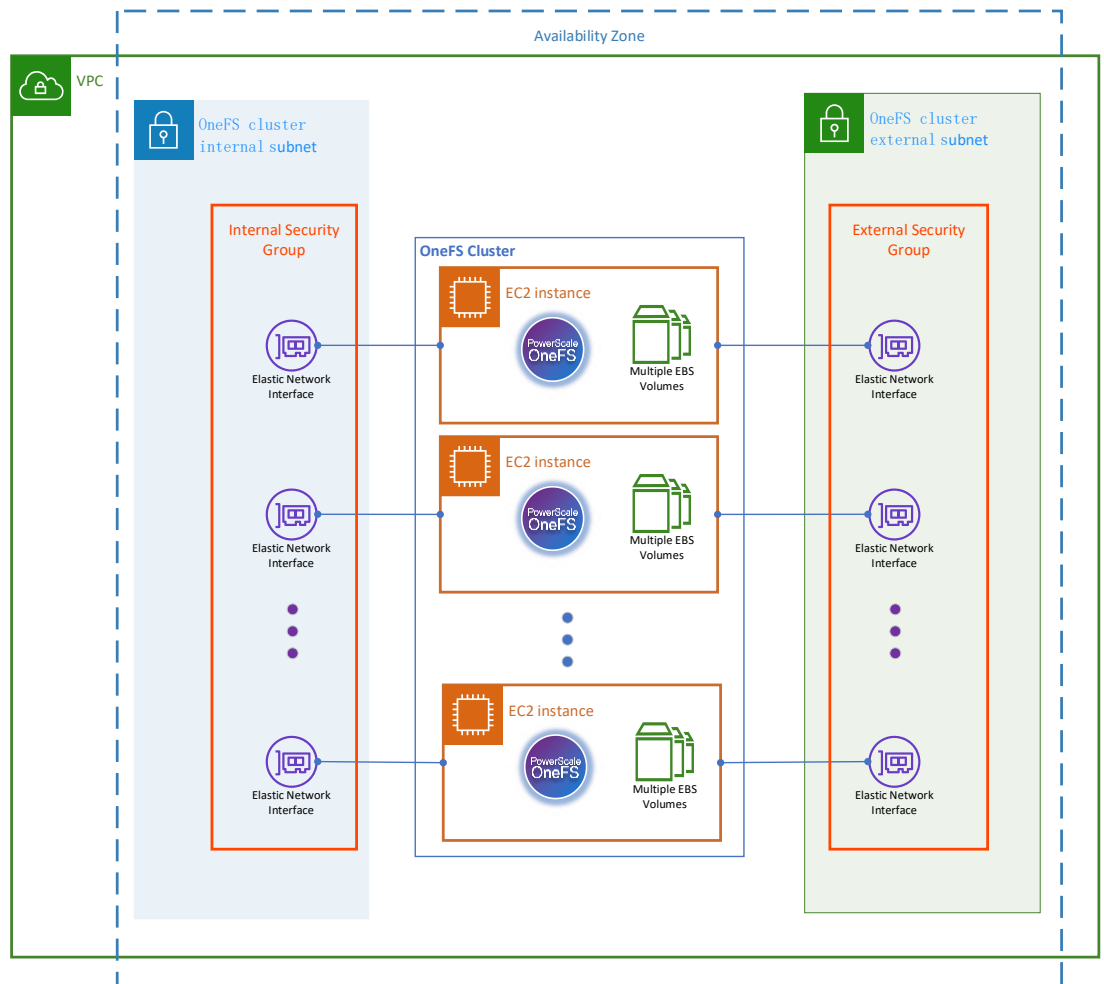


Figure 1. APEX File Storage for AWS architecture

Deployment requirement

Before you deploy a cluster, verify that the following requirements are satisfied:

AWS subscription

APEX File Storage for AWS requires an AWS subscription. Visit the Amazon website for more details of [AWS subscription pricing](#).

AWS permissions

You need the AWS permissions listed in [Appendix A: required AWS permission for deployer user](#) to deploy clusters in AWS.

EC2 instance type

Ensure that the instance types are available in your AWS availability zone. See the AWS documentation [Find an Amazon EC2 instance type](#) for details. Starting from OneFS 9.7, APEX File Storage for AWS supports the following instance types:

- EC2 m5dn instances: m5dn.8xlarge, m5dn.12xlarge, m5dn.16xlarge, m5dn.24xlarge
- EC2 m6idn instances: m6idn.8xlarge, m6idn.12xlarge, m6idn.16xlarge, m6idn.24xlarge
- EC2 m5d instances: m5d.24xlarge
- EC2 i3en instances: i3en.12xlarge

Note: You must run a Proof of Concept (PoC) if you intend to use m5d.24xlarge or i3en.12xlarge EC2 instance types. For details, contact your Dell account team.

EBS volume type

Every cluster node has an EBS volume for the OneFS operating system, a journaling device by leveraging the EC2 instance local NVMe storage, and a fixed set of data volumes for the OneFS file system /ifs. Unless otherwise specified, we only refer to EBS data volumes for /ifs. APEX File Storage for AWS supports st1 and gp3 EBS types as the /ifs data volume.

AWS provides EBS encryption capability for each EBS volumes. The encryption is finally transparent to OneFS software. Therefore, when creating an EBS volume, you can choose to enable encryption, and you can select the AWS Key Management Service (KMS) key that will be used for encryption, either AWS managed keys or customer managed keys. This key management allows you to control who can access the encrypted data and provides additional flexibility in key management.

VPC subnet

APEX File Storage for AWS requires two different subnets.

- A dedicated internal subnet for cluster internal network interfaces. The subnet cannot be shared with other EC2 instances.
- An external subnet for cluster external network interfaces.

Note: IPv6 is not currently supported.

AWS Command Line Interface

The AWS Command Line Interface v2 (AWS CLI v2) is a unified tool to manage your AWS services. You should install and configure AWS CLI in advance if you intend to use AWS CLI for deploying AWS resources.

Deployment overview

If you are looking for auto-deployment with Terraform, see the [APEX File Storage for AWS Deployment Guide with Terraform](#). APEX File Storage for AWS deployment involves four high level steps.

- **Plan your deployment:** APEX File Storage for AWS has a predefined supported configuration, including cluster size, cluster capacity, EC2 instance type, EBS volumes type, and so on. You would need to know about the supported configuration before you start to deploy the cluster. See the section [Supported cluster configuration](#) for more details.

- **Prerequisites check list:** you must fulfill all the prerequisites before following this guide to deploy a cluster. See the section [Prerequisites check list](#) for more details.
- **Deploy AWS infrastructure:** after you have planned your deployment and fulfilled the prerequisites, most of the deployment effort is to deploy the AWS infrastructures for the cluster, including the placement group, security group, network resources, and EC2 instances. See the section [Deploy AWS infrastructure](#) for more details.
- **Set up a cluster:** connect to the first node of the cluster to add additional nodes to the cluster. Refer to the section [Set up the cluster](#) for more details.

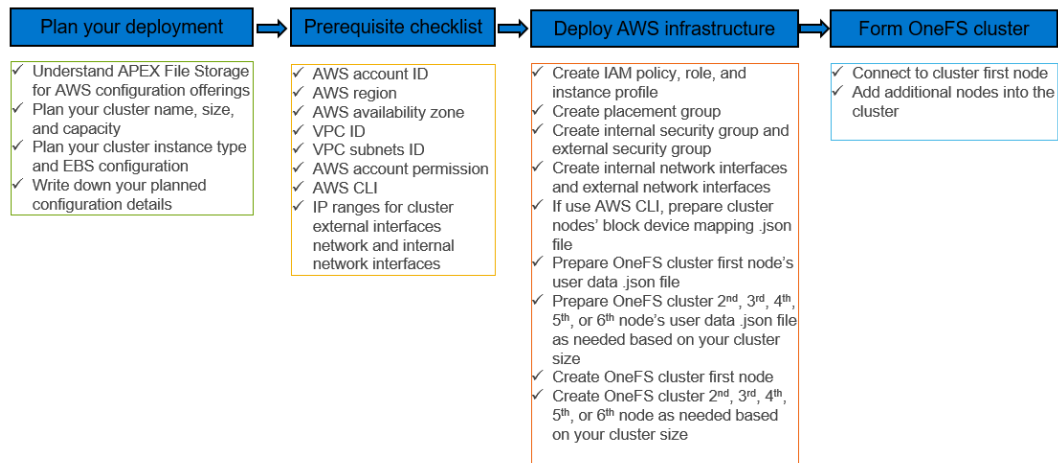


Figure 2. Deployment overview

Plan your deployment

When setting up a new cluster, it is important to consider the type of storage configuration that is supported. Different configurations fulfill different requirements, depending on the intended use and the amount of data that needs to be stored. In general, a supported cluster configuration should be reliable, performant, and offer enough storage capacity to meet your needs.

Therefore, before starting to deploy a cluster, you should be familiar with the supported cluster configuration offering of APEX File Storage for AWS. Based on your actual business needs and future growth, choose an SSD or HDD cluster configuration that meets your requirements.

Supported cluster configuration

For a single cluster of APEX File Storage for AWS, all nodes in the cluster must use the same configuration, including EC2 instance type and size, EBS volume type, and EBS volume size.

When a new cluster is deployed, the default protection level is set to +2n. All the files on the cluster inherit this level of protection. If the cluster only has the minimum number of nodes (that is, 4), +2n protection means that the data on the cluster will remain protected even when two nodes of the cluster fail simultaneously. In a hyperscaler's environment,

each of the OneFS virtual nodes within a cluster resides on different underlying physical hardware. Therefore, the chances of two nodes failing simultaneously are slim.

In the highly unlikely event that two of the four nodes fail simultaneously, the cluster will only have two functional nodes and it loses quorum. Regardless of the cluster protection level, a cluster loses quorum when at least half the number of nodes in the cluster are unavailable. To protect the data, the cluster becomes unresponsive to client I/O. When at least one of the two failed nodes is back in the cluster, the cluster automatically becomes writable. The data will be unavailable until at least one node is back online (to bring back the quorum) but there will be no data loss. If needed, some data will be rebuilt automatically. If the cluster must remain available and writable even when two nodes fail with +2n protection, the cluster size should be increased to five nodes.

APEX File Storage for AWS supports the following cluster configuration.

SSD cluster

Table 2 shows the supported configuration for an SSD cluster.

Table 2. Supported configuration for an SSD cluster

Configuration items	Supported options
Cluster size	4 to 6 nodes
EC2 instance type	<p>All nodes in a cluster must be the same instance size. The supported instance sizes are:</p> <ul style="list-style-type: none"> EC2 m5dn instances: m5dn.8xlarge, m5dn.12xlarge, m5dn.16xlarge, m5dn.24xlarge EC2 m6idn instances: m6idn.8xlarge, m6idn.12xlarge, m6idn.16xlarge, m6idn.24xlarge EC2 m5d instances: m5d.24xlarge EC2 i3en instances: i3en.12xlarge <p>Note: It is required to run PoC if you intend to use m5d.24xlarge or i3en.12xlarge EC2 instance types. Contact your Dell account team for the details.</p>
EBS volume type	gp3
EBS volume counts per node	5, 6, 10, 12, 15, 18, or 20
Single EBS volume sizes	1TiB - 16TiB
Cluster raw capacity	20TiB - 1.6PiB
Cluster protection level	+2n

Note: all criteria in this table must be met. Therefore, **not all combinations of cluster size, volume count, and single volume size are supported.** For some combinations of cluster size, volume account, and single volume size, the total cluster raw capacity may fall outside the maximum supported cluster raw capacity. For example, for a 6-nodes cluster, if each node contains 20 volumes, and each volume size is 16TiB, the final total cluster raw capacity is 1920TiB which exceeds the maximum supported raw capacity. This combination is therefore not supported.

See the section [Appendix B: supported cluster configuration details](#) for all supported combinations.

HDD cluster

[Table 3](#) shows the supported configuration for an HDD cluster.

Table 3. Supported configuration for an HDD cluster

Configuration items	Supported options
Cluster size	4 to 6 nodes
EC2 instance type	<p>All nodes in a cluster must be the same instance size. The supported instance sizes are:</p> <ul style="list-style-type: none"> EC2 m5dn instances: m5dn.8xlarge, m5dn.12xlarge, m5dn.16xlarge, m5dn.24xlarge EC2 m6idn instances: m6idn.8xlarge, m6idn.12xlarge, m6idn.16xlarge, m6idn.24xlarge EC2 m5d instances: m5d.24xlarge EC2 i3en instances: i3en.12xlarge <p>Note: It is required to run PoC if you intend to use m5d.24xlarge or i3en.12xlarge EC2 instance types. Contact your Dell account team for the details.</p>
EBS volume type	st1
EBS volume counts per node	5 or 6
Single EBS volume sizes	4TiB or 10TiB
Cluster raw capacity	80TiB - 360TiB
Cluster protection level	+2n

Available regions

The [Table 4](#) list all available regions for each instance type.

Table 4. Available regions

AWS location	Region code	Supported instance types
US East (N. Virginia)	us-east-1	m5dn, m6idn, m5d.24xlarge, i3en.12xlarge
US East (Ohio)	us-east-2	m5dn, m6idn, m5d.24xlarge, i3en.12xlarge
US West (N. California)	us-west-1	m6idn, m5d.24xlarge, i3en.12xlarge
US West (Oregon)	us-west-2	m5dn, m6idn, m5d.24xlarge, i3en.12xlarge
South America (Sao Paulo)	sa-east-1	m5d.24xlarge, i3en.12xlarge
Canada (Central)	ca-central-1	m5d.24xlarge, i3en.12xlarge
EU (Frankfurt)	eu-central-1	m5dn, m6idn, m5d.24xlarge, i3en.12xlarge
Europe (Zurich)	eu-central-2	m5d.24xlarge, i3en.12xlarge
EU (Stockholm)	eu-north-1	m6idn, m5d.24xlarge, i3en.12xlarge
EU (Milan)	eu-south-1	m5d.24xlarge, i3en.12xlarge
Europe (Spain)	eu-south-2	m5d.24xlarge, i3en.12xlarge

AWS location	Region code	Supported instance types
EU (Ireland)	eu-west-1	m5dn, m6idn, m5d.24xlarge, i3en.12xlarge
EU (London)	eu-west-2	m5d.24xlarge, i3en.12xlarge
EU (Paris)	eu-west-3	m5d.24xlarge, i3en.12xlarge
Israel (Tel Aviv)	il-central-1	m5d.24xlarge, i3en.12xlarge
Middle East (UAE)	me-central-1	m5d.24xlarge, i3en.12xlarge
Middle East (Bahrain)	me-south-1	m5d.24xlarge, i3en.12xlarge
Africa (Cape Town)	af-south-1	m5d.24xlarge, i3en.12xlarge
Asia Pacific (Hong Kong)	ap-east-1	m5d.24xlarge, i3en.12xlarge
Asia Pacific (Tokyo)	ap-northeast-1	m5dn, m6idn, m5d.24xlarge, i3en.12xlarge
Asia Pacific (Seoul)	ap-northeast-2	m5d.24xlarge, i3en.12xlarge
Asia Pacific (Osaka)	ap-northeast-3	m5d.24xlarge, i3en.12xlarge
Asia Pacific (Mumbai)	ap-south-1	m5d.24xlarge, i3en.12xlarge
Asia Pacific (Hyderabad)	ap-south-2	m5d.24xlarge, i3en.12xlarge
Asia Pacific (Singapore)	ap-southeast-1	m5dn, m6idn, m5d.24xlarge, i3en.12xlarge
Asia Pacific (Sydney)	ap-southeast-2	m6idn, m5d.24xlarge, i3en.12xlarge
Asia Pacific (Jakarta)	ap-southeast-3	m5d.24xlarge, i3en.12xlarge
Asia Pacific (Melbourne)	ap-southeast-4	m5d.24xlarge, i3en.12xlarge

Note: You must run a Proof of Concept (PoC) if you intend to use m5d.24xlarge or i3en.12xlarge EC2 instance types. For details, contact your Dell account team. AWS can add new regions and also make the supported instance types available in any of the regions/availability zones anytime. Verify that the instance type you intend to use is available in your desired region/availability zone.

Large file support

Starting from OneFS 9.7, OneFS Large file support is available for APEX File Storage for AWS. Ensure that your cluster configuration meets the prerequisites mentioned in the document [PowerScale OneFS Large File Support](#).

Analyze your business requirements

Based on your actual business workflow requirement, determine the required cluster size, raw capacity, and performance. You should consider the future growth of your business needs when planning your deployment. It is easier to determine the cluster raw capacity than cluster performance.

In general, cluster performance is related to your cluster size, instance type, and EBS volume configuration. More cluster nodes deliver higher throughput and IOPS. The higher instance size has a higher bandwidth limit.

- There are also different performance characteristics between gp3 and st1 EBS volumes. gp3 volumes deliver a consistent baseline IOPS performance of 3,000 IOPS and a consistent baseline throughput performance of 125 MiB/s, which

are included with the price of storage. See the AWS documentation for more details about [gp3 volume performance](#).

- st1 volumes deliver a throughput optimized HDD storage, which provides low-cost magnetic storage that defines performance in terms of throughput rather than IOPS. This volume type is a good fit for large, sequential workloads and archive workloads. The st1 volume size determines both the baseline throughput and burst throughput of your volume. See the AWS documentation for more details about [st1 volume performance](#).

For more cluster performance considerations and performance test results, see the document [APEX File Storage for AWS - Overview and Performance](#).

Planning cluster configuration details

After you have understood the supported cluster configuration and have analyzed your business requirements, you can start to plan your deployment details. Write down the configuration, including your OneFS cluster settings and AWS infrastructure information. The following table contains an example used in this guide for each item. For this example, we will deploy a 4-node SSD cluster using the m5dn.12xlarge instance type.

Configuration items	Example used in this guide	Note
Cluster name	vonefs-cfv	To manage your cluster's AWS infrastructure efficiently, it is highly recommended to tag all your AWS resources related to a specific cluster by adding a tag: {key=cluster-name; value=<cluster-name>}
EC2 instance type	m5dn.12xlarge	
EBS type	gp3	
Cluster size	4	If you are deploying a 4-node or 5-node cluster, and you need to expand the cluster size in the future, you must ensure that you are using a correct combination of volume count per node and single volume size, which allows you to expand to a maximum of 6-nodes. See section Appendix B: supported cluster configuration details for all supported combinations
EBS volume counts per node	6	
Single EBS volume sizes	1TiB	
Cluster raw capacity	24TiB	Ensure your cluster raw capacity is within the supported range. See section Supported cluster configuration for the capacity limitation.

Prerequisites check list

This section describes the prerequisites details that you should fulfill before proceeding with the deployment.

IP range of cluster internal network interfaces

A cluster requires a dedicated internal subnet for cluster internal network interfaces. The subnet cannot be shared with other EC2 instances. For a single cluster, you must have six contiguous IPs to ensure that the cluster can expand to a maximum of six nodes.

IP range of cluster external network interfaces

Default network pool

During cluster deployment, you create a default cluster network pool named groupnet0.subnet0.pool0. Each node in the cluster is assigned one IP address from this pool. The IP addresses used in the pool groupnet0.subnet0.pool0 are the AWS primary IPv4 addresses, and cannot be moved from one node to another. Thus, the allocation type of this pool cannot be changed to dynamic.

You must have six contiguous IPs for the default network pool, to ensure that the cluster has enough IPs to expand to a maximum of six nodes.

The Externally Managed IPs feature in OneFS 9.7 introduces a limited form of DHCP for managing IP allocation in groupnet0.subnet0.pool0 and ensures that an IP in the OneFS network pool is assigned to the correct network interface of a node as the primary IP. To ensure the integrity of this process and mitigate potential security risks of a rogue DHCP server, it is recommended to add an inbound rule in the cluster external security group in AWS. See Table 5 for the details of the rule:

Table 5. Inbound rule for DHCP

Setting	Value
Rule name	For example, DHCP
Type	Ingress
From port	67
To port	68
Protocol	udp
Allowed CIDR blocks	<cluster-gateway>/32

Additional network pools

After a cluster is deployed, users are allowed to create additional network pools. These new pools can use static or dynamic allocation. Any unused IPs from the cluster external subnet CIDR range can be used to create pools. The IPs from additional network pools are assigned to cluster nodes as AWS secondary IPv4 addresses.

Note: APEX File Storage for AWS does not support IPv6.

Prerequisites checklist details and example

Table 6 shows the prerequisites check list details and examples used in this guide.

Table 6. Prerequisites checklist details

Prerequisite item	Example used in this guide	Description
AWS account ID	551948851026	Your AWS account ID. Will be used if you use AWS CLI to do the deployment.
AWS region	us-east-1	The region in which the cluster will run.
AWS availability zone	us-east-1b	The availability zone in which the cluster will run.

Prerequisite item	Example used in this guide	Description
AWS VPC ID	vpc-06639db65d7446720	The VPC ID in which the cluster will run.
Cluster internal subnet ID	subnet-0dc84f8a0a6f8dd8d	The subnet for the cluster internal network interfaces.
Cluster internal subnet IPv4 CIDR	10.0.0.16/28	
IP range of cluster internal network interfaces	10.0.0.20 - 10.0.0.25	The contiguous IPs for the cluster internal network interfaces, one IP per node.
Cluster external subnet ID	subnet-058548a2c6df9591c	The subnet for the cluster external network interfaces.
Cluster external subnet IPv4 CIDR	10.0.0.0/28	Make sure there are enough IPs for the default network pool, SmartConnect Service IPs (SSIPs), and any additional network pools that you may create later.
IP range of cluster external network interfaces	10.0.0.4 - 10.0.0.9	The contiguous IPs of the cluster default network pool groupnet0.subnet0.pool0 assigned to the external network interfaces, one IP per node.

Deploy AWS infrastructure

This section provides instructions for deploying the required AWS infrastructure resources for APEX File Storage for AWS, including:

- IAM policy, role, and instance profile: this is a one-time activity. The IAM policy and role is reusable for additional cluster deployment.
- Placement group: A spread strategy placement group is required to ensure that OneFS nodes are placed on distinct hardware to ensure high availability. See the [AWS placement groups](#) documentation for additional details.
- Security group:
 - Creating an internal security group for cluster internal network interfaces only.
 - Creating an external security group for cluster external network interfaces to allow specific ingress traffic from clients.
- Network interfaces:
 - Network interfaces for cluster internal network interfaces
 - Network interfaces for cluster external network interfaces
- If you use AWS CLI to perform the deployment, you must prepare the cluster nodes block device mapping .json file. OneFS uses EBS devices as data volumes. All EC2 instances must share the same configuration of block device mapping. See [AWS block device mapping](#) documentation for more details.
- Prepare the cluster nodes EC2 instance user data: OneFS requires user data in a .json format file. This file provides new instances running OneFS with the information needed to form a OneFS cluster.

- Create an interface endpoint if you are using a private VPC.
- Create EC2 instances.

This guide provides instructions for using the AWS CLI and AWS Management Console. You can choose the method that is best for you to perform the deployment.

Create IAM policy, role, and instance profile

Note: Creating the IAM policy, role, and instance profile for a OneFS cluster is a one-time activity for the same AWS account. The profiles are reusable for deploying more additional clusters.

Cluster nodes require an instance profile attached. The minimum permissions required is `ec2:AssignPrivateIpAddresses` on network interfaces, which is defined in the [onefs-runtime-policy.json](#).

AWS CLI instructions

1. Save the [onefs-runtime-policy.json](#) content as a .json file named `onefs-runtime-policy.json`, and replace the `<aws_account_id>` with your AWS account ID.
2. Open your OS CLI, which has the AWS CLI ready (Windows CMD in this guide) and change your current directory to `C:\json-files-template`. In this guide, we use `C:\json-files-template` as an example directory which stores all required .json files.

```
> cd C:\json-files-template
```

3. Create the IAM policy and write down the `Policy.Arn` field of the created policy. It is `arn:aws:iam::551948851026:policy/onefs-runtime-policy` in the following output example.

```
> aws iam create-policy --policy-name onefs-runtime-policy --
policy-document file://onefs-runtime-policy.json
```

Command output example:

```
{
  "Policy": {
    "PolicyName": "onefs-runtime-policy",
    "PolicyId": "ANPAYBAVXC5JA25QNSXQA",
    "Arn": "arn:aws:iam::55194881026:policy/onefs-
runtime-policy",
    "Path": "/",
    "DefaultVersionId": "v1",
    "AttachmentCount": 0,
    "PermissionsBoundaryUsageCount": 0,
    "IsAttachable": true,
    "CreateDate": "2022-11-17T08:14:23+00:00",
    "UpdateDate": "2022-11-17T08:14:23+00:00"
  }
}
```

4. Create the IAM assume role.

```
> aws iam create-role --role-name onefs-runtime-role --
assume-role-policy-document file://onefs-runtime-assume-
role.json
```

5. Using the policy ARN from step 3 to attach the policy to the role.


```
> aws iam attach-role-policy --role-name onefs-runtime-role --policy-arn arn:aws:iam::551948851026:policy/onefs-runtime-policy
```

6. Now create the instance profile named `onefs-runtime-instance-profile`.

```
> aws iam create-instance-profile --instance-profile-name onefs-runtime-instance-profile
```

7. Finally attach the role to the instance profile.

```
> aws iam add-role-to-instance-profile --instance-profile-name onefs-runtime-instance-profile --role-name onefs-runtime-role
```

AWS Management Console instructions

1. Sign into the AWS Management Console and open the IAM console at <https://console.aws.amazon.com/iam/>.
1. In the navigation pane on the left, choose **Policies**, and then choose **Create policy**.
2. Choose the **JSON** tab.
3. Copy the contents of [onefs-runtime-policy.json](#), and replace the `<aws_account_id>` with your AWS account ID.
4. Paste the modified contents of [onefs-runtime-policy.json](#) to the **JSON** tab. See the example in [Figure 3](#).

Policy editor

```

1 ▼ {
2     "Version": "2012-10-17",
3     "Statement": [
4         {
5             "Sid": "APEXFileStorageSmartConnectIPFailover",
6             "Effect": "Allow",
7             "Action": [
8                 "ec2:UnassignPrivateIpAddresses",
9                 "ec2:AssignPrivateIpAddresses"
10            ],
11            "Resource": "arn:aws:ec2:*:551948851030:network-interface/*"
12        },
13        {
14            "Sid": "APEXFileStorageSmartConnectValidation",
15            "Effect": "Allow",
16            "Action": [
17                "ec2:DescribeNetworkInterfaces",
18                "ec2:DescribeInstanceTypes"
19            ],
20            "Resource": "*"
21        }
22    ]
23 }

```

Figure 3. OneFS runtime policy

5. Choose **Next: Tags**.
6. Choose **Next: Review**.
7. On the **Review policy** page, type **"onefs-runtime-policy"** in the **Name** field. Then choose **Create policy** to complete the creation.
8. Next, we will create an IAM role. Open the IAM console and in the navigation pane on the left, choose **Roles**, and then choose **Create role**.
9. Under the **Select trusted entity**, choose **AWS service** for **Trusted entity type**, and choose **EC2** for **Use case**. Then, choose **Next**.
10. Search for the IAM policy **onefs-runtime-policy** and select the policy, then choose **Next**.

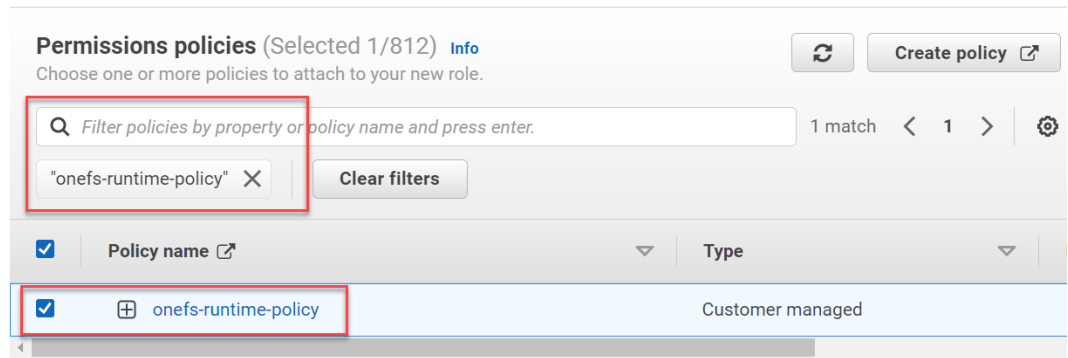


Figure 4. Add IAM policy

11. Type **onefs-runtime-role** for the Role name field.
12. Scroll down to the end and choose **Create role** to finish creating the role.

Note: When you use the AWS Management Console to create a role for Amazon EC2, the console automatically creates an instance profile and gives it the same name as the role. Now you also have an instance profile created named **onefs-runtime-role**. For more details, see the AWS documentation [Using instance profiles](#).

Create a placement group

A spread strategy placement group is required to ensure that OneFS nodes are placed on distinct hardware to ensure high availability. It is highly recommended to name your placement group with a format of `<cluster_name>-onefs-placement-group` by replacing the `<cluster_name>` with your planned configuration defined in the section [Planning cluster configuration details](#). Write down your placement group name for future reference.

AWS CLI instructions

Run the following command by replacing the `<aws_region>` and `<cluster_name>` with your settings. It is recommended that you add a tag `{Key=cluster-name, Value=<cluster_name>}` for future management convenience.

```
aws ec2 create-placement-group --strategy spread --group-name
<cluster_name>-onefs-placement-group --region <aws_region> --tag-
specifications "ResourceType=placement-group,Tags=[{Key=cluster-
name,Value=<cluster_name>}]"
```

The following is an example to create the placement group:

```
> aws ec2 create-placement-group --strategy spread --group-name
vonefs-cfv-onefs-placement-group --region us-east-1 --tag-
specifications "ResourceType=placement-group,Tags=[{Key=cluster-
name,Value=vonefs-cfv}]"
```

Command output example:

```
{
  "PlacementGroup": {
    "GroupName": "vonefs-cfv-onefs-placement-group",
    "State": "available",
```

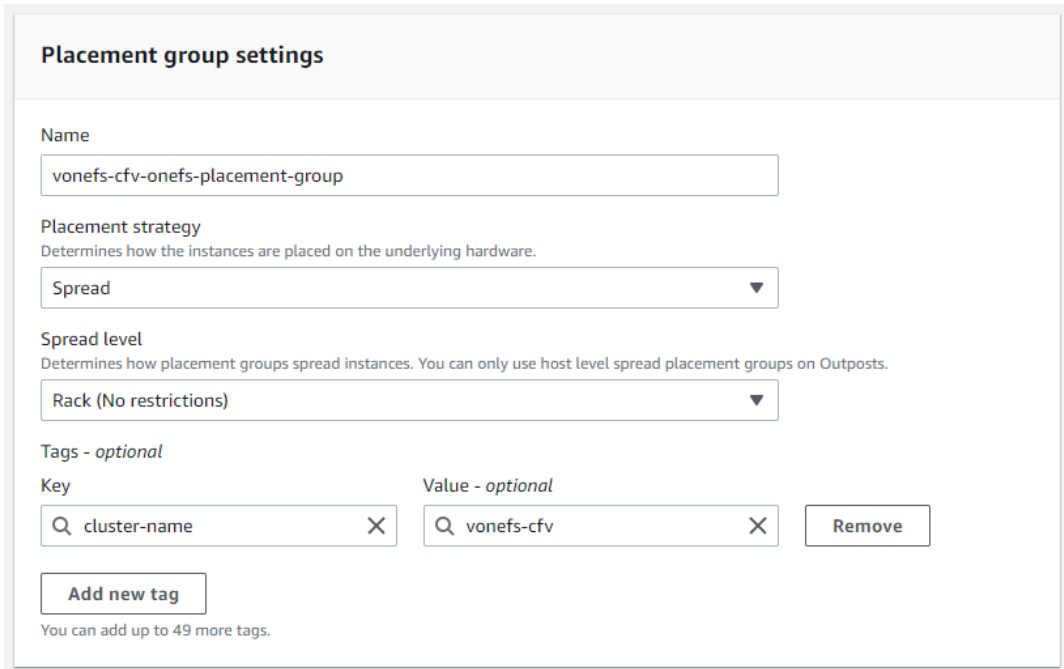
```

    "Strategy": "spread",
    "GroupId": "pg-0081802c575441e29",
    "Tags": [
      {
        "Key": "cluster-name",
        "Value": "vonefs-cfv"
      }
    ],
    "GroupArn": "arn:aws:ec2:us-east-1:551948851026:placement-group/vonefs-cfv-onefs-placement-group",
    "SpreadLevel": "rack"
  }
}

```

AWS Management Console instructions

1. Sign into the AWS Management Console and open the EC2 console at <https://console.aws.amazon.com/ec2/>.
1. In the navigation pane, choose **Placement Groups**, and then choose **Create placement group**.
2. In the **Name** field, it is recommended that you specify a name with a format of **<cluster_name>-onefs-placement-group** by replacing **<cluster_name>** with your own setting.
3. Choose **Spread** for the **Placement strategy** field.
4. Choose **Rack (No restrictions)** for the **Spread level** field.
5. Choose **Add new tag**. The tag key is **cluster-name**, and the tag value is your OneFS cluster name.
6. Choose **Create group**. [Figure 5](#) is an example.



Placement group settings

Name
vonefs-cfv-onefs-placement-group

Placement strategy
Determines how the instances are placed on the underlying hardware.
Spread

Spread level
Determines how placement groups spread instances. You can only use host level spread placement groups on Outposts.
Rack (No restrictions)

Tags - optional

Key	Value - optional	
cluster-name	vonefs-cfv	Remove

Add new tag

You can add up to 49 more tags.

Figure 5. Placement group

Create the internal security group

The security group for the cluster internal network interfaces is required for each cluster to limit traffic between the cluster nodes internal network interfaces only.

AWS CLI instructions

1. Run the following command by replacing the `<aws_vpc_id>`, `<aws_region>` and `<cluster_name>` with your own settings.

```
aws ec2 create-security-group --vpc-id <aws_vpc_id> --group-name <cluster_name>-internal-sg --region <aws_region> --tag-specifications "ResourceType=security-group,Tags=[{Key=cluster-name,Value=<cluster_name>}]" --description "Internal security group for OneFS cluster <cluster_name>"
```

Below is an example to create the internal security group. Write down the GroupId in the output.

```
> aws ec2 create-security-group --vpc-id vpc-06639db65d7446720 --group-name vonefs-cfv-internal-sg --region us-east-1 --tag-specifications "ResourceType=security-group,Tags=[{Key=cluster-name,Value=vonefs-cfv}]" --description "Internal security group for OneFS cluster vonefs-cfv"
```

Command output example:

```
{
  "GroupId": "sg-07f220483ab7e3bf7",
  "Tags": [
```

```

        {
            "Key": "cluster-name",
            "Value": "vonefs-cfv"
        }
    ]
}

```

2. Using the GroupId from the previous step to add ingress rules and egress rules to the security group allows all traffic between cluster internal network interfaces only. Running the following command by replacing the `<internal_sg_id>`, `<aws_region>` and `<cluster_name>` with your own settings to create the rule.

Create ingress rules:

```

aws ec2 authorize-security-group-ingress --group-id
<internal_sg_id> --source-group <internal_sg_id> --protocol
all --region <aws_region> --tag-specifications
"ResourceType=security-group-rule,Tags=[{Key=cluster-
name,Value=<cluster_name>}] "

```

Create egress rules:

```

aws ec2 authorize-security-group-egress --group-id
<internal_sg_id> --source-group <internal_sg_id> --protocol
all --region <aws_region> --tag-specifications
"ResourceType=security-group-rule,Tags=[{Key=cluster-
name,Value=<cluster_name>}] "

```

The following is an example to create the ingress rule and egress rule:

```

> aws ec2 authorize-security-group-ingress --group-id sg-
07f220483ab7e3bf7 --source-group sg-07f220483ab7e3bf7 --
protocol all --region us-east-1 --tag-specifications
"ResourceType=security-group-rule,Tags=[{Key=cluster-
name,Value=vonefs-cfv}] "
> aws ec2 authorize-security-group-egress --group-id sg-
07f220483ab7e3bf7 --source-group sg-07f220483ab7e3bf7 --
protocol all --region us-east-1 --tag-specifications
"ResourceType=security-group-rule,Tags=[{Key=cluster-
name,Value=vonefs-cfv}] "

```

AWS Management Console instructions

1. Sign into the AWS Management Console and open the VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, choose **Security Groups**, and then choose **Create security group**.
3. In the **Security group name** field, it is recommended that you specify a name with a format of `<cluster_name>-internal-sg` by replacing `<cluster_name>` with your own setting.
4. Type a short description for the **Description** field.
5. Choose your VPC ID in the **VPC** field.

6. Keep the **inbound rules** as default.
7. Keep the **outbound rules** as default.
8. Choose **Add new tag**. The tag key is **cluster-name**, and the tag value is your OneFS cluster name.
9. Choose **Create security group**.
10. Write down the security group ID. [Figure 6](#) is an example.

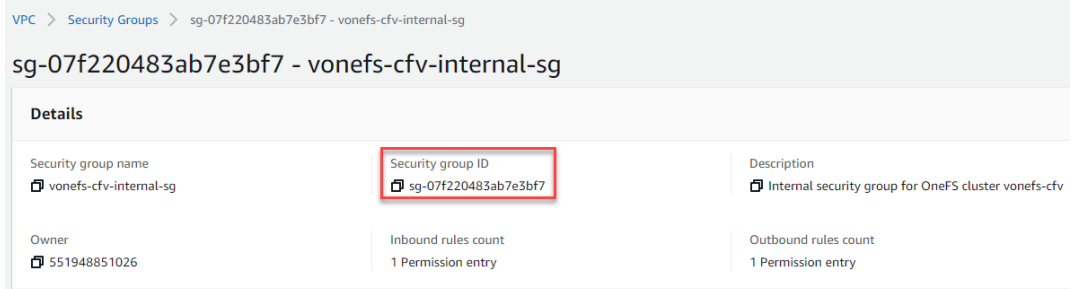


Figure 6. Internal security group ID

11. Choose **Actions**, and then choose **Edit inbound rules**. See [Figure 7](#) for an example.

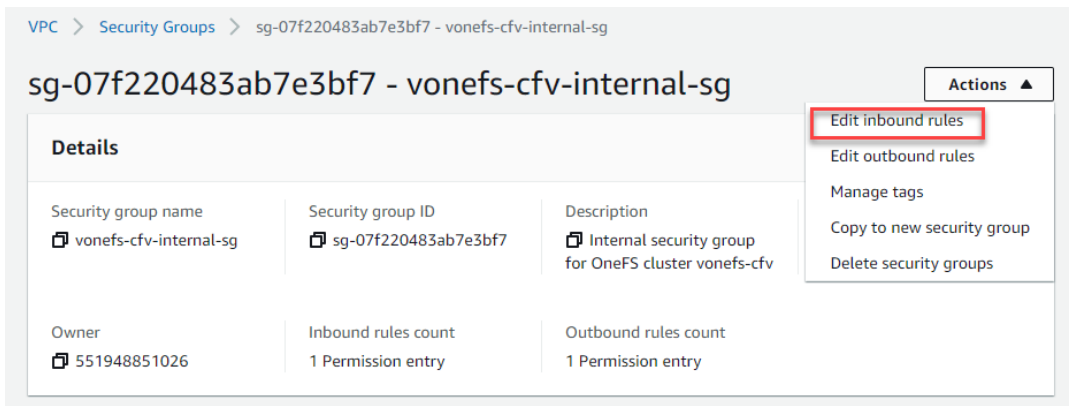


Figure 7. Edit inbound rules

12. Add an inbound rule to allow all traffic within the same internal security group ID, which is gathered in step 10. This rule only allows traffic between cluster internal network interfaces. See [Figure 8](#) for an example.

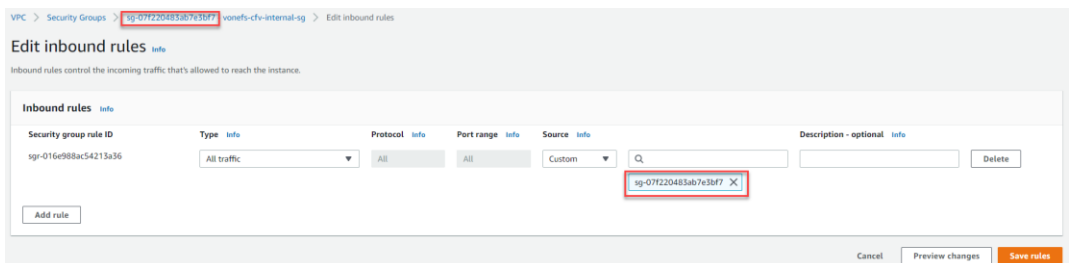


Figure 8. Internal security group inbound rules

13. Open the page of step 11 again and choose **Edit outbound rules**. Delete the default outbound rule, and add a new outbound rule to allow all traffic within the

same internal security group ID, which is gathered in step 10. This rule only allows traffic between cluster internal network interfaces. See [Figure 9](#) for an example.

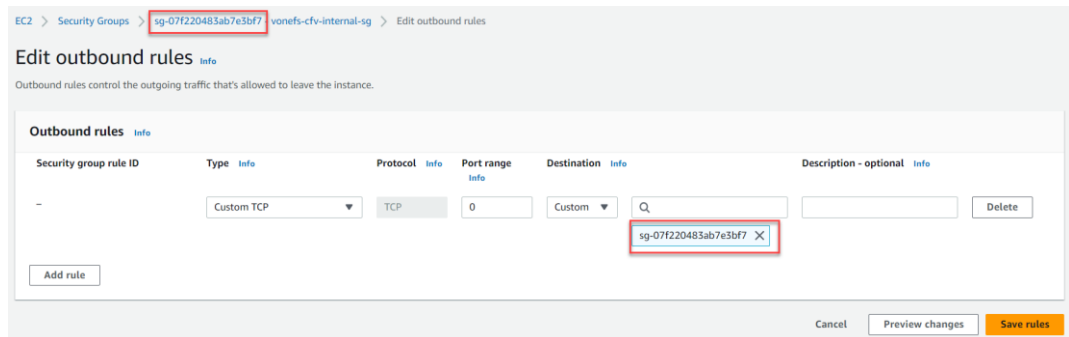


Figure 9. Internal security group outbound rules

Create an external security group

A security group is required to apply to the OneFS cluster external network interfaces.

AWS CLI instructions

1. Run the following command by replacing the `<aws_vpc_id>`, `<aws_region>` and `<cluster_name>` with your own settings.

```
aws ec2 create-security-group --vpc-id <aws_vpc_id> --group-name <cluster_name>-external-sg --region <aws_region> --tag-specifications "ResourceType=security-group,Tags=[{Key=cluster-name,Value=<cluster_name>}]" --description "External security group for OneFS cluster <cluster_name>"
```

Below is an example to create the external security group. Write down the GroupId in the output.

```
> aws ec2 create-security-group --vpc-id vpc-06639db65d7446720 --group-name vonefs-cfv-external-sg --region us-east-1 --tag-specifications "ResourceType=security-group,Tags=[{Key=cluster-name,Value=vonefs-cfv}]" --description "External security group for OneFS cluster vonefs-cfv"
```

Command output example:

```
{
  "GroupId": "sg-0f43cc7cb8a51cff1"
  "Tags": [
    {
      "Key": "cluster-name",
      "Value": "vonefs-cfv"
    }
  ]
}
```

2. Use the GroupId from the previous step to add ingress rules to the security group on the required ports. The details of the security group rules will depend on your

planned use case. For more information about creating an external security group, refer to the Network port usage section of [PowerScale OneFS 9.6 Security Configuration Guide](#). Below is an example used in this documentation, which allows ingress of the OneFS WebUI, NFS, SMB, S3, DNS, SSH traffic, and so on.

```
aws ec2 authorize-security-group-ingress --group-id <sg-id> --cidr=<gateway>/32 --port 67 --protocol udp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 3-4 --protocol icmp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 22 --protocol tcp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 53 --protocol tcp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 53 --protocol udp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 111 --protocol tcp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 111 --protocol udp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 135 --protocol tcp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 135 --protocol udp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 300 --protocol tcp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 300 --protocol udp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 302 --protocol tcp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 302 --protocol udp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 304 --protocol tcp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 304 --protocol udp --region us-east-1
```

```
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 305 --protocol tcp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 305 --protocol udp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 443 --protocol tcp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 445 --protocol tcp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 2049 --protocol tcp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 8080 --protocol tcp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 9020 --protocol tcp --region us-east-1
aws ec2 authorize-security-group-ingress --group-id sg-0f43cc7cb8a51cfff1 --cidr=0.0.0.0/0 --port 9021 --protocol tcp --region us-east-1
```

AWS Management Console instructions

1. Sign into the AWS Management Console and open the VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, choose **Security Groups**, and then choose **Create security group**.
3. In the **Security group name** field, it is recommended that you specify a name with a format of **<cluster_name>-external-sg** by replacing **<cluster_name>** with your own setting.
4. Type a short description for the **Description** field.
5. Choose your VPC ID in the **VPC** field.
6. Add **Inbound rules** to the security group on the required ports for the OneFS cluster. The details of the security group rules will depend on your planned use case. For more information about creating an external security group for your deployment, refer to the Network port usage section of [PowerScale OneFS 9.6 Security Configuration Guide](#). For test purposes, you can only allow traffic from a specific IPv4 CIDR.
7. Keep the **outbound rules** as the default.
8. Choose **Add new tag**. The tag key is **cluster-name**, and the tag value is your cluster name.
9. Choose **Create security group**.

Security group ID summary

It is recommended that you review and write down your security group information for reference later. Below is the example in this guide:

- Internal security group ID: sg-0f43cc7cb8a51cff1
- External security group ID: sg-07f220483ab7e3bf7

Create cluster internal network interfaces

Each cluster node requires an internal network interface. Create all internal network interfaces for your cluster nodes with the planned IPs in section [Prerequisites checklist details and example](#).

AWS CLI instructions

Run the following command by replacing the `<ip_addr>`, `<internal_subnet_id>`, `<internal_security_group_id>`, `<aws_region>`, `<cluster_name>`, and `<node_number>` with your own settings. To manage cluster nodes conveniently in the future, it is recommended that you add the two tags, which contain the cluster name and node number information.

```
aws ec2 create-network-interface --private-ip-address <ip_addr>
--subnet-id <internal_subnet_id> --groups
<internal_security_group_id> --region <aws_region> --tag-
specifications "ResourceType=network-interface,Tags=[{Key=cluster-
name,Value=<cluster_name>},{Key=Name,Value=<cluster_name>-
node<node_number>-int-a}]"
```

Below is an example to create the internal network interfaces for four nodes:

1. Create internal network interface for the first node - node1. Write down the NetworkInterfaceId in the output.

```
> aws ec2 create-network-interface --private-ip-address 10.0.0.20
--subnet-id subnet-0dc84f8a0a6f8dd8d --groups sg-07f220483ab7e3bf7
--region us-east-1 --tag-specifications "ResourceType=network-
interface,Tags=[{Key=cluster-name,Value=vonefs-
cfv},{Key=Name,Value=vonefs-cfv-node1-int-a}]"
```

2. Create the internal network interface for node2. Write down the NetworkInterfaceId in the output.

```
> aws ec2 create-network-interface --private-ip-address 10.0.0.21
--subnet-id subnet-0dc84f8a0a6f8dd8d --groups sg-07f220483ab7e3bf7
--region us-east-1 --tag-specifications "ResourceType=network-
interface,Tags=[{Key=cluster-name,Value=vonefs-
cfv},{Key=Name,Value=vonefs-cfv-node2-int-a}]"
```

3. Create the internal network interface for node3. Write down the NetworkInterfaceId in the output.

```
> aws ec2 create-network-interface --private-ip-address 10.0.0.22
--subnet-id subnet-0dc84f8a0a6f8dd8d --groups sg-07f220483ab7e3bf7
--region us-east-1 --tag-specifications "ResourceType=network-
interface,Tags=[{Key=cluster-name,Value=vonefs-
cfv},{Key=Name,Value=vonefs-cfv-node3-int-a}]"
```

4. Create the internal network interface for node4. Write down the NetworkInterfaceId in the output.

```
> aws ec2 create-network-interface --private-ip-address 10.0.0.23
--subnet-id subnet-0dc84f8a0a6f8dd8d --groups sg-07f220483ab7e3bf7
--region us-east-1 --tag-specifications "ResourceType=network-
interface,Tags=[{Key=cluster-name,Value=vonefs-
cfv},{Key=Name,Value=vonefs-cfv-node4-int-a}]"
```

AWS Management Console instructions

1. Sign into the AWS Management Console and open the EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the navigation pane, choose **Network Interfaces**.
3. Choose **Create network interface**.
4. For **Subnet**, select a subnet.
5. For **Private IPv4 address**, Choose **Custom** and enter an IP address of your cluster internal network interface. Figure 10 is an example to create the internal network interface for the first node in the cluster.

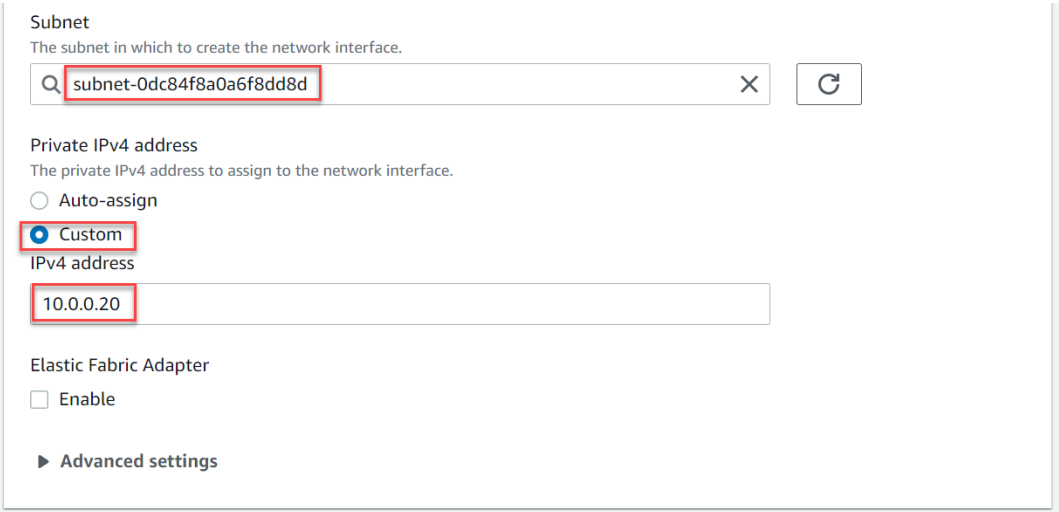


Figure 10. IP and subnet for first node internal interface

6. For **Security groups**, choose the cluster internal security group created in the section [Create the internal security group](#). Figure 11 is an example of how to create the internal network interface for the first node of the cluster.

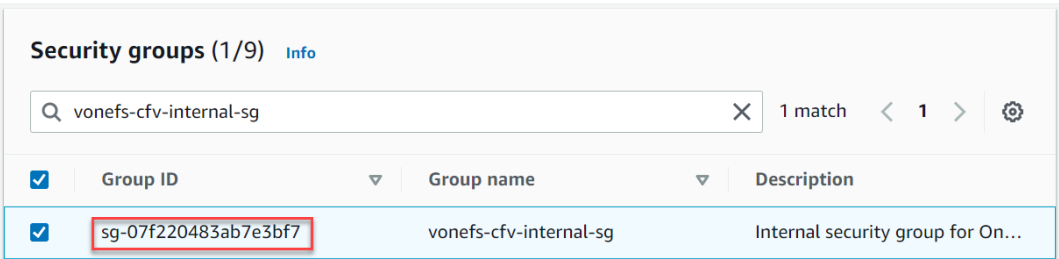


Figure 11. Security group for first node internal interface

7. Choose **Add new tag**. The tag key is **cluster-name**, and the tag value is your cluster name.
8. Choose **Add new tag** again. The tag key is **Name**, and the tag value is the network interface name. It is highly recommended that you add this tag value with the format **<cluster_name>-node<node_number>-int-a**, so that you can identify the network interface easily for different nodes later. Figure 12 is an example of how to create the internal network interface for the first node of the cluster.

Tags - optional

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - optional	
cluster-name	vonefs-cfv	Remove
Name	vonefs-cfv-node1-int-a	Remove

Add new tag

You can add 48 more tags

Figure 12. Tags for first node internal interface

9. Choose **Create network interface**. Write down the Network interface ID.
10. Repeat the previous Step 1 through Step 9 to create the remainder of the network interfaces for the other cluster nodes internal interfaces.

Create cluster external network interfaces

Each cluster node requires an external network interface. Create all external network interfaces for your cluster nodes with the planned IPs in section [Prerequisites checklist details and example](#).

AWS CLI instructions

Run the following command by replacing the **<ip_addr>**, **<external_subnet_id>**, **<external_security_group_id>**, **<aws_region>**, **<cluster_name>**, and **<node_number>** with your own settings. To manage cluster nodes conveniently in the future, it is recommended that you add the two tags that contain the cluster name and node number information.

```
aws ec2 create-network-interface --private-ip-address <ip_addr>
--subnet-id <external_subnet_id> --groups
<external_security_group_id> --region <aws_region> --tag-
specifications "ResourceType=network-interface,Tags=[{Key=cluster-
name,Value=<cluster_name>},{Key=Name,Value=<cluster_name>-
node<node_number>-ext-1}]"
```

Below is an example to create external network interfaces for 4 nodes:

1. Create an external network interface for the first node - node1. Write down the NetworkInterfaceId in the output.

```
> aws ec2 create-network-interface --private-ip-address 10.0.0.4 -
--subnet-id subnet-058548a2c6df9591c --groups sg-0f43cc7cb8a51cff1
--region us-east-1 --tag-specifications "ResourceType=network-
interface,Tags=[{Key=cluster-name,Value=vonefs-
cfv},{Key=Name,Value=vonefs-cfv-node1-ext-1}]"
```

2. Create the external network interface for node2. Write down the NetworkInterfaceId in the output.

```
> aws ec2 create-network-interface --private-ip-address 10.0.0.5 -
--subnet-id subnet-058548a2c6df9591c --groups sg-0f43cc7cb8a51cff1
--region us-east-1 --tag-specifications "ResourceType=network-
interface,Tags=[{Key=cluster-name,Value=vonefs-
cfv},{Key=Name,Value=vonefs-cfv-node2-ext-1}]"
```

3. Create an external network interface for node3. Write down the NetworkInterfaceId in the output.

```
> aws ec2 create-network-interface --private-ip-address 10.0.0.6 -
--subnet-id subnet-058548a2c6df9591c --groups sg-0f43cc7cb8a51cff1
--region us-east-1 --tag-specifications "ResourceType=network-
interface,Tags=[{Key=cluster-name,Value=vonefs-
cfv},{Key=Name,Value=vonefs-cfv-node3-ext-1}]"
```

4. Create the external network interface for node4. Write down the NetworkInterfaceId in the output.

```
> aws ec2 create-network-interface --private-ip-address 10.0.0.7 -
--subnet-id subnet-058548a2c6df9591c --groups sg-0f43cc7cb8a51cff1
--region us-east-1 --tag-specifications "ResourceType=network-
interface,Tags=[{Key=cluster-name,Value=vonefs-
cfv},{Key=Name,Value=vonefs-cfv-node4-ext-1}]"
```

AWS Management Console instructions

1. Sign into the AWS Management Console and open the EC2 console at <https://console.aws.amazon.com/ec2/>.
2. In the navigation pane, choose **Network Interfaces**.
3. Choose **Create network interface**.
4. For **Subnet**, select a subnet.
5. For **Private IPv4 address**, choose **Custom** and enter an IP address of your cluster external network interface. [Figure 13](#) is an example to create the external network interface for the first node of the cluster.

Subnet
The subnet in which to create the network interface.

subnet-058548a2c6df9591c

Private IPv4 address
The private IPv4 address to assign to the network interface.

☐ Auto-assign

☒ Custom

IPv4 address

10.0.0.4

Elastic Fabric Adapter

☐ Enable

► Advanced settings

Figure 13. IP and subnet for first node external interface

- For **Security groups**, choose the cluster external security group created in the section [Create an external security group](#). [Figure 14](#) is an example to create the external network interface for the first node of the cluster.

Security groups (1/9) [Info](#)

Search: vonefs-cfv-external-sg 1 match

<input checked="" type="checkbox"/>	Group ID	Group name	Description
<input checked="" type="checkbox"/>	sg-0f43cc7cb8a51cff1	vonefs-cfv-external-sg	External security group for On...

Figure 14. Security group for first node external interface

- Choose **Add new tag**. The tag key is **cluster-name**, and the tag value is your OneFS cluster name.
- Choose **Add new tag** again. The tag key is **Name**, and the tag value is the network interface name. It is highly recommended that you add this tag value with the format **<cluster_name>-node<node_number>-ext-1**, so that you can identify the network interface easily for different nodes later. [Figure 15](#) is an example to create the external network interface for the first node of the cluster.

Tags - optional

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - optional	
Q cluster-name X	Q vonefs-cfv X	Remove
Q Name X	Q vonefs-cfv-node1-ext-1 X	Remove

Add new tag

You can add 48 more tags

Figure 15. Tags for first node external interface

9. Choose **Create network interface**. Write down the Network interface ID.
10. Repeat the above steps 1 through step 9 to create the remainder of the network interfaces of the other cluster nodes external network interfaces.

Network interfaces ID summary

It is recommended that you review and write down your network interfaces information for reference later, as shown in [Table 7](#):

Table 7. Network interfaces ID summary

Network interfaces	Network interfaces ID
First node internal network interface ID	eni-044ba549fc8239cec
First node external network interface ID	eni-0e21d2c270a4b64de
Second node internal network interface ID	eni-0d947a68641d2db6f
Second node external network interface ID	eni-02929b093d9cfaccc
Third node internal network interface ID	eni-0371c901933c7ec28
Third node external network interface ID	eni-03aa64794d62b76af
Fourth node internal network interface ID	eni-09c1b9a66c08e969d
Fourth node external network interface ID	eni-0bf4fee125a58dffe

Prepare cluster nodes block device mapping

If you use AWS CLI to perform the deployment, the block device mapping json file is required.

OneFS uses EBS as data volumes. All EC2 instances in a cluster must share the same configuration of block device mapping. See [AWS block device mapping](#) documentation for more details.

To prepare the block device mapping json file, you can modify the content of section [block-device-mappings-template.json](#) by replacing the below items with your settings:

- **Volume size in GiB:** replace **<volume-size-in-GiB>** with your settings. Consider the [Supported cluster configuration](#), it should be 1024 to 16384 for gp3, 4096 or 10240 for st1.

- **Volume type:** replace `<volume_type>` with your settings. Consider the [Supported cluster configuration](#), the volume type should be gp3 or st1.

Save the modified content to the **block-device-mappings.json** file. In this guide, we assume the file path is **C:\json-files-template\block-device-mappings.json**

Note: the [block-device-mappings-template.json](#) content is a template for six volumes only; you can add more volumes as needed. You can also specify a throughput and IOPS configuration for gp3.

To enable EBS encryption, you need to add the following EBS options in the block device mapping based on your key type: AWS managed keys or customer managed keys.

- Encrypted: set to true to enable EBS encryption.
- KmsKeyId: required when using customer managed key. Set to your customer managed key ID.

Prepare the EC2 instance user data for the first node

OneFS requires user data in a json file for each node. This json file provides information needed to form a OneFS cluster for each node. You must prepare a **user-data-node-`<node_number>`.json** file for each node being deployed.

The first cluster node must contain an **acs_config** key, which will provide the node with information it needs to form a cluster. See the section [first-node-user-data-template.json](#) for details. [Table 8](#) shows the description about the content [first-node-user-data-template.json](#). Keep other options as the default if they are not listed in [Table 8](#).

The EC2 Instance user data passed to each node can only be accessed from within the node through AWS Instance Meta Data Service (IMDS). However, anyone who has direct access to the instance, and potentially any software running on the instance, can view this data. In order to protect sensitive data like password, we recommend cluster administrators grant `ISI_PRIV_LOGIN_SSH` to as few accounts as possible. To protect password, we also recommend using one of the following methods:

- Change the password when the cluster deployment is complete. See the OneFS documentation for [resetting a user password](#).
- Use a hashed password generated using openssl (sha256) and set the key "credentials_hashed" to true. The example below makes a sha256 hashed value for plaintext, replace `<password>` to your settings.

```
# openssl passwd -5 -salt `head -c 8 /dev/random | xxd -p`
<password>
```

The section [User data of first node example](#) is an example.

Table 8. User data for first node

json file key name	Description
hal_dongle_serialno	The serial number of the cluster node in the format SV200-930073-<device_id> where <device_id> is an integer sequence assigned to deployed nodes sequentially starting from 0 with 4 digits of 0 padding. Therefore, for the first node, it is SV200-930073-0000.
hal_volume_type	The EBS volume type for the cluster, gp3 or st1.
acs_config.cluster.password	The root password for the OneFS cluster
acs_config.cluster.admin_user_password	The admin password for the OneFS cluster
acs_config.cluster.credentials_hashed	Default is false. You can set this to true if you want to protect the root password and admin password in the user data.
acs_config.cluster.name	The OneFS cluster name. Cluster names must begin with a letter and can contain only numbers, letters, and hyphens. If the cluster is joined to an Active Directory domain, the cluster name must be 11 characters or fewer.
acs_config.cluster.nodes	Do not change, it should contain a single object with the serial number of the first node.
acs_config.cluster.timezone	The cluster time zone. Several available options: Greenwich Mean Time, Eastern Time Zone, Central Time Zone, Mountain Time Zone, Pacific Time Zone. You can change the time zone after the cluster is deployed by following the OneFS documentation – Set the cluster date and time .
acs_config.external_networking.dns_domains	This is required, it is the cluster DNS domain, deployment may fail if it is empty.
acs_config.external_networking.dns_servers	This is required, it is the cluster DNS servers, deployment may fail if it is empty.
acs_config.external_networking.external_interfaces.gateway	The cluster gateway
acs_config.external_networking.external_interfaces.ip_address_ranges.low	Set to the first IP address of the cluster external interfaces IP range.
acs_config.external_networking.external_interfaces.ip_address_ranges.high	Set to the last IP address of the cluster external interfaces IP range.
acs_config.external_networking.external_interfaces.netmask	The netmask of cluster external interfaces
acs_config.internal_networking.internal_interfaces.ip_address_ranges.low	Set to the first IP address of the cluster internal interfaces IP range.

json file key name	Description
acs_config.internal_networking. internal_interfaces.ip_address_ranges.high	Set to the last IP address of the cluster internal interfaces IP range.
acs_config.internal_networking. internal_interfaces.netmask	The netmask of cluster internal interfaces
devices[*].ext-1	The external IP address of each node
devices[*].int-a	The internal IP address of each node
devices[*].serial_number	The serial number of each node

Prepare the EC2 instance user data for additional nodes

The user data of the additional cluster nodes provides the nodes with the information they need to form a cluster. See the section [additional-node-user-data-template.json](#) for details. [Table 9](#) shows the description of [additional-node-user-data-template.json](#). Keep other options as the default if they are not listed in [Table 9](#).

The section [User data of additional nodes example](#) provides examples.

Table 9. User data for additional nodes

json file key name	Description
hal_dongle_serialno	The serial number of the cluster node in the format SV200-930073-<device_id> where <device_id> is an integer sequence assigned to deployed nodes sequentially starting from 0 with 4 digits of 0 padding. Therefore, it is SV200-930073-0001 for the second node, SV200-930073-0002 for the third node, and so on.
hal_volume_type	The EBS volume type for the cluster, gp3 or st1.
devices[*].ext-1	The external IP address of each node
devices[*].int-a	The internal IP address of each node
devices[*].serial_number	The serial number of each node

Create an interface endpoint

When you add an IP address to a node in the OneFS operating system, the node needs to make an EC2 API call to the AWS EC2 service to associate the IP address with the network interface as a secondary IP address. The following cluster configuration and features will require such an EC2 API call initiated from cluster nodes.

- **SmartConnect:** Once you configure the SmartConnect Service IP (SSIP) for a cluster, the cluster node makes an EC2 API call to assign the SSIP to the node as the AWS secondary IP address.
- **Add additional network pools:** When you add network pools, the IPs in the network pools are also assigned as the AWS secondary IP address on each node.
- **Dynamic IP pool failover:** Once an IP failover happens, OneFS moves the IP to a different node, which will re-assign the IP to a different node in AWS level. See more details about dynamic IP pool in the OneFS documentation [IP address allocation](#).

If your VPC has an Internet gateway, NAT device, VPN connection, or AWS Direct Connect connection, which allows your cluster nodes to communicate with AWS EC2 service directly, the EC2 API call initiated from cluster nodes works as expected.

However, if you are using a private VPC, which cannot communicate with AWS EC2 service, you need to create an interface endpoint for nodes to connect directly to the AWS EC2 services using private IP addresses, as if the EC2 service is hosted in the cluster VPC. See [AWS PrivateLink concepts](#) and [Create interface endpoint](#) for more details about AWS VPC Interface endpoints. See the following instructions to use the AWS CLI to create an endpoint for the VPC.

AWS CLI instructions

To create an endpoint for the VPC in which a cluster is deployed, run the following command by replacing `<aws_vpc_id>`, `<aws_region>`, `<external_subnet_id>`, and `<external_security_group_id>` with your own setting. Ensure that you have allowed the ingress HTTPS traffic on TCP port 443 for the interface endpoint in the security group.

```
aws ec2 create-vpc-endpoint --vpc-endpoint-type Interface --vpc-id
<aws_vpc_id> --region <aws_region> --service-name
com.amazonaws.<aws_region>.ec2 --subnet-ids <external_subnet_id> -
-security-group-ids <external_security_group_id>
```

The following is an example.

```
> aws ec2 create-vpc-endpoint --vpc-endpoint-type Interface --vpc-id
vpc-06639db65d7446720 --region us-east-1 --service-name
com.amazonaws.us-east-1.ec2 --subnet-ids subnet-058548a2c6df9591c
--security-group-ids sg-07f220483ab7e3bf7
```

AWS Management Console instructions

1. Open the Amazon VPC console at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, choose **Endpoints**.
3. Choose **Create endpoint**.
4. For **Service category**, choose **AWS services**.
5. For **Service name**, select the ec2 service. For example, **com.amazonaws.us-east-1.ec2** in this guide.

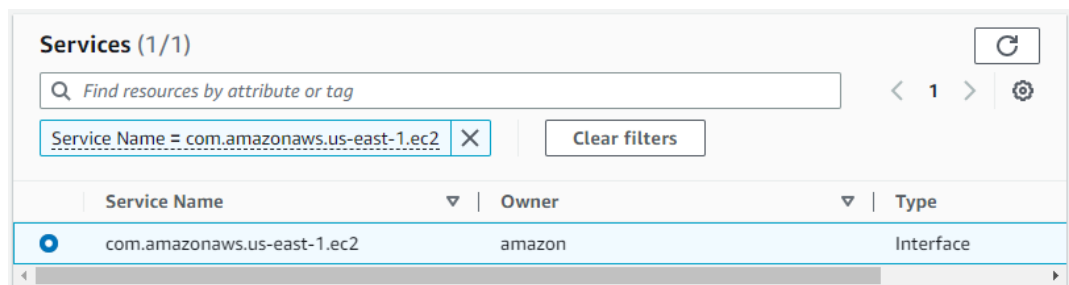


Figure 16. ec2 service

6. For **VPC**, select the VPC of the cluster.
7. For **Subnets**, select the external subnet of the cluster.

8. For **Security group**, select the security groups that allow the ingress HTTPS traffic on TCP port 443. The cluster external security group is used for this example.
9. For **Policy**, select **Full access**. We will be restricting the access through the policy created in the section [Create IAM policy, role, and instance profile](#).
10. (Optional) To add a tag, choose **Add new tag** and enter the tag key and the tag value.
11. Choose **Create endpoint**.

Create EC2 instances for the cluster

You have now completed most of the work required to deploy a cluster in AWS. This section demonstrates how you create cluster nodes based on the AWS resources in the previous sections.

Find the OneFS AMI ID

After you purchase APEX File Storage for AWS, you should be able to find the OneFS AMI image in the AWS Web Console by using the following steps:

1. Open the AWS Management Console.
2. Open the EC2 service portal and select the correct region.
3. Click the **AMI** under the **Images** menu item.
4. Change to **Private images** view.
5. Find the AMI image named **onefs-9.6.0.0** and write down the AMI image ID.

Figure 17 is an example of OneFS AMI ID **ami-0a76f79b6b95e8792**.

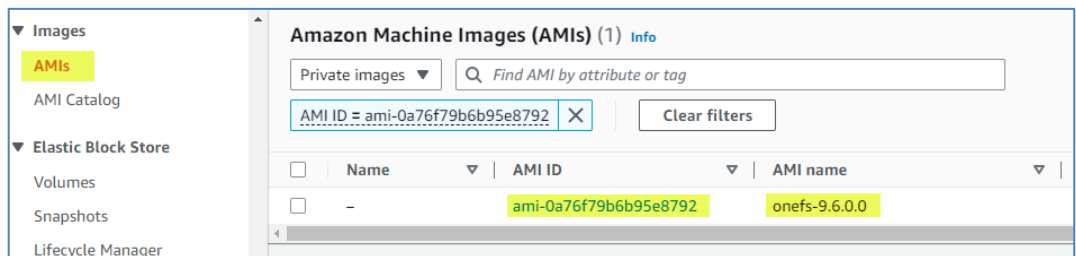


Figure 17. OneFS AMI image

AWS CLI instructions

Use the following steps to create the cluster nodes with the AWS CLI.

1. In this guide, we use **C:\json-files-template** as an example directory which stores all of the required .json files, including block device mappings and user data for each node. Open your OS CLI which has AWS CLI ready (Windows CMD in this example) and change your current directory to **C:\json-files-template**.

```
> cd C:\json-files-template
```

2. To create the cluster nodes, run the following command by replacing the **<onefs_ami_id>**, **<onefs_instance_type>**, **<cluster_name>**, **<internal_network_interface_id>**, **<external_network_interface_id>**, **<aws_region>**, and **<node_number>** with your settings. To manage cluster nodes

conveniently in the future, it is recommended that you add the two tags which contain the cluster name and node number information.

```
aws ec2 run-instances --image-id <onefs_ami_id> --instance-type
<onefs_instance_type> --placement GroupName=<cluster_name>-
onefs-placement-group --network-interfaces
NetworkInterfaceId=<internal_network_interface_id>,DeviceIndex=0
NetworkInterfaceId=<external_network_interface_id>,DeviceIndex=1
--region <aws_region> --iam-instance-profile Name=onefs-runtime-
instance-profile --user-data file://user-data-node-1.json --
block-device-mappings file://block-device-mappings.json --tag-
specifications "ResourceType=instance,Tags=[{Key=cluster-
name,Value=<cluster_name>},{Key=Name,Value=<cluster_name>-
node<node_number>}]"
```

The following example shows how to create cluster node1, node2, node3, and node4.

Create first node - node1:

```
> aws ec2 run-instances --image-id ami-0a76f79b6b95e8792 --
instance-type m5dn.12xlarge --placement GroupName=vonefs-cfv-
onefs-placement-group --network-interfaces NetworkInterfaceId=eni-
044ba549fc8239cec,DeviceIndex=0 NetworkInterfaceId=eni-
0e21d2c270a4b64de,DeviceIndex=1 --region us-east-1 --iam-instance-
profile Name=onefs-runtime-instance-profile --user-data
file://user-data-node-1.json --block-device-mappings file://block-
device-mappings.json --tag-specifications
"ResourceType=instance,Tags=[{Key=cluster-name,Value=vonefs-
cfv},{Key=Name,Value=vonefs-cfv-node1}]"
```

Create second node - node2:

```
> aws ec2 run-instances --image-id ami-0a76f79b6b95e8792 --
instance-type m5dn.12xlarge --placement GroupName=vonefs-cfv-
onefs-placement-group --network-interfaces NetworkInterfaceId=eni-
0d947a68641d2db6f,DeviceIndex=0 NetworkInterfaceId=eni-
02929b093d9cfaccc,DeviceIndex=1 --region us-east-1 --iam-instance-
profile Name=onefs-runtime-instance-profile --user-data
file://user-data-node-2.json --block-device-mappings file://block-
device-mappings.json --tag-specifications
"ResourceType=instance,Tags=[{Key=cluster-name,Value=vonefs-
cfv},{Key=Name,Value=vonefs-cfv-node2}]"
```

Create third node - node3:

```
> aws ec2 run-instances --image-id ami-0a76f79b6b95e8792 --
instance-type m5dn.12xlarge --placement GroupName=vonefs-cfv-
onefs-placement-group --network-interfaces NetworkInterfaceId=eni-
0371c901933c7ec28,DeviceIndex=0 NetworkInterfaceId=eni-
03aa64794d62b76af,DeviceIndex=1 --region us-east-1 --iam-instance-
profile Name=onefs-runtime-instance-profile --user-data
file://user-data-node-3.json --block-device-mappings file://block-
```

```
device-mappings.json --tag-specifications
"ResourceType=instance,Tags=[{Key=cluster-name,Value=vonefs-
cfv},{Key=Name,Value=vonefs-cfv-node3}]"
```

Create fourth node - node4:

```
> aws ec2 run-instances --image-id ami-0a76f79b6b95e8792 --
instance-type m5dn.12xlarge --placement GroupName=vonefs-cfv-
onefs-placement-group --network-interfaces NetworkInterfaceId=eni-
09c1b9a66c08e969d,DeviceIndex=0 NetworkInterfaceId=eni-
0bf4fee125a58dffe,DeviceIndex=1 --region us-east-1 --iam-instance-
profile Name=onefs-runtime-instance-profile --user-data
file:///user-data-node-4.json --block-device-mappings file:///block-
device-mappings.json --tag-specifications
"ResourceType=instance,Tags=[{Key=cluster-name,Value=vonefs-
cfv},{Key=Name,Value=vonefs-cfv-node4}]"
```

AWS Management Console instructions

Use the following steps to create the cluster nodes with the AWS Management Console.

1. Sign into the AWS Management Console and open the EC2 console at <https://console.aws.amazon.com/ec2/>.
2. Choose **Instances**, and then **Launch instances**.
3. For the **Name and tags**, choose **Add additional tags**.
4. The tag key is **Name**, and the tag value is instance name. It is highly recommended that you add this **Name** tag with the format **<cluster_name>-node<node_number>**, so that you can identify the node easily in the future for management.
5. Choose **Add tag**. The tag key is **cluster-name**, and the tag value is your OneFS cluster name. [Figure 18](#) is an example for the first node of the cluster.

▼ **Name and tags** [Info](#)

Key Info	Value Info	Resource types Info
<input type="text" value="cluster-name"/>	<input type="text" value="vonefs-cfv"/>	<input type="text" value="Select resource ty..."/> <input type="button" value="Instances"/>
<input type="text" value="Name"/>	<input type="text" value="vonefs-cfv-node"/>	<input type="text" value="Select resource ty..."/> <input type="button" value="Instances"/>

48 remaining (Up to 50 tags maximum)

Figure 18. Tags for cluster node

- For the **Application and OS Images (Amazon Machine Image)**, choose **My AMIs**, and then choose **Shared with me**. Finally select the OneFS AMI with the AMI ID in section [Find the OneFS AMI ID](#). [Figure 19](#) is an example.

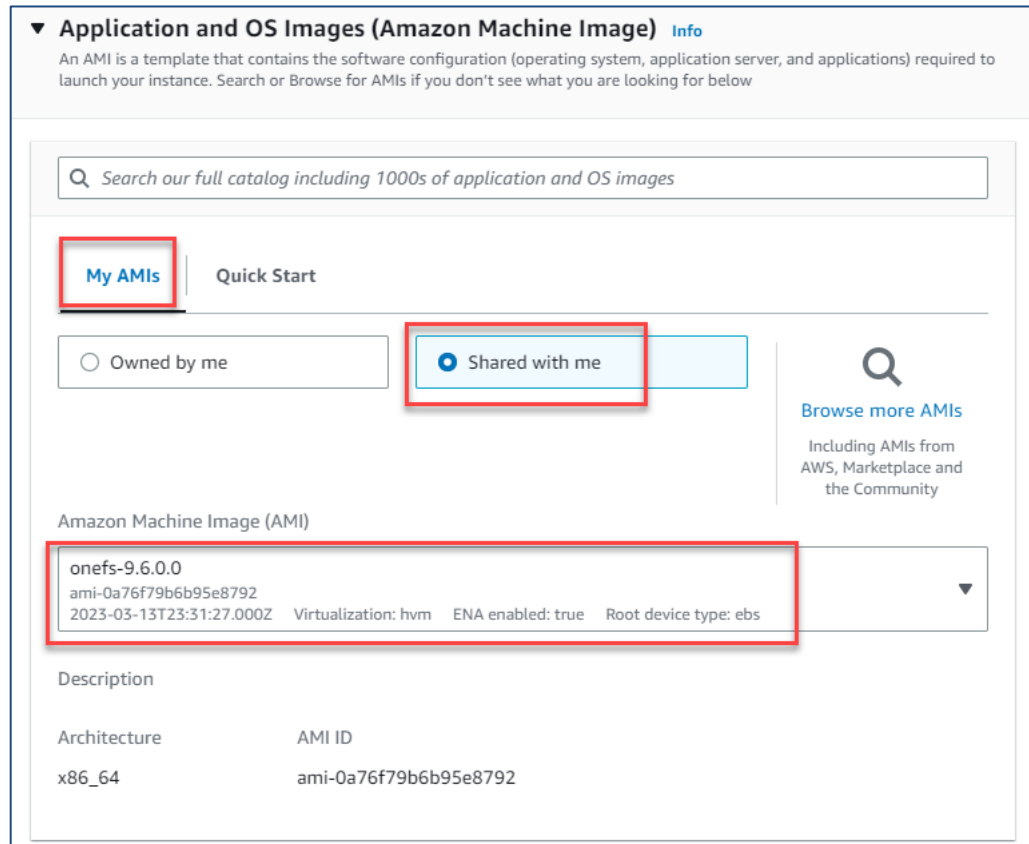


Figure 19. Choose OneFS AMI

- For **Instance type**, choose an instance type for your deployment. [Figure 20](#) is an example. All cluster nodes must use a same instance type.

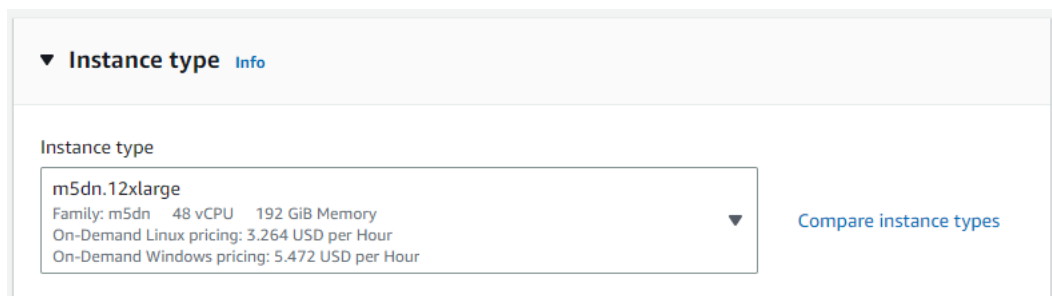


Figure 20. Instance type

- For the **Key pair (login)**, choose **Proceed without a key pair**.
- For **Network settings**, choose your VPC, and choose your subnet (**Note:** you should choose your cluster internal subnet). Choose **Select existing security group**, and then choose **Advanced network configuration**.

▼ Network settings [Info](#)

VPC - required [Info](#)

vpc-06639db65d7446720 (demo-vpc)
10.0.0.0/16

Subnet [Info](#)

subnet-0dc84f8a0a6f8dd8d onefs-cfv-internal-subnet
VPC: vpc-06639db65d7446720 Owner: 551948851026
Availability Zone: us-east-1b IP addresses available: 6 CIDR: 10.0.0.16/28

Auto-assign public IP [Info](#)

Disable

Firewall (security groups) [Info](#)
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☐ Create security group ☒ Select existing security group

Common security groups [Info](#)

Select security groups

Security groups that you add or remove here will be added to or removed from all your network interfaces.

► Advanced network configuration

Don't choose any security group here.

Figure 21. Network setting

- Under **Advanced network configuration**, choose the **cluster node internal network interface** which is created in section [Create cluster internal network interfaces](#). Then choose **Add network interface**. Figure 22 is an example for the cluster first node internal network interface. Ensure you add the internal network interface as the instance first network interface.

▼ Advanced network configuration

Network interface 1

Device index [Info](#)

0

Subnet [Info](#)

Select

Secondary IP [Info](#)

Select

IPv6 Prefixes [Info](#)

Select

Network card index [Info](#)

Select

The selected instance type does not support multiple network cards.

[Add network interface](#)

Network interface [Info](#)

eni-044ba549fc8239cec

Security groups [Info](#)

Select security groups

IPv6 IPs [Info](#)

Select

Delete on termination [Info](#)

Select

Elastic Fabric Adapter [Info](#)

☐ Enable

EFA is only compatible with certain instance types.

Description [Info](#)

Primary IP [Info](#)

IPv4 Prefixes [Info](#)

Select

Figure 22. Add internal network interface

11. Choose the **cluster node external network interface** which is created in section [Create cluster external network interfaces](#). The [Figure 23](#) is an example for the cluster first node external network interface.

Network interface 2

Device index [Info](#)

1

Subnet [Info](#)

Select

Secondary IP [Info](#)

Select

IPv6 Prefixes [Info](#)

Select

Network card index [Info](#)

Select

The selected instance type does not support multiple network cards.

[Add network interface](#)

Network interface [Info](#)

eni-0e21d2c270a4b64de

Security groups [Info](#)

Select security groups

IPv6 IPs [Info](#)

Select

Delete on termination [Info](#)

Select

Elastic Fabric Adapter [Info](#)

☐ Enable

EFA is only compatible with certain instance types.

Description [Info](#)

Primary IP [Info](#)

IPv4 Prefixes [Info](#)

Select

[Remove](#)

Figure 23. Add external network interface

12. For **Configure storage**, choose **Advanced**. Keep the default first 48GB gp3 OS volume and do not make any changes. The volume is used by the OneFS operating system. Then choose **Add new volume** to add data volumes for the cluster. See [Figure 24](#) for an example.

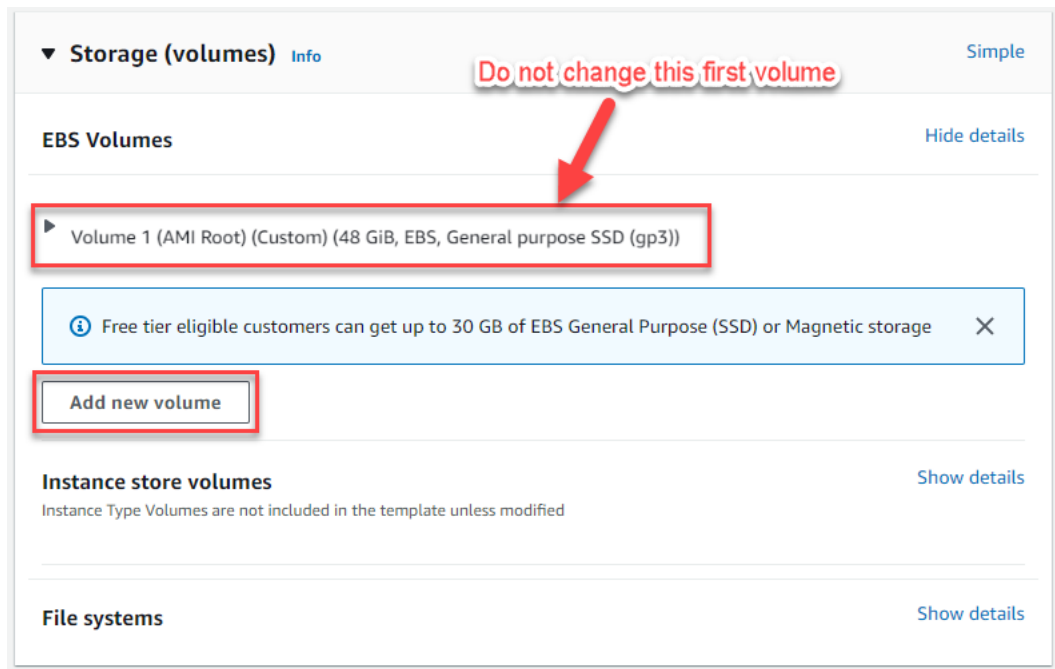


Figure 24. Advanced storage configuration

For the **volume 2**, which is the first data volume of the cluster node, specify a custom **Device name: xvda**, and then choose your planned volume size, volume type. Then choose **Yes** for the **Delete on termination** field. This option can help you to delete the volume automatically when you terminate your cluster node in the future. Otherwise, choose **No** if you want to keep the volume, even after you have terminated the cluster node. [Figure 25](#) is an example.

▼ Storage (volumes) Info

Simple

EBS Volumes

Hide details

▶ Volume 1 (AMI Root) (Custom) (48 GiB, EBS, General purpose SSD (gp3))

▼ Volume 2 (Custom)

Remove

Storage type Info

EBS

Device name - required Info

xvda

Snapshot Info

Select

Size (GiB) Info

1024

Volume type Info

gp3

IOPS Info

3000

Delete on termination Info

Yes

Encrypted Info

Not encrypted

KMS key Info

Select

Throughput Info

125

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage

Add new volume

Figure 25. Add first data volume

13. Choose **Add new volume** to add more volumes with the custom Device name: **xvdb**, **xvdc**, **xvdd**, **xvde**, **xvdf**, and so on. Ensure that all volumes use the same configuration for all cluster nodes. [Figure 26](#) is an example to add a second data volume.

Note: while adding volumes with the device names of xvdb, xvdc, and so on, you might see a warning: “The selected AMI does not support this device name.” This is expected behavior, and you can ignore it.

▼ Volume 3 (Custom) Remove

Storage type [Info](#)
EBS

Device name - *required* [Info](#)
xvdb ▼
⚠ The selected AMI does not support this device name.

Snapshot [Info](#)
Select ▼

Size (GiB) [Info](#)
1024

Volume type [Info](#)
gp3 ▼

IOPS [Info](#)
3000

Delete on termination [Info](#)
Yes ▼

Encrypted [Info](#)
Not encrypted ▼

KMS key [Info](#)
Select ▼
KMS keys are only applicable when encryption is set on this volume.

Throughput [Info](#)
125

ⓘ Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage ×

Add new volume

Figure 26. Add second data volume

14. The [Figure 27](#) is a volume example of the cluster first node.

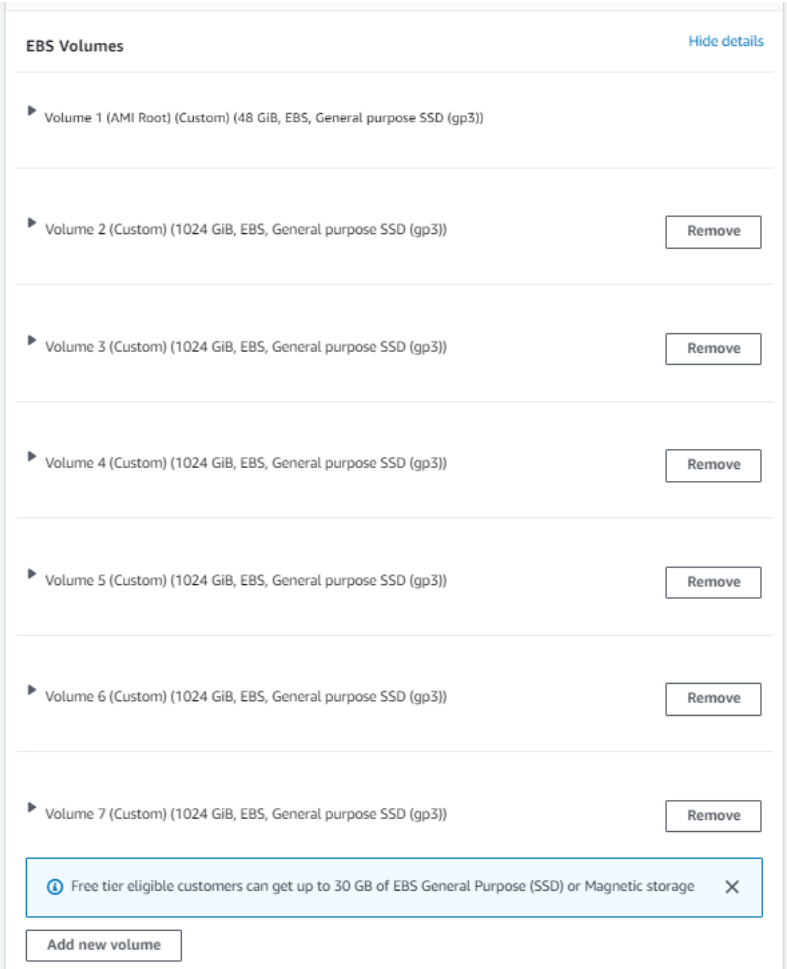


Figure 27. Data volumes of first node

15. Scroll down to **Advance Details**, choose the instance profile created in the section [Create IAM policy, role, and instance profile](#) for the **IAM instance profile** field and select placement group created in section [Create a placement](#) group for the **Placement group** field. [Figure 28](#) and [Figure 29](#) are examples.

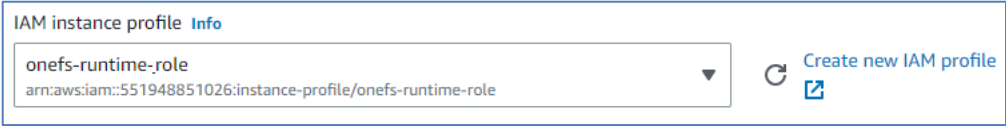


Figure 28. Add instance profile

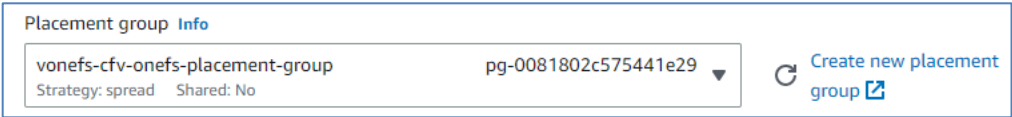


Figure 29. Add placement group

16. Scroll down to the end of **Advance Details**. For the **User data** field, copy and paste the user data of each node prepared in sections [Prepare the EC2 instance user data for the first](#) node and [Prepare the EC2 instance user data for additional nodes](#). [Figure 30](#) is an example of cluster first node.

Figure 30. User data for the cluster first nodeUser data - optional [Info](#)

Enter user data in the field.

```
{
  "hal_dongle_serialno": "SV200-930073-0000",
  "hal_is_first_node": true,
  "hal_volume_type": "gp3",
  "acs_config": {
    "cluster": {
      "admin_user_password": "Password01!",
      "cluster_name_nt4_compatibility": false,
      "encoding": "utf-8",
      "l3_cache": {
        "ssd_l3_cache_default_enabled": false
      },
      "name": "vonefs-cfv",
      "nodes": [
        {
          "serial_number": "SV200-930073-0000"
        }
      ],
      "password": "Password01!",
      "timezone": {
        "name": "Eastern Time Zone"
      }
    }
  }
}
```

☐ User data has already been base64 encoded

17. Click **Launch instance** to proceed.
18. Repeat the previous steps (Step 1 through Step18) to launch the remainder of the cluster nodes.

Set up the cluster

Now that you have completed the AWS infrastructure deployment, you need to access the command line of the cluster first node to complete the cluster formation. This can be accomplished in different ways, including:

- Use the AWS EC2 serial console of the first node
- SSH to the first node with external network interface IP

This guide uses the AWS EC2 serial console of the first node to set up the cluster. See the AWS documentation [Connect to the EC2 Serial Console](#) for more details. When you connect to the cluster first node, complete the cluster setup using the following steps.

Note: Cluster nodes can take several minutes to start up for the first time. If you get an error when connecting to and authenticating with OneFS, wait a few more minutes and then try again.

1. When you can connect to the cluster's first node, login to the node with root.
2. When you are connected, check the cluster health by running the following command until the cluster and node health report a status of OK. Note that the command may fail in various ways as the cluster is booting. If this occurs, wait several minutes, and then try again.

```
# isi status
```

```
vonefs-cfv-1# isi status
Cluster Name: vonefs-cfv
Cluster Health: [ OK ]
Data Reduction: 1.00 : 1
Storage Efficiency: 1.00 : 1
Cluster Storage: HDD                      SSD Storage
Size: 0 (0 Raw)                          5.9T (5.9T Raw)
VHS Size: 0
Used: 0 (n/a)                            411.2M (< 1%)
Avail: 0 (n/a)                          5.9T (> 99%)

ID | IP Address      Health Ext  Throughput (bps)  HDD Storage      SSD Storage
  |                |DASR|C/N|  In  Out  Total| Used / Size      |Used / Size
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1|10.0.0.4         |OK|C| 112.0| 0|112.0|(No Storage HDDs)| 411M/ 5.9T(< 1%)
-----+-----+-----+-----+-----+-----+-----+
Cluster Totals:           |112.0| 0|112.0|(No Storage HDDs)| 411M/ 5.9T(< 1%)

Health Fields: D = Down, A = Attention, S = Smartfailed, R = Read-Only
External Network Fields: C = Connected, N = Not Connected
```

Figure 31. First node status

3. Check that the authorization system is up by running the following command. Note that the command may fail in various ways as the cluster is booting. If this occurs, wait several minutes, and try again.

```
# isi auth user list
```

```
vonefs-cfv-1# isi auth user list
Name
-----
Guest
root
admin
compadmin
remotesupport
ese
ftp
insightiq
www
ntpd
_ypldap
tests
_lldpd
nobody
git_daemon
isdmgmt
-----
Total: 16
```

Figure 32. Authentication service status

- Wait for additional nodes to boot. Check that the additional nodes are ready to join by running the following command until they appear as available.

```
# isi devices node list
```

```
vonefs-cfv-1# isi devices node list
Serial Number      Product                                     Version      Status
-----
SV200-930073-0001  AWS-m5dn.4xlarge-Cloud-Single-64GB-1x1GE-6144GB SSD B_MAIN_3386R available
SV200-930073-0002  AWS-m5dn.4xlarge-Cloud-Single-64GB-1x1GE-6144GB SSD B_MAIN_3386R available
SV200-930073-0003  AWS-m5dn.4xlarge-Cloud-Single-64GB-1x1GE-6144GB SSD B_MAIN_3386R available
-----
Total: 3
```

Figure 33. Additional nodes status

- Add the additional nodes into the cluster one by one, starting from node 2. Run the following command to add the second node:

```
# isi devices node add SV200-930073-0001 --async
```

Check the cluster status by running the following command and wait for the cluster and node health to report a status of OK before you proceed.

```
# isi status
```

- Run the following command to add the third node:

```
# isi devices node add SV200-930073-0002 --async
```

Check the cluster status by running the following command and wait for the cluster and node health to report a status of OK before proceeding.

```
# isi status
```

- Run the following command to add the fourth node:

```
# isi devices node add SV200-930073-0003 --async
```

Check the cluster status by running the following command and wait for the cluster and node health to report a status of OK before proceeding.

```
# isi status
```

- Then wait for the node to join by watching the cluster and node status. Wait for the nodes to appear in the output and report a status of OK, and the cluster to return a status of OK.

```
# isi status
```

```
vonefs-cfv-1# isi status
Cluster Name: vonefs-cfv
Cluster Health: [ OK ]
Data Reduction: 1.02 : 1
Storage Efficiency: 0.34 : 1
Cluster Storage: HDD                      SSD Storage
Size:           0 (0 Raw)                  21.6T (23.8T Raw)
VHS Size:       2.2T
Used:           0 (n/a)                    1.3G (< 1%)
Avail:          0 (n/a)                    21.6T (> 99%)

ID | IP Address      Health Ext  Throughput (bps)  HDD Storage      SSD Storage
  |                |DASR|C/N|  In   Out  Total| Used / Size      |Used / Size
-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
1|10.0.0.4         | OK | C | 0|  0|  0| (No Storage HDDs) | 440M/ 5.4T (< 1%)
2|10.0.0.5         | OK | C | 0|16.8k|16.8k| (No Storage HDDs) | 443M/ 5.4T (< 1%)
3|10.0.0.6         | OK | C | 0|  0|  0| (No Storage HDDs) | 455M/ 5.4T (< 1%)
4|10.0.0.7         | OK | C | 0|797k|797k| (No Storage HDDs) | 6.6M/ 5.4T (< 1%)
-----+-----+-----+-----+-----+-----+-----+
Cluster Totals:      |  |  | 0| 813k| 813k| (No Storage HDDs) | 1.3G/21.6T (< 1%)

Health Fields: D = Down, A = Attention, S = Smartfailed, R = Read-Only
External Network Fields: C = Connected, N = Not Connected
```

Figure 34. Cluster status

Now, you have completed all steps to deploy an APEX File Storage for AWS cluster. It is highly recommended to refer to **APEX File Storage for AWS Getting Started Guide** in [Dell Technologies Support](#) website for more details about OneFS.

Expand a OneFS cluster

You can scale-out an existing cluster at any time to meet your business needs. This section shows how to add a new node into an existing cluster. We will use the 4-node cluster created in section [Set up](#) as an example.

1. Collect the existing cluster information, including:
 - Cluster name
 - AWS region
 - AWS availability zone
 - EC2 instance type
 - EC2 instance profile name
 - Placement group name
 - Internal subnet ID and external subnet ID
 - Internal security group ID and external security group ID
 - EBS configuration, including the block device mapping json file
 - OneFS AMI ID
2. Create the internal network interfaces with the same process described in section [Create cluster internal network interfaces](#).

Check if the IP address of the new internal network interface is present in the internal address range of the cluster using:

```
# isi config
# iprange
```

If the address is not in the range, expand the range to include the IP address of the newly created internal network interface

```
# iprange int-a <low_address>-<high_address>
# commit
```

3. Create the external network interface with the same process described in section [Create cluster external network interfaces](#).

Check if the IP address of the new external network interface is present in the IP Ranges of the default network pool using:

```
# isi network pool view groupnet0.subnet0.pool0
```

If the address is not in the IP ranges, add the IP address of the newly created external network interface using

```
# isi network pool modify groupnet0.subnet0.pool0 --add-ranges
<IP_address_of_the_new_interface>
```

4. Write down the new node's network interface IDs. Table 10 is the **Network Interface ID** examples for this guide.

Table 10. New node NetworkInterfaceId

Network interfaces	Network interfaces ID
Fifth node internal network interface ID	eni-0f3028b55cb03512a
Fifth node external network interface ID	eni-06ce6abe88aca7c47

Prepare the user data json file for the new node. Note that you must include the existing cluster node information along with the new node in the user data json file. The section [user-data-node-5.json](#) is an example

5. Create an EC2 instance with the same process described in the section [Create EC2 instances for the cluster](#).
6. Connect to the cluster first node. The user login to the cluster requires the following OneFS RBAC privileges to execute cluster commands used in this procedure.

- ISI_PRIV_LOGIN_SSH
- ISI_PRIV_DEVICES
- ISI_PRIV_NETWORK
- ISI_PRIV_STATISTICS

Check that the new node is ready to join by running the following command until the node appears as available.

```
# isi devices node list
```

7. Add the new node into the cluster. Run the following command to add the fifth node:

```
# isi devices node add SV200-930073-0004 -async
```

- Then wait for the node to join by watching the cluster and node status. Wait for the node to appear in the output and report a status of OK, and the cluster to return to a status of OK.

```
# isi status
```

```
vonefs-cfv-1# isi status
Cluster Name: vonefs-cfv
Cluster Health: [ OK ]
Data Reduction: 1.03 : 1
Storage Efficiency: 0.25 : 1
Cluster Storage: HDD                      SSD Storage
Size: 0 (0 Raw)                          21.6T (23.8T Raw)
VHS Size: 2.2T
Used: 0 (n/a)                            2.0G (< 1%)
Avail: 0 (n/a)                          21.6T (> 99%)
```

ID	IP Address	Health Ext		Throughput (bps)			HDD Storage		SSD Storage	
		DASR	C/N	In	Out	Total	Used / Size	Used / Size		
1	10.0.0.4	OK	C	78.3k	1.2M	1.3M	(No Storage HDDs)	709M/ 5.4T (< 1%)		
2	10.0.0.5	OK	C	287k	0	287k	(No Storage HDDs)	458M/ 5.4T (< 1%)		
3	10.0.0.6	OK	C	229k	942k	1.2M	(No Storage HDDs)	450M/ 5.4T (< 1%)		
4	10.0.0.7	OK	C	50.0k	521k	571k	(No Storage HDDs)	448M/ 5.4T (< 1%)		
5	10.0.0.8	OK	C	0	28.4k	28.4k	(No Storage HDDs)	592k/ 5.4T (< 1%)		
Cluster Totals:				689k	18.0M	18.7M	(No Storage HDDs)	2.0G/21.6T (< 1%)		

Health Fields: D = Down, A = Attention, S = Smartfailed, R = Read-Only
 External Network Fields: C = Connected, N = Not Connected

Figure 35. Expanded cluster status

Now, you have finished all steps to scale-out an existing cluster of APEX file Storage for AWS.

OneFS SmartConnect

OneFS SmartConnect uses the existing Domain Name Service (DNS) Server and provides a layer of intelligence within the OneFS software application. Connections are balanced across the cluster, which ensures optimal resource utilization and performance. You can set up SmartConnect for a cluster on AWS.

Based on your business requirement, you can leverage either an existing private DNS server or the AWS Route 53 Resolver to configure SmartConnect. This guide uses Route 53 Resolver as an example.

- Find an available IP address in the cluster external subnet as the cluster SmartConnect Service IP.
- Use SSH to connect to the cluster.
- Add the SmartConnect service IP (SSIP) by running the following command.

```
# isi network subnets modify groupnet0.subnet0 --sc-service-  
addr=<SSIP>
```

4. Ensure the SSIP is added successfully by running the following command. OneFS makes an EC2 API call to add the SSIP as the secondary IP of a node as explained in section [Create an interface endpoint](#).

```
# isi network interfaces list
```

5. Configure the SmartConnect FQDN.

```
# isi network pools modify groupnet0.subnet0.pool0 --sc-dns-zone=<smartconnect-fqdn>
```

6. Create the AWS Route 53 Resolver inbound endpoint and outbound endpoints for your VPC by following the [AWS documentation](#).
7. Add an AWS Route 53 Resolver rule, ensuring that the cluster SmartConnect FQDN, service IP, and VPC are included in the rule correctly.

The screenshot displays the 'powerscale-ve-sc-forwarder Configuration' page in the AWS Route 53 Resolver console. It shows the configuration for a Resolver rule with the following details:

- Domain name:** sc01.vlab.local (labeled 'Your OneFS cluster SmartConnect FQDN')
- VPCs (2):** A table listing two VPCs:

VPC ID	Name	Association ID	Status
vpc-06659db65c7446720	-	rsrv-rassoc-75929ac4104f4ad7a	Complete
vpc-076917046ab150b23	-	rsrv-rassoc-205cfa0236304c0ab	Complete

 (The first VPC is labeled 'Your OneFS cluster VPC')
- Target IP addresses (1):** A table listing one target IP address:

IPV4 address	Port
10.0.0.75	53

 (The IP address is labeled 'Your OneFS cluster SmartConnect Service IP')

Figure 36. Resolver rule for cluster SmartConnect

8. Verify that SmartConnect works correctly in a client of the VPC.

```

C:\Users\lieven>ping sc01.vlab.local

Pinging sc01.vlab.local [10.0.0.70] with 32 bytes of data:
Reply from 10.0.0.70: bytes=32 time<1ms TTL=64
Reply from 10.0.0.70: bytes=32 time<1ms TTL=64
Reply from 10.0.0.70: bytes=32 time=1ms TTL=64
Reply from 10.0.0.70: bytes=32 time<1ms TTL=64

Ping statistics for 10.0.0.70:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\Users\lieven>ping sc01.vlab.local

Pinging sc01.vlab.local [10.0.0.68] with 32 bytes of data:
Reply from 10.0.0.68: bytes=32 time<1ms TTL=64
Reply from 10.0.0.68: bytes=32 time<1ms TTL=64
Reply from 10.0.0.68: bytes=32 time<1ms TTL=64
Reply from 10.0.0.68: bytes=32 time<1ms TTL=64

Ping statistics for 10.0.0.68:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 0ms, Maximum = 0ms, Average = 0ms

```

Figure 37. Verify SmartConnect

Replace a volume for a cluster node

This section shows how to replace a data volume as needed for an existing cluster node.

Prior to replacing the EBS volume, we need to ensure that the volume is in the correct state. We need the node-lnn (node logical number) where the drive to be replaced is located and the bay that the drive resides in. Note that only volumes in the state of REPLACE are ready to be removed and replaced.

Table 11 lists the different drive states you can expect to see.

Table 11. Node drive state

Drive State	Description
HEALTHY	The drive is functioning correctly.
SMARTFAIL	The drive is in the process of being removed safely either because of an I/O error or by user request.
REPLACE	The drive was SmartFailed successfully and is ready to be removed.
PREPARING	The drive is undergoing a format operation. The drive state changes to HEALTHY when the format is successful.
JOURNAL	The drive is a journal drive and cannot be SmartFailed.
NEW	The drive is new and blank. This is the state that a drive is in when you run the isi dev command with the -a add option.
EMPTY	No drive assigned to that Bay.

Note: it may take quite some time (on the order of days) for a drive to finish SmartFailing. And note that only drives in the REPLACE state are ready to be replaced, and only drives in the HEALTHY state can be manually SmartFailed.

Before you make any changes on AWS resources, you should identify the drive information and state of the drive that is to be replaced with the following instructions.

1. Log in to the node where the drive to be replaced exists, and then run the following command to list the drives of the cluster.

```
# isi devices drive list
```

Example output:

```
vonefs-cfv-1# isi devices drive list --node-lnn=all
```

Lnn	Location	Device	Lnum	State	Serial
1	Bay 0	/dev/nvd8	N/A	JOURNAL	AWS33E14FDDB06375F68
1	Bay 1	/dev/nvd7	N/A	NEW	AWS229D6FB6CDDDB4A3D6
1	Bay 2	/dev/nvd6	1	HEALTHY	vol04da47796b537ebcc
1	Bay 3	/dev/nvd5	2	HEALTHY	vol0684d4b04d700f913
1	Bay 4	/dev/nvd4	3	HEALTHY	vol05f7251de4928ac0c
1	Bay 5	/dev/nvd3	4	HEALTHY	vol00caced3a6000567f
1	Bay 6	/dev/nvd2	5	HEALTHY	vol05b52418a379fca65
1	Bay 7	/dev/nvd1	0	HEALTHY	vol08197cfc7d89ad317
2	Bay 0	/dev/nvd8	N/A	JOURNAL	AWS2345F9AD5A8194936
2	Bay 1	/dev/nvd7	N/A	NEW	AWS31120833E6FC7289C
2	Bay 2	/dev/nvd6	1	HEALTHY	vol0c2af15523723c455
2	Bay 3	/dev/nvd5	2	HEALTHY	vol0152b5109b83c53e0
2	Bay 4	/dev/nvd4	3	HEALTHY	vol09ca884a23d551daf
2	Bay 5	/dev/nvd3	4	HEALTHY	vol0cd5c0b312be88f8d
2	Bay 6	/dev/nvd2	5	HEALTHY	vol02065cff1707a4b53
2	Bay 7	/dev/nvd1	0	HEALTHY	vol017b4aba1d8305bb7

Write down the following information:

- **Lnn:** logical node number that drive to be replaced in resides. For the example above this is 1, which indicates that this is the first node of the cluster.
 - **Location:** the location of a drive. For the example above, this is Bay 3.
 - **Serial:** Serial of the drive to be replaced, for the example above vol0684d4b04d700f913.
2. SmartFailing a HEALTHY drive. Typically drives that are to be replaced have failed and have already automatically been SmartFailed. Therefore, this step is not necessary other than for procedural testing purposes. Run the following command to SmartFail the drive highlighted in the previous step.

```
# isi devices drive smartfail --node-lnn 1 --bay 3
```

3. Before proceeding, wait for the drive to be in the REPLACE status

Next, we remove the drive from the node using either the AWS CLI or AWS Management Console

AWS CLI instructions

1. Find the cluster node instance ID, which you will use to perform the volume replacement. The instance ID **i-0b7c9dbe87cf06955** is used as an example.
2. The new drive must have an identical configuration as the SmartFailed drive. Therefore, before removing the drive from the node, collect the underlying EBS volume information of the node, including:
 - volume type
 - volume size
 - volume device name
 - volume availability zone
 - Delete on termination
 - IOPS (for gp3 only)
 - Throughput (for gp3 only)

To obtain the above EBS volume information, the volume ID will be necessary to query the volume. The volume ID can be obtained by inserting the "-" character after "vol" into the Serial obtained from the previous step. As an example, the drive Serial vol0684d4b04d700f913 represents the EBS volume ID vol-0684d4b04d700f913.

Now you can run the following AWS CLI to collect the EBS volumes.

```
# aws ec2 describe-volumes --volume-ids vol-0684d4b04d700f913
--output json --region us-east-1
```

Example output:

```
{
  "Volumes": [
    {
      "Attachments": [
        {
          "AttachTime": "2023-01-31T09:24:39+00:00",
          "Device": "xvdd",
          "InstanceId": "i-0b7c9dbe87cf06955",
          "State": "attached",
          "VolumeId": "vol-0684d4b04d700f913",
          "DeleteOnTermination": true
        }
      ],
      "AvailabilityZone": "us-east-1b",
      "CreateTime": "2023-01-31T09:24:39.948000+00:00",
      "Encrypted": false,
      "Size": 1024,
      "SnapshotId": "",
```



```

    "State": "in-use",
    "VolumeId": "vol-0684d4b04d700f913",
    "Iops": 3000,
    "VolumeType": "gp3",
    "MultiAttachEnabled": false,
    "Throughput": 125
  }
]
}

```

Write down the EBS volume information based on the previous output.

Table 12. Volume information

Volume options	Examples for this guide
Volume type	gp3
Volume size in GiB	1024
Volume device name	xvdd
Volume availability zone	us-east-1b
Delete on termination	True
Iops (for gp3 only)	3000
Throughput in MiB/s (for gp3 only)	125

- Now you can detach the EBS volume from the cluster node by running the following AWS CLI.

```
# aws ec2 detach-volume --volume-id vol-0684d4b04d700f913 --
instance-id i-0b7c9dbe87cf06955 --region us-east-1
```

Example output:

```

{
  "AttachTime": "2023-01-31T09:24:39+00:00",
  "Device": "xvdd",
  "InstanceId": "i-0b7c9dbe87cf06955",
  "State": "detaching",
  "VolumeId": "vol-0684d4b04d700f913"
}

```

- Log in to the cluster node, and verify the drive on the cluster is no longer present by running the following OneFS CLI:

```
# isi devices drive list
```

Below is the example for this guide.

```

vonefs-cfv-1# isi devices drive list --node-lnn=1
Lnn  Location  Device    Lnum  State   Serial
-----
1    Bay 0    /dev/nvd8 N/A    JOURNAL AWS33E14FDDB06375F68
1    Bay 1    /dev/nvd7 N/A    NEW     AWS229D6FB6CDD4A3D6

```

```

1   Bay 2   /dev/nvd6 1   HEALTHY vol04da47796b537ebcc
1   Bay 3   -           N/A   REPLACE
1   Bay 4   /dev/nvd4 3   HEALTHY vol05f7251de4928ac0c
1   Bay 5   /dev/nvd3 4   HEALTHY vol00caced3a6000567f
1   Bay 6   /dev/nvd2 5   HEALTHY vol05b52418a379fca65
1   Bay 7   /dev/nvd1 0   HEALTHY vol08197cfc7d89ad317

```

5. Delete the unused and detached EBS volume by running the following AWS CLI.

```
# aws ec2 delete-volume --volume-id vol-0684d4b04d700f913 --
region us-east-1
```

Query the volume to ensure it was deleted and no longer exists:

```
# aws ec2 describe-volumes --volume-ids vol-0123456789abcdef
--output text --region us-east-1
```

Example output:

An error occurred (InvalidVolume.NotFound) when calling the DescribeVolumes operation: The volume 'vol-0684d4b04d700f913' does not exist.

6. Using the volume information collected in step 2, create an EBS volume for replacement. Below is an example. For more information, see the AWS documentation about the [create-volume](#) command.

```
# aws ec2 create-volume --availability-zone us-east-1b --size
1024 --volume-type gp3 --iops 3000 --throughput 125 --region
us-east-1
```

Example output, write down the new EBS volume ID vol-0765111ed49661be2.

```
{
  "AvailabilityZone": "us-east-1b",
  "CreateTime": "2023-03-29T08:31:02+00:00",
  "Encrypted": false,
  "Size": 1024,
  "SnapshotId": "",
  "State": "creating",
  "VolumeId": "vol-0765111ed49661be2",
  "Iops": 3000,
  "Tags": [],
  "VolumeType": "gp3",
  "MultiAttachEnabled": false,
  "Throughput": 125
}
```

7. Attach the new EBS volume to the cluster node with device name collected in step 2.

```
# aws ec2 attach-volume --volume-id vol-0765111ed49661be2 --
instance-id i-0b7c9dbe87cf06955 --device xvdd --region us-
east-1
```

Highly recommended: Using the instance-id, and the DeviceName of the EBS volume, modify the EBS volume attribute DeleteOnTermination and set to true. This will indicate that the EBS volume will be deleted upon instance termination to avoid having leftover, unattached EBS volumes when the node (EC2 instance) is terminated.

```
# aws ec2 modify-instance-attribute --instance-id i-
0b7c9dbe87cf06955 --block-device-mappings "[{\\"DeviceName\\":
\\"xvdd\\",\\"Ebs\\":{\\"DeleteOnTermination\\":true}}]" --region
us-east-1
```

Before proceeding, check that the EBS Volume has finished attaching from the AWS side by verifying that the State is in-use:

```
# aws ec2 describe-volumes --volume-ids vol-0765111ed49661be2
--region us-east-1
```

8. Finally, log in to the cluster node, and verify the drive is present on the cluster node by running the following OneFS CLI:

```
# isi devices drive list
```

Below is an example for this guide.

```
vonefs-cfv-1# isi devices drive list --node-lnn=1
Lnn  Location  Device      Lnum  State   Serial
-----
1    Bay 0    /dev/nvd8  N/A    JOURNAL AWS33E14FDDB06375F68
1    Bay 1    /dev/nvd7  N/A    NEW      AWS229D6FB6CDDDB4A3D6
1    Bay 2    /dev/nvd6  1      HEALTHY  vol04da47796b537ebcc
1    Bay 3    /dev/nvd5  2      HEALTHY  vol0765111ed49661be2
1    Bay 4    /dev/nvd4  3      HEALTHY  vol05f7251de4928ac0c
1    Bay 5    /dev/nvd3  4      HEALTHY  vol00caced3a6000567f
1    Bay 6    /dev/nvd2  5      HEALTHY  vol05b52418a379fca65
1    Bay 7    /dev/nvd1  0      HEALTHY  vol08197cfc7d89ad317
```

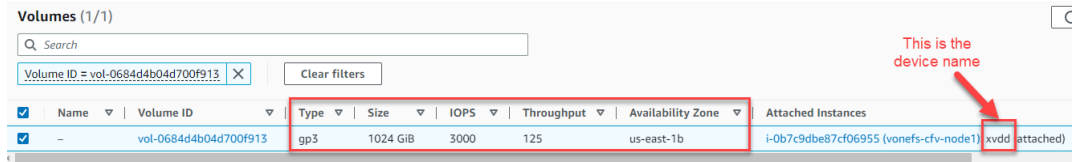
AWS Management Console instructions

1. Find the cluster node instance ID, which you will use to perform the volume replacement. The instance ID **i-0b7c9dbe87cf06955** is used as an example.
2. Sign in to the AWS Management Console and open the EC2 console at <https://console.aws.amazon.com/ec2/>.
3. Choose **Elastic Block Store**, and then **Volumes**.
4. The new drive must have an identical configuration with the SmartFailed drive. Therefore, before removing the drive from the node, collect the underlying EBS volume information of the node, including:
 - volume type
 - volume size
 - volume device name
 - volume availability zone

- lops (for gp3 only)
- Throughput (for gp3 only)

To obtain the above EBS volume information, the volume ID will be necessary to query the volume. The volume ID can be obtained by inserting the "-" character after "vol" into the Serial obtained from the previous step. As an example, the drive Serial vol0684d4b04d700f913 represents the EBS volume ID vol-0684d4b04d700f913.

Use the filter "Volume ID = vol-0684d4b04d700f913" to obtain the volume details. Figure 38 is an example.



Name	Volume ID	Type	Size	IOPS	Throughput	Availability Zone	Attached Instances
-	vol-0684d4b04d700f913	gp3	1024 GiB	3000	125	us-east-1b	i-0b7c9dbe87cf06955 (vonefs-cfv-node1) xvdd attached

Figure 38. EBS volume information

Write down the EBS volume information based on the above output.

Table 13. Volume information

Volume options	Examples for this guide
Volume type	gp3
Volume size in GiB	1024
Volume device name	xvdd
Volume availability zone	us-east-1b
lops (for gp3 only)	3000
Throughput in MiB/s (for gp3 only)	125

5. Select the EBS volume in step 5, then choose **Actions**, and finally choose **Detach volume**.
6. Log in to the cluster node, and verify the drive on the cluster is no longer present by running the following OneFS CLI:

```
# isi devices drive list
```

Below is an example.

```
vonefs-cfv-1# isi devices drive list --node-lnn=1
Lnn  Location  Device  Lnum  State  Serial
-----
1    Bay 0    /dev/nvd8  N/A    JOURNAL  AWS33E14FDDDB06375F68
1    Bay 1    /dev/nvd7  N/A    NEW      AWS229D6FB6CDDDB4A3D6
1    Bay 2    /dev/nvd6  1      HEALTHY  vol104da47796b537ebcc
1    Bay 3    -          N/A    REPLACE
1    Bay 4    /dev/nvd4  3      HEALTHY  vol105f7251de4928ac0c
1    Bay 5    /dev/nvd3  4      HEALTHY  vol100caced3a6000567f
1    Bay 6    /dev/nvd2  5      HEALTHY  vol105b52418a379fca65
1    Bay 7    /dev/nvd1  0      HEALTHY  vol108197cfc7d89ad317
```

7. Return to the EC2 volume console to delete the volume. Select the EBS volume in step 5, then choose **Actions**, and finally choose **Delete volume**.
8. Next, using the volume information collected in step 4, create an EBS volume for replacement. [Figure 39](#) is an example.

EC2 > Volumes > Create volume

Create volume [Info](#)

Create an Amazon EBS volume to attach to any EC2 instance in the same Availability Zone.

Volume settings

Volume type [Info](#)

General Purpose SSD (gp3) ▼

Size (GiB) [Info](#)

1024

Min: 1 GiB, Max: 16384 GiB. The value must be an integer.

IOPS [Info](#)

3000

Min: 3000 IOPS, Max: 16000 IOPS. The value must be an integer.

Throughput (MiB/s) [Info](#)


125

Min: 125 MiB, Max: 1000 MiB. Baseline: 125 MiB/s.

Availability Zone [Info](#)

us-east-1b ▼

Snapshot ID - optional [Info](#)

Don't create volume from a snapshot ▼ 

Encryption [Info](#)

Use Amazon EBS encryption as an encryption solution for your EBS resources associated with your EC2 instances.

☐ Encrypt this volume

Figure 39. Create an EBS volume


9. Attach the new EBS volume to the cluster node with the device name collected in step 4. [Figure 40](#) is an example.

EC2 > Volumes > vol-0765111ed49661be2 > Attach volume


Attach volume [Info](#)

Attach a volume to an instance to use it as you would a regular physical hard disk drive.

Basic details

Volume ID
 [vol-0765111ed49661be2](#)


Availability Zone
us-east-1b

Instance [Info](#)
 

Only instances in the same Availability Zone as the selected volume are displayed.

Device name [Info](#)

Recommended device names for Linux: /dev/sda1 for root volume. /dev/sd[f-p] for data volumes.



Newer Linux kernels may rename your devices to **/dev/xvdf** through **/dev/xvdp** internally, even when the device name entered here (and shown in the details) is **/dev/sdf** through **/dev/sdp**.

Figure 40. Attach the EBS volume

Highly recommended: Using the instance-id, and the DeviceName of the EBS volume, modify the EBS volume attribute DeleteOnTermination set to true. This will indicate that the EBS volume will be deleted upon instance termination to avoid having leftover, unattached EBS volumes when the node (EC2 instance) is terminated.

```
# aws ec2 modify-instance-attribute --instance-id i-
0b7c9dbe87cf06955 --block-device-mappings "[{\"DeviceName\":
\\\"xvdd\\\", \\\"Ebs\\\": {\\\"DeleteOnTermination\\\": true}}]" --region
us-east-1
```

Note: There is no option available from the AWS Management Console to modify the EBS volume DeleteOnTermination attribute. We have to use above AWS CLI command for that.

Before proceeding, check that the EBS Volume has finished attaching from the AWS side by verifying that the State is In-use:

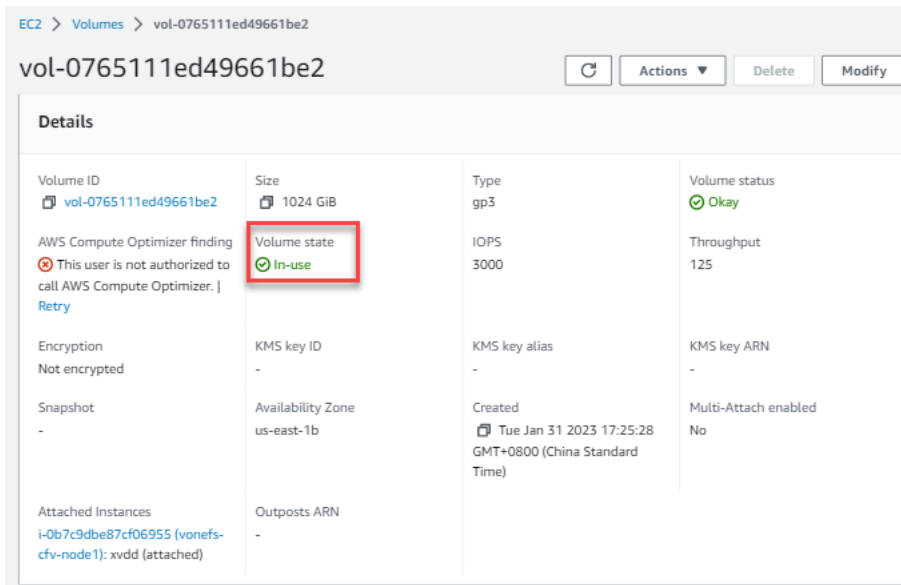


Figure 41. Verify volume state

- Finally, log in to the cluster node, and verify that the drive is present on the cluster node by running the following OneFS CLI:

```
# isi devices drive list
```

Below is an example.

```
vonefs-cfv-1# isi devices drive list --node-lnn=1
Lnn  Location  Device    Lnum  State   Serial
-----
1    Bay 0    /dev/nvd8 N/A    JOURNAL AWS33E14FDDB06375F68
1    Bay 1    /dev/nvd7 N/A    NEW     AWS229D6FB6CDDDB4A3D6
1    Bay 2    /dev/nvd6 1     HEALTHY vol04da47796b537ebcc
1    Bay 3    /dev/nvd5 2     HEALTHY vol0765111ed49661be2
1    Bay 4    /dev/nvd4 3     HEALTHY vol05f7251de4928ac0c
1    Bay 5    /dev/nvd3 4     HEALTHY vol00caced3a6000567f
1    Bay 6    /dev/nvd2 5     HEALTHY vol05b52418a379fca65
1    Bay 7    /dev/nvd1 0     HEALTHY vol08197cfc7d89ad317
```

Replace a cluster node

You can replace a node of an existing cluster when the node is not working. This section shows how to replace a cluster node. We will use the 5-nodes cluster expanded in section [Expand a OneFS cluster](#) as example by replacing the fifth node in the cluster.

- Collect the existing cluster information firstly, including:
 - Cluster name
 - AWS region
 - AWS availability zone
 - EC2 instance type
 - EC2 instance profile name

- Placement group name
 - Internal subnet ID and external subnet ID
 - Internal security group ID and external security group ID
 - EBS configuration, including the block device mapping json file
 - OneFS AMI ID
2. Connect to the cluster and use the following command to Smartfail the node that needs to be replaced. Below is an example.

```
# isi devices node smartfail --node-lnn 5
```

It may take some time to complete the Smartfail operation. You may check the status using the command

```
# isi status -q
```

Wait until the node disappear from the cluster.

3. Terminate the cluster node.
4. Prepare network interfaces.
 - If the “Delete on instance termination” behavior is disabled for network interfaces, you can choose to reuse the terminated node’s internal network interface and external network interface, then you can go to step 5 directly. We will reuse the interfaces for this guide.
 - If you choose to delete the terminated node’s internal network interface and external network interface, then you must create internal network interfaces with the same process described in the section [Create cluster internal network interfaces](#).

Remove the IP address of the old network interface from the cluster default network pool using:

```
# isi net pools modify groupnet0.subnet0.pool0 --remove-ranges <IP_address_of_the_old_interface >
```

Check if the IP address of the new internal network interface is present in the internal address range of the cluster using:

```
# isi config
>> iprange
```

If the address is not in the range, expand the range to include the IP address of the newly created internal network interface

```
>> iprange int-a <low_address>-<high_address>
>> commit
```

And then create an external network interface with the same process described in the section [Create cluster external network interfaces](#).

Check if the IP address of the new external network interface is present in the IP Ranges of the default network pool using:


```
# isi network pool view groupnet0.subnet0.pool0
```

If the address is not in the IP ranges, add the IP address of the newly created external network interface using

```
# isi network pool modify groupnet0.subnet0.pool0 --add-ranges <IP_address_of_the_new_interface>
```

5. Prepare the user data json file for the new node. Note that you must include the existing cluster node information since you will replace the fifth node. The section [user-data-node-5.json](#) is an example.
6. Create an EC2 instance using the same process described in the section [Create EC2 instances for the cluster](#).
7. Connect to the cluster and wait for the new node to boot. Check that the new node is ready to join by running the following command and wait until they appear as available.

```
# isi devices node list
```

8. Add the new node into the cluster. Run the following command to add the fifth node:

```
# isi devices node add SV200-930073-0004 -async
```

9. Wait for the node to join by watching the cluster and node status. Wait for the node to appear in the output and report a status of OK, and the cluster to return to a status of OK.

```
# isi status
```

```
vonefs-cfv-1# isi status
Cluster Name: vonefs-cfv
Cluster Health: OK
Data Reduction: 1.00 : 1
Storage Efficiency: 0.20 : 1
Cluster Storage: HDD                      SSD Storage
Size: 0 (0 Raw)                          296.4G (444.7G Raw)
VHS Size: 148.3G
Used: 0 (n/a)                            2.9G (< 1%)
Avail: 0 (n/a)                            293.5G (> 99%)
```

ID	IP Address	Health	Ext	Throughput (bps)			HDD Storage	SSD Storage
		DASR	C/N	In	Out	Total	Used / Size	Used / Size
1	10.0.0.4	OK	C	0	0	0	(No Storage HDDs)	499M/59.3G(< 1%)
2	10.0.0.5	OK	C	0	70.6k	70.6k	(No Storage HDDs)	771M/59.3G(1%)
3	10.0.0.6	OK	C	0	23.5k	23.5k	(No Storage HDDs)	487M/59.3G(< 1%)
4	10.0.0.7	OK	C	0	13.4k	13.4k	(No Storage HDDs)	675M/59.3G(1%)
5	10.0.0.8	OK	C	0	0	0	(No Storage HDDs)	566M/59.3G(< 1%)
Cluster Totals:				0	108k	108k	(No Storage HDDs)	2.9G/ 296G(< 1%)

```
Health Fields: D = Down, A = Attention, S = Smartfailed, R = Read-Only
External Network Fields: C = Connected, N = Not Connected
```

Finally, wait for the FlexProtectLin job event to succeed. Use the following command to list the recent job events history in the cluster. Wait until the most recent run of FlexProtectLin job events show completion with a "Succeeded" status.

```
# isi job events list
```

You have now finished all the steps necessary to replace an existing cluster node.

Maintenance events from AWS

AWS can schedule maintenance events for your instances, such as a reboot, stop/start, or retirement. See the AWS documentation [Scheduled events for your instances](#) for more details. OneFS monitor scheduled maintenance events periodically from every node through IMDS and performs the required actions. OneFS will also generate CELOG events with the event details. To complete some of the actions, OneFS may shut down the affected node and user needs to start the instance back.

Table 14 contains a list of AWS maintenance events, action taken by OneFS to resolve, type of CELOG events generated by OneFS, and the expected user actions.

Table 14. AWS maintenance events and OneFS actions

AWS event type	AWS event description	OneFS action	OneFS CELOG event	User action
Instance stop	At the scheduled time, the instance is stopped. When you start it again, it is migrated to a new host. Applies only to instances backed by AWS EBS.	Shutdown node	Node is scheduled to shutdown due to host hardware failure.	Manually start the instance
Instance retirement	At the scheduled time, the instance is stopped if it is backed by AWS EBS, or terminated if it is backed by instance store.	Shutdown node	Node is scheduled to shutdown due to host hardware failure.	Manually start the instance
Instance reboot	At the scheduled time, the instance is rebooted.	Reboot node	Node is scheduled to reboot following maintenance.	No manual action required
System reboot	At the scheduled time, the host for the instance is rebooted.	Shutdown node	Host is scheduled to reboot following maintenance.	Manually start the instance
System maintenance	At the scheduled time, the instance might be temporarily affected by network maintenance or power maintenance.	Shutdown node	Host is scheduled to reboot following maintenance.	Manually start the instance

Stopping OneFS node instances

OneFS nodes use one of the instance store volumes as the journal device. OneFS needs to save the content of the journal before stopping an instance since any data on the volume is lost when the instance is stopped. OneFS nodes use EBS volume as their root device and save the contents of the journal to the EBS volume during nodes' shutdown.

When stopping an instance, while you can do so from the AWS Management Console directly, the best practice is to shut the instance down gracefully, by using the OneFS CLI command `isi cluster shutdown --node-lnn=<node-lnn>` or by using the OneFS WebUI. This also takes the instance to a 'stopped' state from the AWS perspective.

It is recommended not to force an instance to stop from the AWS Management Console using [Force stop instance](#) or using the AWS CLI with the 'force' option. Forcing instances to stop will not give the nodes an opportunity to save the journal and the nodes will be lost if they happen to restart on a new host hardware.

To prevent your instance from being accidentally stopped or terminated, you can enable stop protection or termination protection for the instance. See the AWS documentation [Enable stop protection](#) and [Enable termination protection](#) for details.

If the journal was not saved successfully when a node went down, the node will fail to boot while restoring the content from the journal. This may also happen when a node is restarted after it went down due to host hardware failures. Though it rarely happens, a system status check failure will indicate such problem with the AWS system on which the OneFS node was running. The only way to recover the lost storage space due to the OneFS node failing to boot up is by performing the node replacement.

Appendix A: required AWS permission for deployer user

The following json format content lists all required AWS permissions for this guide.

```
{
  "Statement": [
    {
      "Action": [
        "sts:GetCallerIdentity",
        "ec2:DescribeVpcs",
        "ec2:DescribeVolumes",
        "ec2:DescribeTags",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeRouteTables",
        "ec2:DescribePlacementGroups",
        "ec2:DescribeNetworkInterfaces",
        "ec2:DescribeInstances",
        "ec2:DescribeInstanceTypes",
        "ec2:DescribeInstanceAttribute",
        "ec2:DescribeAccountAttributes",
        "ec2:DeletePlacementGroup",
        "ec2:CreateTags"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Sid": ""
    },
    {
      "Action": "ec2:DetachNetworkInterface",
      "Effect": "Allow",
      "Resource": "arn:aws:ec2:
<aws_region>:<aws_account_id>:*/*",
      "Sid": ""
    },
    {
      "Action": [
        "ec2:TerminateInstances",
        "ec2:RunInstances",
        "ec2:ModifyInstanceAttribute",
        "ec2:AttachNetworkInterface"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:ec2:
<aws_region>:<aws_account_id>:instance/${ec2:InstanceId}",
      "Sid": ""
    },
    {
      "Action": [
        "ec2:RunInstances",
```

```

        "ec2:DeleteNetworkInterface",
        "ec2:CreateNetworkInterface",
        "ec2:AttachNetworkInterface"
    ],
    "Effect": "Allow",
    "Resource": "arn:aws:ec2:
<aws_region>:<aws_account_id>:network-
interface/${ec2:NetworkInterfaceId}",
    "Sid": ""
},
{
    "Action": [
        "ec2:RunInstances",
        "ec2:CreatePlacementGroup"
    ],
    "Effect": "Allow",
    "Resource": "arn:aws:ec2:
<aws_region>:<aws_account_id>:placement-
group/${ec2:PlacementGroupName}",
    "Sid": ""
},
{
    "Action": [
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:AuthorizeSecurityGroupEgress"
    ],
    "Effect": "Allow",
    "Resource": "arn:aws:ec2:
<aws_region>:<aws_account_id>:security-group-
rule/${ec2:SecurityGroupRuleId}",
    "Sid": ""
},
{
    "Action": [
        "ec2:RunInstances",
        "ec2:RevokeSecurityGroupIngress",
        "ec2:RevokeSecurityGroupEgress",
        "ec2>DeleteSecurityGroup",
        "ec2>CreateSecurityGroup",
        "ec2:CreateNetworkInterface",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:AuthorizeSecurityGroupEgress"
    ],
    "Effect": "Allow",
    "Resource": "arn:aws:ec2:
<aws_region>:<aws_account_id>:security-
group/${ec2:SecurityGroupId}",
    "Sid": ""
},
{

```

```

        "Action": [
            "ec2:RunInstances",
            "ec2:DeleteSubnet",
            "ec2:CreateSubnet",
            "ec2:CreateNetworkInterface"
        ],
        "Effect": "Allow",
        "Resource": "arn:aws:ec2:
<aws_region>:subnet/${ec2:SubnetId}",
        "Sid": ""
    },
    {
        "Action": "ec2:RunInstances",
        "Effect": "Allow",
        "Resource":
"arn:aws:ec2:<aws_region>:<aws_account_id>:volume/${ec2:VolumeId}"
    },
    {
        "Action": [
            "ec2:DescribeVpcAttribute",
            "ec2:CreateSubnet",
            "ec2:CreateSecurityGroup"
        ],
        "Effect": "Allow",
        "Resource": "arn:aws:ec2:
<aws_region>:<aws_account_id>:vpc/${ec2:VpcId}",
        "Sid": ""
    },
    {
        "Action": "ec2:RunInstances",
        "Effect": "Allow",
        "Resource": "arn:aws:ec2:
<aws_region>::image/${ec2:ImageId}",
        "Sid": ""
    }
],
"Version": "2012-10-17"
}

```

Appendix B: supported cluster configuration details

SSD cluster of gp3

The following table shows the available combination of cluster size, volume count, and single volume size for an SSD cluster of gp3.

Table 15. Supported combination of gp3 cluster

Cluster size	EBS volume count per node	Single EBS volume size (TiB)		Cluster raw capacity (TiB)		Approximate usable capacity percentage
		minimum	maximum	minimum	maximum	
4	5	1	16	20	320	50%
5	5	1	16	25	400	60%
6	5	1	16	30	480	67%
4	6	1	16	24	384	50%
5	6	1	16	30	480	60%
6	6	1	16	36	576	67%
4	10	1	16	40	640	50%
5	10	1	16	50	800	60%
6	10	1	16	60	960	67%
4	12	1	16	48	768	50%
5	12	1	16	60	960	60%
6	12	1	16	72	1152	67%
4	15	1	16	60	960	50%
5	15	1	16	75	1200	60%
6	15	1	16	90	1440	67%
4	18	1	16	72	1152	50%
5	18	1	16	90	1440	60%
6	18	1	15.170	108	1638.360	67%
4	20	1	16	80	1280	50%
5	20	1	16	100	1600	60%
6	20	1	13.653	120	1638.360	67%

HDD cluster of st1

The following table shows the available combination of cluster size, volume count, and single volume size for an HDD cluster of st1.

Table 16. Supported combination of st1 cluster

Cluster size	EBS volume count per node	Single EBS volume size (TiB)	Cluster raw capacity (TiB)	Approximate usable capacity percentage
4	5	4 or 10	80 or 200	50%
5	5	4 or 10	100 or 250	60%
6	5	4 or 10	120 or 300	67%
4	6	4 or 10	96 or 240	50%
5	6	4 or 10	120 or 300	60%
6	6	4 or 10	144 or 360	67%

Appendix C: json file template

```
onefs-runtime-policy.json {
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "APEXFileStorageSmartConnectIPFailover",
            "Effect": "Allow",
            "Action": [
                "ec2:UnassignPrivateIpAddresses",
                "ec2:AssignPrivateIpAddresses"
            ],
            "Resource":
                "arn:aws:ec2:*:<aws_account_id>:network-interface/*"
        },
        {
            "Sid": "APEXFileStorageSmartConnectValidation",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstanceTypes"
            ],
            "Resource": "*"
        }
    ]
}
```

```
onefs-runtime-assume-role.json {
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "ec2.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

```
block-device-mappings-template.json [
    {
        "DeviceName": "xvda",
        "Ebs": {
            "VolumeSize": <volume-size-in-GiB>,
            "VolumeType": "<volume-type>",
            "DeleteOnTermination": true
        }
    },
    {
        "DeviceName": "xvdb",
```



```

    "Ebs": {
      "VolumeSize": <volume-size-in-GiB>,
      "VolumeType": "<volume-type>",
      "DeleteOnTermination": true
    }
  },
  {
    "DeviceName": "xvdc",
    "Ebs": {
      "VolumeSize": <volume-size-in-GiB>,
      "VolumeType": "<volume-type>",
      "DeleteOnTermination": true
    }
  },
  {
    "DeviceName": "xvdd",
    "Ebs": {
      "VolumeSize": <volume-size-in-GiB>,
      "VolumeType": "<volume-type>",
      "DeleteOnTermination": true
    }
  },
  {
    "DeviceName": "xvde",
    "Ebs": {
      "VolumeSize": <volume-size-in-GiB>,
      "VolumeType": "<volume-type>",
      "DeleteOnTermination": true
    }
  },
  {
    "DeviceName": "xvdf",
    "Ebs": {
      "VolumeSize": <volume-size-in-GiB>,
      "VolumeType": "<volume-type>",
      "DeleteOnTermination": true
    }
  }
]

```

**first-node-user-
data-
template.json**

```

{
  "hal_dongle_serialno": "SV200-930073-0000",
  "hal_volume_type": "<volume_type>",
  "acs_config": {
    "cluster": {
      "admin_user_password": "<admin_password>",
      "cluster_name_nt4_compatibility": false,
      "encoding": "utf-8",
      "l3_cache": {
        "ssd_l3_cache_default_enabled": false
      }
    }
  }
}

```

```

    },
    "name": "<cluster_name>",
    "nodes": [
      {
        "serial_number": "SV200-930073-0000"
      }
    ],
    "password": "<root_password>",
    "credentials_hashed": false,
    "timezone": {
      "name": "<timezone_name>"
    }
  },
  "external_networking": {
    "dns_domains": [
      "<dns_domain>"
    ],
    "dns_servers": [
      "<dns_server>"
    ],
    "external_interfaces": [
      {
        "gateway": "<cluster_gateway>",
        "interface": "ext-1",
        "ip_address_ranges": [
          {
            "low": "<first_external_ip_address>",
            "high": "<last_external_ip_address>"
          }
        ],
        "mtu": 1500,
        "netmask": "<external_ip_netmask>"
      }
    ],
    "internal_as_external": false
  },
  "internal_networking": {
    "internal_interfaces": [
      {
        "interface": "int-a",
        "ip_address_ranges": [
          {
            "low": "<first_internal_ip_address>",
            "high": "<last_internal_ip_address>"
          }
        ],
        "netmask": "<internal_ip_netmask>"
      }
    ],
    "internal_mtu": 9001
  }
}

```

```

    },
    "join_mode": "auto"
  },
  "devices": [
    {
      "ext-1": "<first_node_external_ip>",
      "int-a": "<first_node_internal_ip>",
      "serial_number": "SV200-930073-0000"
    },
    {
      "ext-1": "<second_node_external_ip>",
      "int-a": "<second_node_internal_ip>",
      "serial_number": "SV200-930073-0001"
    },
    {
      "ext-1": "<third_node_external_ip>",
      "int-a": "<third_node_internal_ip>",
      "serial_number": "SV200-930073-0002"
    },
    {
      "ext-1": "<fourth_node_external_ip>",
      "int-a": "<fourth_node_internal_ip>",
      "serial_number": "SV200-930073-0003"
    }
  ]
}

```

additional-node- user-data- template.json

```

{
  "hal_dongle_serialno": "SV200-930073-<device_id>",
  "hal_volume_type": "<volume_type>",
  "devices": [
    {
      "ext-1": "<first_node_external_ip>",
      "int-a": "<first_node_internal_ip>",
      "serial_number": "SV200-930073-0000"
    },
    {
      "ext-1": "<second_node_external_ip>",
      "int-a": "<second_node_internal_ip>",
      "serial_number": "SV200-930073-0001"
    },
    {
      "ext-1": "<third_node_external_ip>",
      "int-a": "<third_node_internal_ip>",
      "serial_number": "SV200-930073-0002"
    },
    {
      "ext-1": "<fourth_node_external_ip>",
      "int-a": "<fourth_node_internal_ip>",
      "serial_number": "SV200-930073-0003"
    }
  ]
}

```

Appendix C: json file template

```
}  
]  
}
```

Appendix D: EC2 instance user data example

User data of first node example

```

user-data-node-1.json
{
  "hal_dongle_serialno": "SV200-930073-0000",
  "hal_volume_type": "gp3",
  "acs_config": {
    "cluster": {
      "admin_user_password": "Password01!",
      "cluster_name_nt4_compatibility": false,
      "encoding": "utf-8",
      "l3_cache": {
        "ssd_l3_cache_default_enabled": false
      },
      "name": "vonefs-cfv",
      "nodes": [
        {
          "serial_number": "SV200-930073-0000"
        }
      ],
      "password": "Password01!",
      "credentials_hashed": false,
      "timezone": {
        "name": "Eastern Time Zone"
      }
    },
    "external_networking": {
      "dns_domains": [
        "us-east-1.compute.internal"
      ],
      "dns_servers": [
        "169.254.169.253"
      ],
      "external_interfaces": [
        {
          "gateway": "10.0.0.1",
          "interface": "ext-1",
          "ip_address_ranges": [
            {
              "low": "10.0.0.4",
              "high": "10.0.0.9"
            }
          ],
          "mtu": 1500,
          "netmask": "255.255.255.240"
        }
      ],
      "internal_as_external": false
    }
  },

```

```

    "internal_networking": {
      "internal_interfaces": [
        {
          "interface": "int-a",
          "ip_address_ranges": [
            {
              "low": "10.0.0.20",
              "high": "10.0.0.25"
            }
          ],
          "netmask": "255.255.255.240"
        }
      ],
      "internal_mtu": 9001
    },
    "join_mode": "auto"
  },
  "devices": [
    {
      "ext-1": "10.0.0.4",
      "int-a": "10.0.0.20",
      "serial_number": "SV200-930073-0000"
    },
    {
      "ext-1": "10.0.0.5",
      "int-a": "10.0.0.21",
      "serial_number": "SV200-930073-0001"
    },
    {
      "ext-1": "10.0.0.6",
      "int-a": "10.0.0.22",
      "serial_number": "SV200-930073-0002"
    },
    {
      "ext-1": "10.0.0.7",
      "int-a": "10.0.0.23",
      "serial_number": "SV200-930073-0003"
    }
  ]
}

```

User data of additional nodes example

The difference between the additional nodes user data is the value of the key "hal_dongle_serialno", which defines the serial number of each node. Also the acs_config key is present only on the first node.

user-data-node-2.json

The following is an example of second node user data **user-data-node-2.json**.

```

{
  "hal_dongle_serialno": "SV200-930073-0001",

```

```

"hal_volume_type": "gp3",
"devices": [
  {
    "ext-1": "10.0.0.4",
    "int-a": "10.0.0.20",
    "serial_number": "SV200-930073-0000"
  },
  {
    "ext-1": "10.0.0.5",
    "int-a": "10.0.0.21",
    "serial_number": "SV200-930073-0001"
  },
  {
    "ext-1": "10.0.0.6",
    "int-a": "10.0.0.22",
    "serial_number": "SV200-930073-0002"
  },
  {
    "ext-1": "10.0.0.7",
    "int-a": "10.0.0.23",
    "serial_number": "SV200-930073-0003"
  }
]
}

```

user-data-node-3.json

The following is an example of third node user data **user-data-node-3.json**

```

{
  "hal_dongle_serialno": "SV200-930073-0002",
  "hal_volume_type": "gp3",
  "devices": [
    {
      "ext-1": "10.0.0.4",
      "int-a": "10.0.0.20",
      "serial_number": "SV200-930073-0000"
    },
    {
      "ext-1": "10.0.0.5",
      "int-a": "10.0.0.21",
      "serial_number": "SV200-930073-0001"
    },
    {
      "ext-1": "10.0.0.6",
      "int-a": "10.0.0.22",
      "serial_number": "SV200-930073-0002"
    },
    {
      "ext-1": "10.0.0.7",
      "int-a": "10.0.0.23",
      "serial_number": "SV200-930073-0003"
    }
  ]
}

```

```
    }
}
```

user-data-node-4.json

The following is an example of fourth node user data **user-data-node-4.json**

```
{
  "hal_dongle_serialno": "SV200-930073-0003",
  "hal_volume_type": "gp3",
  "devices": [
    {
      "ext-1": "10.0.0.4",
      "int-a": "10.0.0.20",
      "serial_number": "SV200-930073-0000"
    },
    {
      "ext-1": "10.0.0.5",
      "int-a": "10.0.0.21",
      "serial_number": "SV200-930073-0001"
    },
    {
      "ext-1": "10.0.0.6",
      "int-a": "10.0.0.22",
      "serial_number": "SV200-930073-0002"
    },
    {
      "ext-1": "10.0.0.7",
      "int-a": "10.0.0.23",
      "serial_number": "SV200-930073-0003"
    }
  ]
}
```

User data example for expanding a cluster

user-data-node-5.json

The following is an example of fifth node user data **user-data-node-5.json**

```
{
  "hal_dongle_serialno": "SV200-930073-0004",
  "hal_volume_type": "gp3",
  "devices": [
    {
      "ext-1": "10.0.0.4",
      "int-a": "10.0.0.20",
      "serial_number": "SV200-930073-0000"
    },
    {
      "ext-1": "10.0.0.5",
      "int-a": "10.0.0.21",
      "serial_number": "SV200-930073-0001"
    },
    {
      "ext-1": "10.0.0.6",
      "int-a": "10.0.0.22",
      "serial_number": "SV200-930073-0002"
    },
    {
      "ext-1": "10.0.0.7",
      "int-a": "10.0.0.23",
      "serial_number": "SV200-930073-0003"
    }
  ]
}
```



```
{
  "ext-1": "10.0.0.6",
  "int-a": "10.0.0.22",
  "serial_number": "SV200-930073-0002"
},
{
  "ext-1": "10.0.0.7",
  "int-a": "10.0.0.23",
  "serial_number": "SV200-930073-0003"
},
{
  "ext-1": "10.0.0.8",
  "int-a": "10.0.0.24",
  "serial_number": "SV200-930073-0004"
}
]
```