Accelerating AI and Analytics with Multi-Region NetApp SnapMirror to NetApp ONTAP in Vultr Cloud with NVIDIA GPU

Table of Contents

Introduction

This solution enables centralization of data from multiple on-prem NetApp ONTAP regions into a unified hybrid-cloud platform on Vultr. On-prem clusters at Site-A and Site-B replicate via multi-region SnapMirror into a single NetApp ONTAP instance in Vultr Cloud, giving Vultr compute instances and NVIDIA GPUs secure, read-only, point-in-time access to datasets - ideal for AI, analytics, disaster recovery (DR), and high-performance processing.

Key Architectural Highlights

- Multi-region SnapMirror fan-in design consolidates data from Site-A and Site-B into NetApp ONTAP on Vultr.
- NetApp ONTAP presents replicated data as NFS/SMB volumes, immediately usable by Vultr compute instances and NVIDIA GPUs.
- The resulting hybrid-cloud data platform supports AI workloads, analytics pipelines, DR, and enterprise applications - following NetApp's best practices.

The attached guide provides a complete, step-by-step hybrid-cloud implementation plan. It includes CLI commands, validation checks, and architectural notes to ensure proper setup and smooth execution.

Implementation Steps Covered

- Establish secure connectivity between on-prem ONTAP clusters and Vultr Cloud.
- Perform cluster and SVM peering for authorized cross-environment replication.
- Prepare destination aggregates and volumes on NetApp ONTAP in the cloud.
- Configure SnapMirror policies and relationships to enable multi-region fan-in replication.

Together, these components build a resilient, high-performance hybrid-cloud data foundation - scalable, cloud-ready, aligned for enterprise AI workloads on NVIDIA GPUs, and ready to support modern data-driven applications.

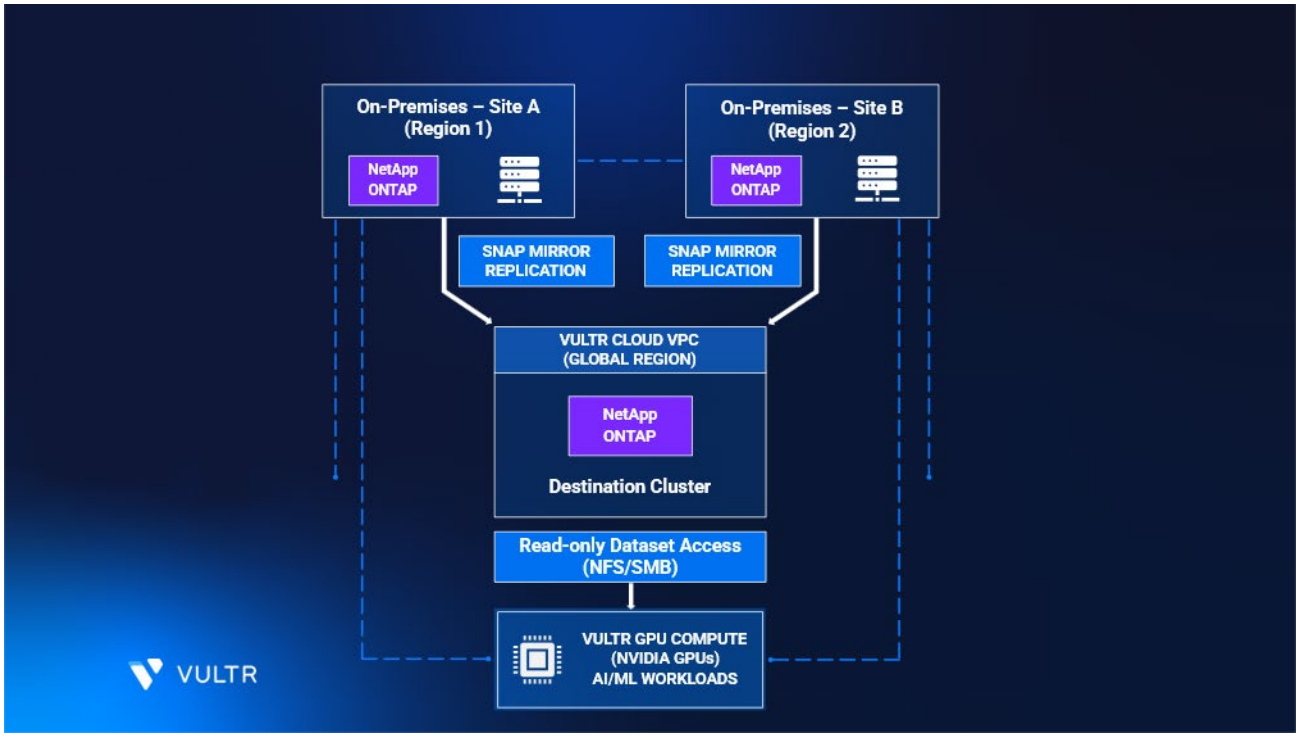


Fig 1: Multi-Region ONTAP Replication into Vultr Cloud (SnapMirror Replication)

Prerequisites

The following prerequisites ensure the environment is ready for multi-region SnapMirror replication into Vultr Cloud.

ONTAP (On-Prem)

- ONTAP 9.8+ on all source clusters
- SnapMirror license enabled
- SnapMirror requires matching major ONTAP version
- Intercluster LIFs configured on each node
- DNS, routing, and firewall rules in place
- Source SVMs and volumes ready for replication

NetApp ONTAP on VMware ESXi (in Vultr Cloud)

- Vultr -provided ESXi 8.x or ESXi 9.0 host with vCenter access to deploy given ONTAP ova file, create/assign datastores, and create port groups / networks
- NetApp ONTAP license available to be deployed on ESXi in Vultr Cloud
- Adequate CPU/RAM and attached block storage
- Management, inter-cluster, and data LIFs configured (follow the instructions below)

Networking

- IPSec VPN or secure tunnel between sites and Vultr
- Connectivity between intercluster LIFs (ping + TCP 11104/11105/10000)
- NFS/SMB ports allowed for compute access
- DNS resolution across environments

Security & Access

- Admin access to all ONTAP clusters
- Vultr console/API access
- Firewall rules permitting cluster/SVM peering and SnapMirror

Storage & Configuration

- Destination aggregates created on NetApp ONTAP
- SVMs defined for replication and data access boundaries
- SnapMirror schedules and policies predefined

Step 1 - Prepare Networking & Environment Prep

A secure, stable network foundation is required before configuring NetApp ONTAP or SnapMirror. That includes creating the VPC, deploying a VPN tunnel, setting up DNS and routing, and deploying NetApp ONTAP in Vultr Cloud. SnapMirror depends on encrypted TCP connectivity between intercluster LIFs - provided by the VPN - so correct resource provisioning and network configuration are critical to ensure all downstream steps work smoothly without impacting production.

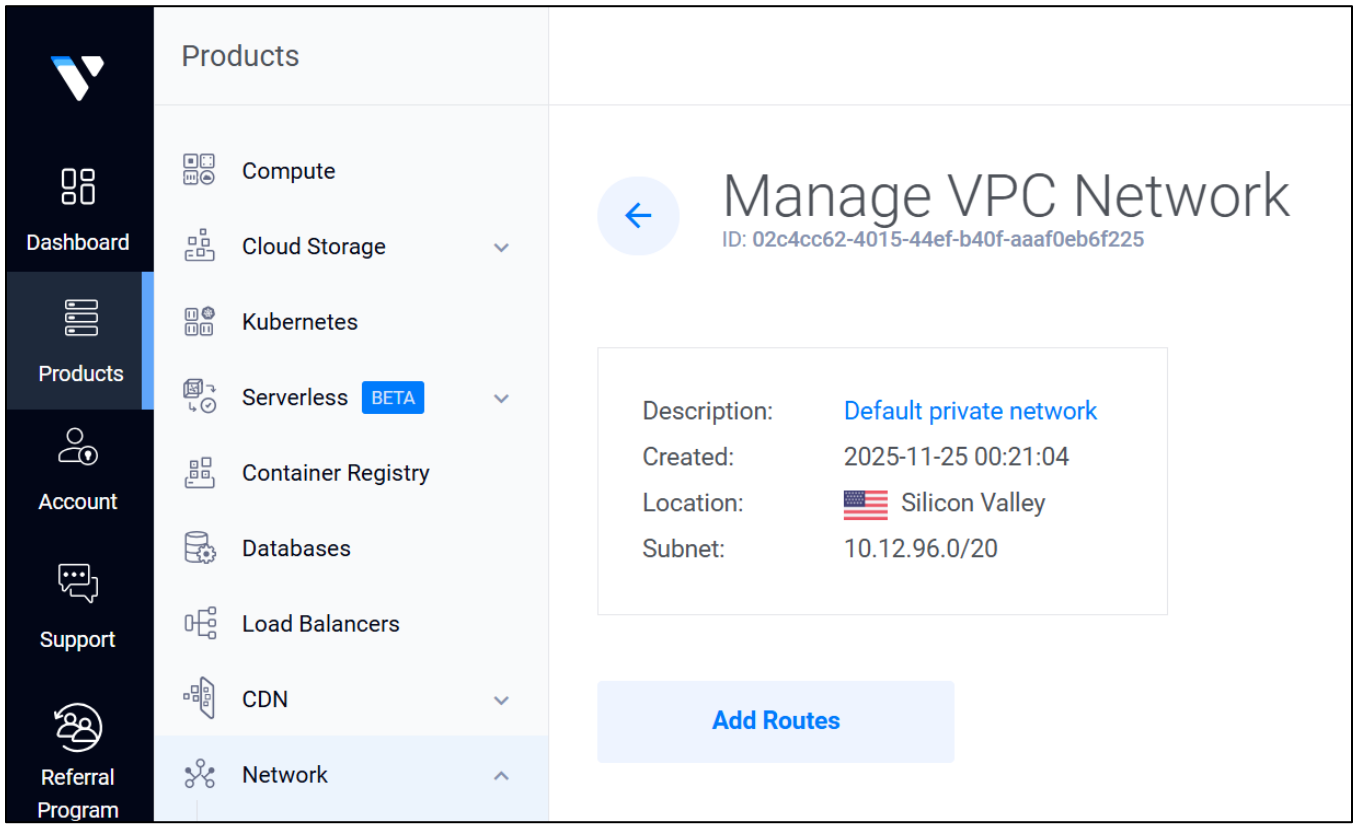
This section covers

- Create and validate Vultr Cloud account
- Create Vultr VPC and required compute instances
- Set up IPSec VPN between on-prem ONTAP and Vultr Cloud
- Validate routing using ICMP, NFS ports, and SnapMirror ports (TCP 11104/11105/10000)
- Deploy NetApp ONTAP image in Vultr
- Create ONTAP SVM (vserver) and configure DNS, routing, and firewall rules

Create Vultr VPC

A Vultr VPC is a logically isolated virtual network that provides private IP addressing and controlled routing for cloud resources. It is needed to securely host NetApp ONTAP and compute nodes in an isolated environment, ensuring predictable connectivity and a dedicated path for SnapMirror traffic.

Provision a VPC to provide network isolation and private IP space for NetApp ONTAP and compute nodes.



Expected Output:

- The VPC should display the correct subnet `10.xx.xx.0/20` and region.
- This confirms the VPC was created and provides the private network required for NetApp ONTAP and SnapMirror traffic.

Expected Output:

- The StrongSwan VM should have a private VPC IP address (e.g., **`10.xx.xx.xx`**).
- This confirms the VM is attached to the correct Vultr VPC and can communicate with NetApp ONTAP.

Deploy ONTAP on VMware ESXi

Deploy the ONTAP VM that will act as the SnapMirror destination. ONTAP instance provides full ONTAP feature compatibility in Vultr Cloud, enabling direct SnapMirror replication and exposing NFS/SMB volumes to compute nodes.

- Log in to your **VMware ESXi host**.
- Go to **Networking → Virtual Switches**.
- Create a new virtual switch named **vSwitch1** and click **Add**. This switch will be used for the VPC connection.
- Select **Port Groups → Add Port Group**.
- Enter a **Port Group Name**, choose **vSwitch1** as the virtual switch, and click **Add**.
- In the left pane, select **Virtual Machines** and click **Create / Register VM**.
- Choose **Deploy a virtual machine** from an OVA file, then click **Next**.
- Provide a **VM Name**, select the **OVA file**, and click **Next**.
- On the **Storage Options** page, click **Next** to continue.
- Under **Deployment Options**, select your VM Network, then click **Next**.
- On **Additional Settings**, enter the admin password and network configuration details, then click **Next → Finish**.
- The deployment may take 4–5 minutes, depending on your network speed.
- Once the login screen appears in the VM console, open a browser and navigate to the configured IP address to access the **ONTAP Deploy Dashboard**.

Deploy ONTAP System Manager Cluster

- Log in to the **ONTAP Deploy Dashboard**.
- **Upload your license file.**
- Under **Add Host to Inventory**, choose **Hypervisor Type → ESX**, enter your **VMware ESXi credentials**, and click **Add**.
- Under **Create a Cluster**, provide a **Cluster Name**, select a **Cluster Size**, and choose the appropriate **Network Configuration**, then click **Done**.
- In **Hypervisor and Network**, configure:
    - Node settings

  - VMware hosts
  - Required networks
- In **Storage**, select **Software RAID**. Under **Storage Pool**, ensure **all disks are the same size** for optimal stability. *Using mismatched disks can cause configuration failures, VM restarts, or cluster instability.*
- Click **Done** to proceed.
- Set the **admin password** for **ONTAP System Manager** and click **Create Cluster**.
- The cluster deployment will take about **4–5 minutes**.
- Return to the **VMware ESXi panel** and wait for the cluster VM to finish deploying.
- When the login page appears on the cluster VM console, go back to the **ONTAP Deploy Dashboard**. Under **Clusters**, select your cluster and click **Launch System Manager**.
- Log in with your credentials to access and manage storage through the **System Manager interface**.

Create LIF on NetApp ONTAP

An **Intercluster LIF on NetApp ONTAP** is a dedicated logical network interface assigned a private IP inside the VPN/VPC subnet and bound to a specific node and port. It serves as the endpoint for all cluster-to-cluster communication, including SnapMirror replication and cluster peering. ONTAP does not create these automatically, so defining at least one intercluster LIF per node is essential - without it, the cluster cannot establish peering or replicate data, even if the VPN tunnel is fully functional.

Check the available ports:

```
net port show
```

```
ONTAPSelectCluster::> net port show
  (network port show)

Node: ONTAPSelectCluster-01
                                                       Speed(Mbps) Health
Port      IPspace      Broadcast Domain Link MTU  Admin/Oper  Status
--------- ------------ ---------------- ---- ---- ----------- --------
e0a       Default      Default          up   1500  auto/auto  healthy
e0b       Default      Default          up   1500  auto/auto  healthy
e0c       Default      Default          up   1500  auto/auto  healthy
3 entries were displayed.

ONTAPSelectCluster::>
```

Use e0a which is the **default, stable, and recommended data-capable port** available on ONTAP for intercluster/LIF traffic in most deployments.

```
net int create -vserver ONTAPSelectCluster -lif ic1 -address 10.12.96.51 \

  -netmask 255.255.240.0 -home-node ONTAPSelectCluster-01 -home-port e0a
```

```
ONTAPSelectCluster::> net int create -vserver ONTAPSelectCluster -lif ic1 -address 10.12.96.51 -netmask 255.255.240.0 -home-node ONTA
PSelectCluster-01 -home-port e0a
```

CLI Validation

Check LIF state:

```
network interface show
```

```
ONTAPSelectCluster::> network interface show
            Logical     Status      Network             Current       Current Is
Vserver     Interface   Admin/Oper  Address/Mask        Node          Port    Home
----------- ----------- ----------- ------------------- ------------- ------- ----
ONTAPSelectCluster
            ONTAPSelectCluster-01_mgmt1
                        up/up    10.12.96.61/20     ONTAPSelectCluster-01
                                                                       e0a     true
            cluster_mgmt up/up    10.12.96.60/20     ONTAPSelectCluster-01
                                                                       e0a     true
            ic1         up/up     10.12.96.51/20     ONTAPSelectCluster-01
                                                                       e0a     true
3 entries were displayed.

ONTAPSelectCluster::>
```

```
net int show -role intercluster -fields role,address,status-oper,status-admin,lif,home-port,home-node
```

```
ONTAPSelectCluster::> net int show -role intercluster -fields role,address,status-oper,status-admin,lif,home-port,home-node
  (network interface show)
vserver           lif role         address     home-node            home-port status-oper status-admin
----------------- --- ------------ ----------- -------------------- --------- ----------- ------------
ONTAPSelectCluster ic1 intercluster 10.12.96.51 ONTAPSelectCluster-01 e0a       up          up
```

Expected Output:

- lif: ic1
- Address: 10.12.96.51
- Home Node: ONTAPSelectCluster-01
- Current Node: ONTAPSelectCluster-01 (unless it has failed over)
- Home Port: **e0a**
- Operational Status: up
- Administrative Status: up

Enable NFS on ONTAP

Enable NFS using the following command on NetApp ONTAP

CLI

```
vserver nfs create -vserver SVM_data -v3 enabled -v4.0 enabled
```

Verify NFS server creating with:

```
nfs show
```

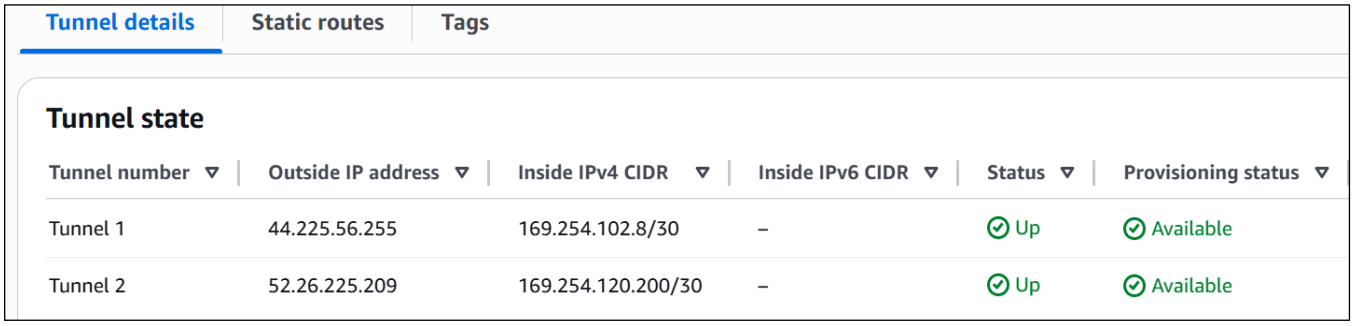
```
ONTAPSelectCluster::> vserver nfs create -vserver SVM_data -v3 enabled -v4.0 enabled

ONTAPSelectCluster::> nfs show

Vserver: SVM_data

        General Access:  true
                    v3:  enabled
                  v4.0:  enabled
                   4.1:  enabled
                   UDP:  enabled
                   TCP:  enabled
                  RDMA:  enabled
  Default Windows User:  -
 Default Windows Group:  -
```

Set Up VPN/IPSec

SnapMirror replication requires secure, encrypted TCP connectivity between intercluster LIFs. A VPN/IPSec tunnel provides that secure transport path across public networks. Set up a secure IPSec tunnel between on-prem ONTAP and Vultr Cloud.

The VPN tunnel provides the **secure transport path** between the on-prem and Vultr environments. Without it, your ONTAP VM inside Vultr has **no route to the on-prem gateway or DNS** for name resolution or reachability tests.

StrongSwan was deployed on the Vultr compute node to serve as the VPN end-point for this build guide.




```
ipsec statusall
```

Expected Output:

- Both IPSec tunnels should show **Status: UP**.

This confirms On-prem has successfully established Phase-1 and Phase-2 security associations with the StrongSwan gateway in Vultr.

Configure DNS & Routing

After the VPN is active, you can safely set default and inter-cluster routes so ONTAP can:

- Resolve hostnames across sites
- Reach on-prem inter-cluster LIFs over the new encrypted tunnel

Steps:

- Add DNS servers
- Add a default route pointing through the VPN gateway
- Add inter-cluster routes for the on-prem CIDRs

Set DNS servers and create default/inter-cluster routes.

Add local host entries for peer clusters

- 10.12.96.51 → **Vultr ONTAP intercluster LIF IP**
- 172.31.157.201 → On-Prem **ONTAP inter-cluster LIF IP**

These are private IPs inside the VPN tunnel - internal intercluster LIF addresses used exclusively for SnapMirror data replication, which relies on these dedicated ONTAP logical interfaces for all replication traffic.

Note: They are *not* the public IPs of StrongSwan or bastion hosts.

They are the *internal* cluster communication IPs used for SnapMirror data replication.

Each ONTAP (On-Prem and Vultr) must be able to:

- Resolve the **peer cluster's intercluster LIF hostname**
- Reach that IP across the VPN (on TCP ports 11104 and 11105)

On Vultr ONTAP

CLI:

```
vserver services name-service dns hosts create -vserver ONTAPSelectCluster -address
172.31.157.201 -hostname inter_1
```

Verify:

```
vserver services name-service dns show
```

```
ONTAPSelectCluster::> vserver services name-service dns hosts show
Vserver    Address         Hostname        Aliases
---------- --------------- --------------- ---------------------
ONTAPSelectCluster
           172.31.157.201 inter_1         -
```

CLI on On-Prem ONTAP

```
vserver services name-service dns hosts create -vserver sx -address 10.12.96.51 -hostname
ic1
```

Verify:

```
vserver services name-service dns hosts show
```

```
sxId0edb1927eae795ba7::> vserver services name-service dns hosts show
 Vserver    Address         Hostname        Aliases
 ---------- --------------- --------------- ---------------------
 sx        10.12.96.51    ic1
```

Create Route

This is the Vultr ONTAP VPC subnet: `10.12.96.0/20`

- ONTAP intercluster LIF: `10.12.96.51`
- StrongSwan LAN IP (VPN gateway in Vultr): `10.12.96.6`

On-Premise ONTAP VPC subnet: `172.31.0.0/16`

- On-Prem ONTAP Intercluster LIF: `172.31.157.201`
- On-Prem local subnet gateway: `172.31.144.1`

Note: The routes are shown for the IPs above. This would change as per setup.

Route to be added on Vultr ONTAP (to reach on-prem equivalent)

To reach anything in the On-Prem ONTAP VPC (`172.31.0.0/16`), send traffic to the StrongSWAN LAN IP (`10.12.96.6`), which is the VPN gateway on the Vultr side.

```
network route create -vserver ONTAPSelectCluster -destination 172.31.0.0/16 -gateway
10.12.96.6
```

CLI Validation

```
network route show
```

```
ONTAPSelectCluster::> network route show
Vserver              Destination     Gateway          Metric
-------------------- --------------- ---------------- ------
ONTAPSelectCluster
                     0.0.0.0/0       10.12.96.1       10
                     172.31.0.0/16   10.12.96.6       20
2 entries were displayed.

ONTAPSelectCluster::>
```

Expected Output:

- A default route exists
- Specific route(s) to on-prem network appear
- No overlapping or incorrect routes

Route to be added On-Prem ONTAP (to reach Vultr ONTAP)

To reach the Vultr ONTAP subnet (`10.12.96.0/20`), send traffic to On-Prem local VPC subnet gateway (`172.31.144.1`)

```
network route create -vserver sx -destination 10.12.96.0/20 -gateway 172.31.144.1
```

CLI Validation

```
network route show
```

```
sxId0edb1927eae795ba7::*> network route show
 Vserver             Destination     Gateway         Metric
 ------------------- --------------- --------------- ------
 sx
                     0.0.0.0/0       172.31.144.1    20
                     10.12.96.0/20   172.31.144.1    20

 2 entries were displayed.
```

Validate Connectivity

To validate that the VPN tunnel and routing are correctly configured for SnapMirror, both sites must be able to reach each other's **intercluster LIFs**. ONTAP provides a built-in way to test this using network ping **from the LIF itself**, ensuring that the test follows the correct broadcast domain, routing table, and VPN path.

CLI Validation

Test From On-Prem → ONTAP(Cloud)

```
network ping -lif inter_1 -vserver sxId0edb1927eae795ba7 -destination <Cloud_IC_LIF_IP>

network ping -lif inter_2 -vserver sxId0edb1927eae795ba7 -destination <Cloud_IC_LIF_IP>
```

Use IP address and not ic1. ic1 is the *name* of a LIF object inside ONTAP, not a DNS hostname, so ONTAP cannot resolve it unless we explicitly create a DNS/hosts entry with that name.

```
sxId0edb1927eae795ba7::> network ping -lif inter_1 -vserver sxId0edb1927eae795ba7 -
destination 10.12.96.51
 10.12.96.51 is alive

sxId0edb1927eae795ba7::> network ping -lif inter_2 -vserver sxId0edb1927eae795ba7 -
destination 10.12.96.51
 10.12.96.51 is alive
```

Test From ONTAP(Cloud) → On-Prem

```
network ping -lif ic1 -vserver ONTAPSelectCluster -destination <ONprem_IC_LIF_1>

ONTAPSelectCluster::> network ping -lif ic1 -vserver ONTAPSelectCluster -destination
172.31.157.201
 172.31.157.201 is alive

network ping -lif ic1 -vserver ONTAPSelectCluster -destination <ONprem_IC_LIF_2>

ONTAPSelectCluster::> network ping -lif ic1 -vserver ONTAPSelectCluster -destination
172.31.144.104
 172.31.144.104 is alive
```

Expected Output:

- **<OnPrem_IP>** is alive
- Symmetric reachability
- No routing errors (e.g., "Network unreachable")

This validates that the ONTAP Vultr Cloud intercluster LIF can return traffic back to the on-prem intercluster LIFs.

SnapMirror Firewall Ports

The following TCP ports **must be enabled bidirectionally** on your firewall between the Inter-cluster LIFs of your ONTAP clusters (On-Prem and Vultr ONTAP) before cluster peering can be established.

- **TCP Port 11104 (SnapMirror Control)**
  - **Purpose:** Mandatory for establishing and maintaining cluster peering.
  - **Function:** Used for the secure control channel, **authentication**, and **metadata exchange** (the peering handshake).
- **TCP Port 10000 (Data Transfer)**
  - **Purpose:** Mandatory for all SnapMirror data transfer operations.
  - **Function:** Used for the **actual block-level data transfer** during replication.

CLI run on On-Prem:

```
nc -zv 10.12.96.51 11104

nc -zv 10.12.96.51 11105
```

```
ubuntu@ip-172-31-12-99:~$ nc -zv 10.12.96.51 11104
Connection to 10.12.96.51 11104 port [tcp/*] succeeded!
ubuntu@ip-172-31-12-99:~$ nc -zv 10.12.96.51 11105
Connection to 10.12.96.51 11105 port [tcp/*] succeeded!
ubuntu@ip-172-31-12-99:~$ nc -zv 10.12.96.51 10000
Connection to 10.12.96.51 10000 port [tcp/webmin] succeeded!
```

CLI run on Vultr :

```
nc -zv 172.31.157.201 11104

nc -zv 172.31.157.201 11105
```

```
root@strongswan-01:~# nc -zv 172.31.157.201 11104
Connection to 172.31.157.201 11104 port [tcp/*] succeeded!
root@strongswan-01:~# nc -zv 172.31.157.201 11105
Connection to 172.31.157.201 11105 port [tcp/*] succeeded!
root@strongswan-01:~# nc -zv 172.31.157.201 10000
Connection to 172.31.157.201 10000 port [tcp/webmin] succeeded!
```

Expected Output:

- Connection to `... port [tcp/*]` succeeded!
- Zero connection timeouts or refusals.
- Confirms bidirectional traffic flow on critical SnapMirror replication ports.

Essential NFS Access Ports

These ports are necessary for your **Vultr Compute/GPU Nodes** to successfully **mount and access the read-only datasets** from the **NetApp ONTAP** system. They must be opened on the firewall governing traffic between the compute nodes and the ONTAP Data LIFs.

- **TCP/UDP Port 111**
  - **Purpose:** Mandatory for initial client-server communication and service discovery.
  - **Function:** Used by the NFS client (Vultr Compute Nodes) to find the dynamic port numbers of other necessary NFS services, such as Mountd.
- **TCP/UDP Port 2049**
  - **Purpose:** Mandatory for all NFS data access and transfer operations.

- **Function:** The primary port used for the core NFS protocol, enabling clients to read and write file data across the network. (Note: For **NFSv4.x**, this port often handles all functions.)

CLI run on ONTAP:

To check if a **Vultr Compute Node** can reach the **NetApp ONTAP Data LIF** (`10.12.96.52`) on the main NFS port (2049):

Get **NetApp ONTAP Data LIF**

```
network interface show -role data
```

```
ONTAPSelectCluster::> network interface show -role data
            Logical    Status     Network            Current       Current Is
Vserver     Interface  Admin/Oper Address/Mask       Node          Port    Home
----------- ---------- ---------- ------------------ ------------- ------- ----
SVM_data
            SVM_data_Cifs up/up   10.12.96.52/20     ONTAPSelectCluster-01 e0c true

ONTAPSelectCluster::>
```

Use the above ONTAP Data LIF to run the netcat command.

```
nc -zv 10.12.96.52 2049
```

```
nc -zv 10.12.96.52 111
```

```
root@strongswan-01:~# nc -zv 10.12.96.52 2049
Connection to 10.12.96.52 2049 port [tcp/nfs] succeeded!
root@strongswan-01:~# nc -zv 10.12.96.52 111
Connection to 10.12.96.52 111 port [tcp/sunrpc] succeeded!
```

Step 2 – Configure ONTAP Cluster Peering

Cluster and SVM peering establish trust and authorization for cross-environment replication. This ensures both on-prem ONTAP and Vultr ONTAP can communicate securely over inter-cluster LIFs, and that destination volumes are ready to receive replicated data. Without proper peering, SnapMirror relationships cannot be created or initialized.

This section covers

- Create intercluster LIFs on both on-prem ONTAP and Vultr NetApp ONTAP
- Configure cluster peering (bidirectional trust)
- Configure SVM peering (data SVM to data SVM authorization)
- Validate encrypted intercluster connectivity
- Prepare destination aggregates and create initial volumes

Cluster Peering

Cluster peering is essential for enabling SnapMirror replication. It establishes a secure trust relationship between source and destination ONTAP clusters, allowing them to authenticate and exchange replication metadata. This process relies entirely on **Inter-cluster LIFs**, which are dedicated network interfaces used solely for managing the ONTAP-to-ONTAP traffic required for peering and data transfer.

Note: Always run the '`cluster peer create`' command first on the destination side - the side that will RECEIVE the peer request.

On ONTAP on Vultr Cloud

```
network interface show -role intercluster
```

```
ONTAPSelectCluster::> network interface show -role intercluster
            Logical    Status     Network            Current       Current Is
Vserver     Interface  Admin/Oper Address/Mask       Node          Port    Home
----------- ---------- ---------- ------------------ ------------- ------- ----
ONTAPSelectCluster
            ic1          up/up    10.12.96.51/20     ONTAPSelectCluster-01 e0a true
```

On On-Prem ONTAP

```
sxId0edb1927eae795ba7::> network interface show -role intercluster
                    Logical      Status              Network                    Current
Current       Is
 Vserver     Interface  Admin/Oper Address/Mask         Node
Port              Home
 -----------   ----------     ---------------  ------------------          ------------
-                     -------            ----
 sxId0edb1927eae795ba7
                    inter_1      up/up              172.31.157.201/20
sxId0edb1927eae795ba7-01   e0e     true
                    inter_2      up/up              172.31.144.104/20
sxId0edb1927eae795ba7-02   e0e     true
 2 entries were displayed.
```

Expected Output

- status-admin = up, status-oper = up
- Correct IPs assigned
- Home-node and home-port match the configuration

CLI for cluster peering to be run on Vultr Cloud NetApp ONTAP

```
cluster peer create -peer-addrs <peer-intercluster-LIF-IP of ON-PREM> -generate-passphrase
true
```

```
ONTAPSelectCluster::*> cluster peer show
This table is currently empty.

ONTAPSelectCluster::*> cluster peer create -peer-addrs 172.31.144.104 -generate-
passphrase true
```

```
Notice:
        Passphrase: tl++CrpFI1Y/Axxp+3YzlU+R
        Expiration Time: 12/3/2025 06:37:24 +00:00
        Initial Allowed Vserver Peers: -
        Intercluster LIF IP: 10.12.96.51
        Peer Cluster Name: sxId0edb1927eae795ba7

       Warning: make a note of the passphrase - it cannot be displayed again.
```

Note: Copy the Passphrase which will be needed in the next command

CLI for cluster peering to be run on On-Prem ONTAP

Here, don't pass '`-generate-passphrase true`', as we need to use the generated passphrase from NetApp ONTAP.

- **Passphrase is generated only once** - on the destination cluster that initiates the peering (NetApp ONTAP).
- The **on-prem source** must **reuse** that same passphrase when replying to complete the peering.

```
cluster peer create -peer-addrs <peer-intercluster-LIF-IP of ON-PREM>
```

```
sxId0edb1927eae795ba7::> cluster peer create -address-family ipv4 -peer-addrs 10.12.96.51

Notice: Use a generated passphrase or choose a passphrase of 8 or more characters. To
ensure the authenticity of the peering relationship, use a phrase or sequence of
characters that would be hard to guess.

Enter the passphrase:
Confirm the passphrase:
```

Validation

```
cluster peer show
```

```
ONTAPSelectCluster::*> cluster peer show
Peer Cluster Name         Cluster Serial Number Availability   Authentication
------------------------- --------------------- -------------- --------------
sxId0edb1927eae795ba7    1-80-000011           Available      ok
```

```
sxId0edb1927eae795ba7::> cluster peer show
Peer Cluster Name         Cluster Serial Number Availability   Authentication
------------------------- --------------------- -------------- --------------
ONTAPSelectCluster        1-80-000011            Available      ok
```

Expected Output

- Availability = **Available**
- Authentication = **ok**
- Remote cluster name displayed
- No timeout or "unreachable" errors

SVM Peering

SVM (vserver) peering links the data SVMs on each cluster so SnapMirror can replicate volumes between them. Since SnapMirror relationships are defined at the SVM level, this is mandatory. Even if clusters are peered, SnapMirror cannot operate unless the source and destination SVMs are authorized to replicate data.

Since your **ONTAP cluster (in Vultr Cloud)** is the replication **destination**, it is the ideal place to initiate the vserver peer create command. The **On-Prem ONTAP** system will then accept the request.

CLI run on NetApp ONTAP (Vultr Cloud)

Run the following command on the ONTAP cluster (the destination). This command creates the pending peering relationship.

```
vserver peer create -vserver <dst_svm_ontap_select> -peer-vserver <src_svm_onprem> –
applications flexcache,snapmirror –peer-cluster <onprem-cluster-name>
```

```
ONTAPSelectCluster::*> vserver peer create -vserver SVM_data -peer-vserver sx -
applications flexcache,snapmirror -peer-cluster sxId0edb1927eae795ba7

Info: [Job 52] 'vserver peer create' job queued
```

CLI run on On-prem ONTAP

Immediately after the create command, run the following command on the corresponding **On-Prem cluster** (the source) to accept the peering request and finalize the relationship.

```
vserver peer accept -vserver <src_svm_onprem> -peer-vserver <dst_svm_ontap_select>
```

```
sxId0edb1927eae795ba7::> vserver peer accept -vserver sx -peer-vserver SVM_data

Info: [Job 82] 'vserver peer accept' job queued
```

Validation from both NetApp ONTAP and On-prem ONTAP

```
vserver peer show
```

```
ONTAPSelectCluster::*> vserver peer show
            Peer        Peer                           Peering        Remote
Vserver     Vserver     State        Peer Cluster      Applications   Vserver
----------- ----------- ------------ ----------------- -------------- ---------
SVM_data    sx          peered       sxId0edb1927eae795ba7
                                                       flexcache, snapmirror
```

```
sxId0edb1927eae795ba7::> vserver peer show
            Peer        Peer                           Peering        Remote
Vserver     Vserver     State        Peer Cluster      Applications   Vserver
----------- ----------- ------------ ----------------- -------------- ---------
sx          SVM_data    peered       ONTAPSelectCluster
                                                       flexcache, snapmirror
                                                                      SVM_data
```

Expected Output

- Peer state = **peered**
- Applications = **snapmirror**
- No "initial" or "pending" states
- No peer conflicts

Step 3 - Setup Destination Volumes & SnapMirror Replication

SnapMirror configuration is at the core of multi-region replication, enabling datasets from each on-prem site to synchronize into the centralized NetApp ONTAP instance. This step defines the destination volumes, security/export rules, policies, and relationships that govern how data flows into the cloud. Once complete, you establish a functioning fan-in replication model ready for AI and analytics consumption.

This section covers

- Network Test-Path (SnapMirror Connectivity Validation)
- Create destination FlexVols on ONTAP
- Configure volume export/security settings (NFS/SMB)
- Create SnapMirror policies (e.g., MirrorAllSnapshots)
- Create SnapMirror relationships for each site
- Initialize baseline transfer
- Verify ongoing replication status
- Test read-only mounts on Vultr compute nodes

Network Test-Path (SnapMirror Connectivity Validation)

The network test-path command is a specialized **NetApp ONTAP utility** used to **proactively validate the entire network path** necessary for SnapMirror and cluster peering operations. It is the most robust way to verify your network is ready.

This command should be run **after** Cluster Peering is set up but **before** you attempt to set up any SnapMirror relationships.

This tests the following:

- **Connectivity & Routing:** Confirms that a dedicated SnapMirror connection can be established between the source and destination cluster nodes over the correct **Intercluster LIFs**.
- **Firewall Status:** Verifies that the firewall allows traffic for both the **SnapMirror control channel** (**TCP 11104**) and the **data transfer channel** (**TCP 10000**).
- **LIF Configuration:** Ensures the Intercluster LIFs on both the source and destination are properly configured, up, and listening for SnapMirror traffic.

Run this from both ONTAP and On-prem ONTAP.

```
network test-path -source-node <Source_Node_Name> -destination-cluster
<Destination_Cluster_Name> -destination-node <Destination_Node_Name>
```

```
sxId0edb1927eae795ba7::*> network test-path -source-node sxId0edb1927eae795ba7-01 -
destination-cluster ONTAPSelectCluster -destination-node ONTAPSelectCluster-01

Warning: This operation will generate large amount of cluster traffic and can cause
temporary cluster traffic slowness.

Do you want to continue? {y|n}: y

Initiating path test. It can take up to 120 seconds for results to be displayed.
     Test Duration: 14.25 secs
   Send Throughput: 27.97 MB/sec
Receive Throughput: 27.97 MB/sec
           MB Sent: 398.62
       MB Received: 398.62
       Avg Latency: 5157.99 ms
```

```
sxId0edb1927eae795ba7::*> network test-path -source-node sxId0edb1927eae795ba7-02 -
destination-cluster ONTAPSelectCluster -destination-node ONTAPSelectCluster-01
Warning: This operation will generate large amount of cluster traffic and can cause
temporary cluster traffic slowness.
Do you want to continue? {y|n}: y

     Test Duration: 14.26 secs
   Send Throughput: 29.64 MB/sec
Receive Throughput: 29.64 MB/sec
           MB Sent: 422.56
       MB Received: 422.56
       Avg Latency: 4797.68 ms
ONTAPSelectCluster::*> network test-path -source-node ONTAPSelectCluster-01 -destination-
cluster  sxId0edb1927eae795ba7 -destination-node sxId0edb1927eae795ba7-01

Warning: This operation will generate large amount of cluster traffic and can cause
temporary cluster traffic slowness.
Do you want to continue? {y|n}: y

Initiating path test. It can take up to 120 seconds for results to be displayed.
     Test Duration: 14.23 secs
   Send Throughput: 20.71 MB/sec
Receive Throughput: 20.71 MB/sec
           MB Sent: 294.69
       MB Received: 294.69
       Avg Latency: 5387.71 ms
```

Prepare Destination Aggregates and Create Initial Volumes

Aggregates are storage pools, and FlexVols are the ONTAP volumes that will receive replicated SnapMirror data. SnapMirror requires an online destination volume before relationships can be created or initialized.

CLI

Create aggregate:

```
storage aggregate create -aggregate <aggr_vultr> -disklist <connected-disk> -node <node>
```

```
ONTAPSelectCluster::> aggr create -aggregate aggr_data -disklist NET-1.1 -ha-policy sfo -
node ONTAPSelectCluster-01

Info: The layout for aggregate "aggr_data" on node "ONTAPSelectCluster-01" would be:

      First Plex

        RAID Group rg0, 1 disks (advanced_zoned checksum, raid0)
                                                          Usable Physical
          Position   Disk                      Type           Size     Size
          ---------- ------------------------- ---------- -------- --------
          data       NET-1.1                   SSD          3.44TB   3.50TB

      Aggregate capacity available for volume use would be 3.10TB.
      3.50TB would be used from capacity license.

Do you want to continue? {y|n}: y
[Job 40] Job succeeded: DONE

ONTAPSelectCluster::>
```

Validation

```
aggregate show
```

```
ONTAPSelectCluster::> aggr show

Aggregate     Size Available Used% State   #Vols  Nodes            RAID Status
--------- -------- --------- ----- ------- ------ ---------------- ------------
aggr0_ONTAPSelectCluster_01
           60.22GB    2.92GB   95% online       1 ONTAPSelectClust raid0,
                                                  er-01            normal
aggr_data   3.10TB    3.10TB    0% online       0 ONTAPSelectClust raid0,
                                                  er-01            normal
2 entries were displayed.
```

Create Destination FlexVol

A **FlexVol** is NetApp's core **logical unit of storage**, a virtual container that holds your data and is easily managed and resized independent of the physical disks. It is essential for SnapMirror because every replication relationship requires a pre-created FlexVol volume to serve as the **source or destination object**, defining the boundaries of the data being mirrored. The FlexVol will use space from the **Aggregate** you created previously, which provides the actual physical disk capacity for the volume.

CLI

```
volume create -vserver <dst_svm_ontap_select> -volume <volname> -aggregate aggr_data -size
<size> -type DP
```

```
ONTAPSelectCluster::> vol create -volume Test_Vol -vserver SVM_data -size 25GB -aggregate
aggr_data -type DP

[Job 55] Job succeeded: Successful
```

Validation

```
volume show
```

```
ONTAPSelectCluster::*> vol show
Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- -----
ONTAPSelectCluster-01 vol0 aggr0_ONTAPSelectCluster_01 online RW 56.98GB 45.33GB  16%
SVM_data  SVM_data_root aggr_data   online     RW          1GB    970.9MB    0%
SVM_data  Test_Vol     aggr_data    online     DP         25GB    25.00GB    0%
SVM_data  Test_Vol_1026 aggr_data   offline    DEL        25GB          -     -
4 entries were displayed.

ONTAPSelectCluster::*>
```

Expected Output

- Volume appears with **state = online**
- Volume assigned to **aggr_data**

Create SnapMirror Policy

A SnapMirror policy defines how replication behaves - what snapshot copies are transferred and how frequently. Each site requires a policy to govern replication consistency, retention, and update behaviour.

This policy **must be created before** defining the SnapMirror relationship. The snapmirror create command requires referencing an existing policy using the -policy *<policy_name>* parameter, and ONTAP will not allow a relationship to be created with a policy that does not yet exist.

CLI on Vultr Cloud ONTAP

Create policy

```
snapmirror policy create -vserver dst_svm -policy MirrorAllSnapshots \

-type async-mirror
```

```
ONTAPSelectCluster::> snapmirror policy create -vserver SVM_data -policy MirrorAllSnapshots -type async-mirror
```

Show snapmirror policies:

```
snapmirror show –fields policy
```

```
ONTAPSelectCluster::*> snapmirror show -fields policy
 source-path          destination-path         policy
 ----------------         ---------------------        -----------------
 sx:Test_cifs_01   SVM_data:Test_Vol     MirrorAllSnapshots
```

```
snapmirror policy show –policy MirrorAllSnapshots
```

```
ONTAPSelectCluster::> snapmirror policy show -policy MirrorAllSnapshots
Vserver Policy            Policy Number        Transfer
Name    Name              Type   Of Rules Tries Priority Comment
------- ----------------- ------ -------- ----- -------- ----------
ONTAPSelectCluster MirrorAllSnapshots async-mirror 2 8 normal  Asynchronous SnapMirror policy for mirroring all Snapshot copies and t
he latest active file system.
  SnapMirror Label: sm_created                       Keep:       1
                    all_source_snapshots                          1
                                                     Total Keep:       2
```

Expected Output

- Policy listed with type = **async-mirror**
- Rules correspond to snapshot transfer requirements

Create SnapMirror Relationship

A SnapMirror relationship links the source volume on the on-prem cluster to the destination volume in Vultr. This establishes the replication path and prepares ONTAP for the initial baseline copy.

CLI on Vultr Cloud ONTAP

```
snapmirror create -source-path sx:Test_cifs_01 \

-destination-path SVM_data:Test_Vol  -policy MirrorAllSnapshots
```

Validation

```
snapmirror show
```

```
ONTAPSelectCluster::*> snapmirror show
                                                                       Progress
Source            Destination Mirror  Relationship   Total            Last
Path        Type  Path        State   Status         Progress  Healthy Updated
----------- ---- ------------ ------- -------------- --------- ------- --------
sx:Test_cifs_01 XDP SVM_data:Test_Vol Uninitialized Idle -    true    -
```

Expected Output

- Relationship state = Uninitialized (before baseline)
- Policy = MirrorAllSnapshots
- No connectivity or permission errors

Initialize Baseline Transfer

The baseline transfer is the first full copy of the source volume to the destination. All subsequent replication updates depend on the initial baseline being completed successfully.

CLI on Vultr Cloud ONTAP

```
snapmirror initialize -destination-path dst_svm:vol1
```

```
ONTAPSelectCluster::> snapmirror initialize -destination-path SVM_data:Test_Vol
Operation is queued: SnapMirror initialize of destination "SVM_data:Test_Vol".
```

CLI Validation

Health checks confirm that SnapMirror continues to operate normally after initialization. Ensures that replication from both regions is consistent and stable.

```
snapmirror show
```

```
ONTAPSelectCluster::> snapmirror show
                                                                       Progress
Source            Destination Mirror  Relationship   Total            Last
Path        Type  Path        State   Status         Progress  Healthy Updated
----------- ---- ------------ ------- -------------- --------- ------- --------
sx:Test_cifs_01
            XDP  SVM_data:Test_Vol
                             Snapmirrored
                                     Finalizing     10.59KB   true    12/04 11:36:41
```

```
ONTAPSelectCluster::> snapmirror show
                                                                       Progress
Source            Destination Mirror  Relationship   Total            Last
Path        Type  Path        State   Status         Progress  Healthy Updated
----------- ---- ------------ ------- -------------- --------- ------- --------
sx:Test_cifs_01
            XDP  SVM_data:Test_Vol
                             Snapmirrored
                                     Transferring   21.19KB   true    12/04 11:36:48
```

```
ONTAPSelectCluster::> snapmirror show
                                                                       Progress
Source            Destination Mirror  Relationship   Total             Last
Path        Type  Path        State   Status         Progress  Healthy Updated
----------- ---- ------------ ------- -------------- --------- ------- --------
sx:Test_cifs_01 XDP SVM_data:Test_Vol Snapmirrored Idle -     true    -
```

Expected Output

- Status = Finalizing, Transferring (during) → Idle (after)
- Progress increases steadily
- Healthy = "true"

Validate Snapmirror

After initializing SnapMirror, use these commands to confirm that the destination volume is receiving Snapshots and that replication updates are occurring successfully.

CLI to be run on Vultr ONTAP

snapshot show lists all Snapshot copies received on the destination DP volume, confirming successful replication.

```
snapshot show -volume <dest>
```

```
ONTAPSelectCluster::> snapshot show -volume Test_Vol
                                                                 ---Blocks---
Vserver  Volume   Snapshot                                  Size Total% Used%
-------- -------- ------------------------------------ -------- ------ -----
SVM_data Test_Vol
                  daily.2025-12-03_0010                    296KB     0%    6%
                  daily.2025-12-04_0010                    228KB     0%    5%
                  hourly.2025-12-04_0605                   228KB     0%    5%
                  hourly.2025-12-04_0705                   224KB     0%    5%
                  backup-0dd2b80b3a261228d                 224KB     0%    5%
                  hourly.2025-12-04_0805                   224KB     0%    5%
                  hourly.2025-12-04_0905                   224KB     0%    5%
                  hourly.2025-12-04_1005                   224KB     0%    5%
                  hourly.2025-12-04_1105                   252KB     0%    5%
                  snapmirror.8e57bbd9-ceae-11f0-8278-00a0b8ca4bc5_2156550278.2025-12-04_113633 164KB 0% 3%
10 entries were displayed.
```

Expected Output:

- **Scheduled snapshots** from the source (daily.*, hourly.*)
- **A "backup-..." snapshot** created by the source SVM's local policies
- **A "snapmirror.*" snapshot** created automatically by SnapMirror during initialization or an update

These names **mirror the Snapshot copies transferred from the on-prem SVM** because SnapMirror policy transfers **all Snapshot labels** (MirrorAllSnapshots).

CLI to be run on Vultr ONTAP

snapmirror show-history displays recent transfer events, letting you verify that updates are running as expected and identify any failures or delays.

```
snapmirror show-history
```

```
ONTAPSelectCluster::*> snapmirror show-history

Destination  Source                   Start         End
Path         Path        Operation    Time          Time        Result
 ----------- ----------- ---------        ----------- ----------- -------
 SVM_data:Test_Vol sx:Test_cifs_01 modify 12/5/2025 05:31:39 12/5/2025 05:31:39 success
 SVM_data:Test_Vol sx:Test_cifs_01 scheduled-update 12/4/2025 11:36:33 12/4/2025 11:36:55
success
 SVM_data:Test_Vol sx:Test_cifs_01 initialize 12/4/2025 11:36:33 12/4/2025 11:36:35
success
 3 entries were displayed.
```

CLI on On-Prem ONTAP

snapmirror list-destinations (run on the source cluster) shows all SnapMirror relationships where the local volume is the source. It identifies each destination SVM/volume paired with this source, confirming that the relationship to NetApp ONTAP is correctly registered.

```
sxId0edb1927eae795ba7::> snapmirror list-destinations
                                                    Progress

 Source             Destination         Transfer      Last         Relationship
 Path         Type  Path         Status Progress   Updated      Id
 ----------- ----- ------------ ------- --------- ------------ ---------------
 sx:Test_cifs_01 XDP SVM_data:Test_Vol Idle -    -            342e4af6-d02d-11f0-8278-
00a0b8ca4bc5
```

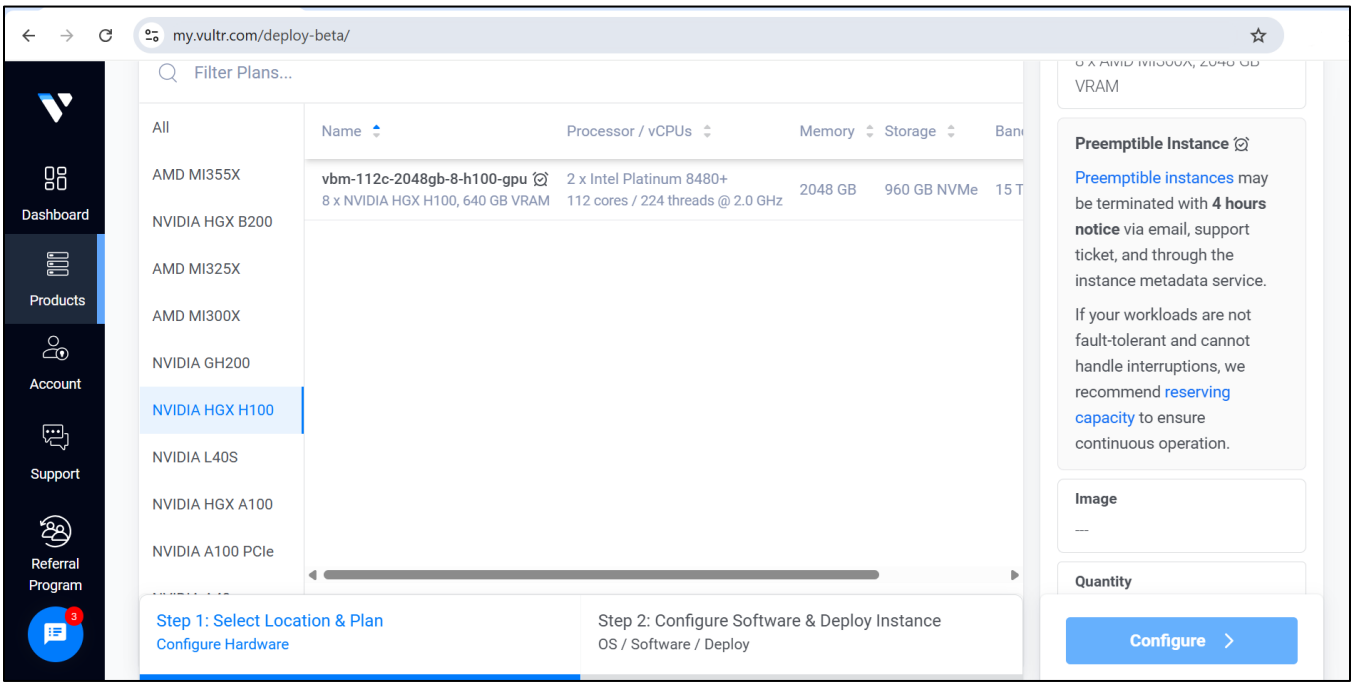
Create a compute node with NVIDIA GPUs

Provision a compute node with NVIDIA GPUs sized appropriately for your application and performance requirements. Deploy the compute node within the same Virtual Private Cloud (VPC) as the NetApp FlexCache destination system to ensure low-latency, local network access and optimal data performance.

Key Steps

- **Provision the Compute Node** – Deploy a GPU-enabled compute instance aligned with workload performance and capacity needs.
- **Select GPU Configuration** – Choose the appropriate NVIDIA model based on workload type, such as training, inference, or data processing.
- **Verify Network Connectivity** – Ensure connectivity between the compute node and storage system is established, with all required ports open for seamless data access.

The following screenshot illustrates some of the available NVIDIA GPU options:



Obtain the IP addresses of the Vultr compute nodes provisioned with NVIDIA GPUs, as these will be required for the configuration of export policies on the storage system.

```
root@Nvidia-Compute:~# hostname -I && hostname
149.28.218.96 10.12.96.8
nvidia-gpu-compute
root@Nvidia-Compute:~#
```

Configure Export/Security Policies (NFS/SMB)

Export policies define which compute nodes can mount the replicated dataset via NFS or SMB. Vultr compute instances must be explicitly granted read-only access to the destination volumes. We will be configuring the following:

- **export-policy create** – Creates an export policy that defines the overall access framework for NFS clients on the SVM.
- **export-policy rule create** – Adds specific access rules (read-only/read-write, security type, client conditions) to the policy so ONTAP knows what permissions to enforce.
- **volume modify -policy <policy-name>** – Applies the export policy to a specific volume, enabling ONTAP to enforce those access rules when the volume is mounted.
- **volume mount -junction-path <path>** – Mounts the volume into the SVM's namespace, making it visible and accessible for NFS clients to mount.

CLI to be run on Vultr Cloud on ONTAP (NFS example)

Create export policy:

```
volume show –volume <volume> -fields policy
```

```
ONTAPSelectCluster::> volume show Test_Vol -fields policy
 vserver  volume   policy
 -------- -------- -------
 SVM_data Test_Vol default
```

```
vserver export-policy create -vserver dst_svm -policyname <policy-name>
```

```
ONTAPSelectCluster::> export-policy create -policyname Test_Expolicy -vserver SVM_data
```

```
ONTAPSelectCluster::> export-policy show
 Vserver               Policy Name
 ---------------  -------------------
 SVM_data          Test_Expolicy
 SVM_data          default
 2 entries were displayed.
```

Add rule:

```
vserver export-policy rule create -vserver dst_svm -policyname <policy-name> \

-clientmatch <nfs_node_on_vultr> -rorule any -rwrule any -protocol nfs3 –superuser any
```

```
ONTAPSelectCluster::*> export-policy rule create -policyname Test_Expolicy -clientmatch
10.12.96.8 -rorule any -rwrule any -vserver SVM_data -protocol nfs3 -superuser any
```

```
export-policy rule show -policyname <policy-name>
```

```
ONTAPSelectCluster::> export-policy rule show -policyname Test_Expolicy -clientmatch 10.12.96.8
             Policy          Rule    Access   Client                RO
Vserver      Name            Index   Protocol Match                 Rule
------------ --------------- ------  -------- --------------------- ---------
SVM_data     Test_Expolicy   4       nfs3     10.12.96.8            any
```

Apply to volume:

The volume modify command assigns an export policy to the volume, enabling ONTAP to apply the correct access rules when clients attempt to mount it. Without an export policy bound to the volume, ONTAP will not allow any NFS access.

```
volume modify -vserver dst_svm -volume <vol> -policy <policy-name>
```

```
ONTAPSelectCluster::> volume modify -volume Test_Vol -vserver SVM_data -policy
Test_Expolicy

Volume modify successful on volume Test_Vol of Vserver SVM_data.
```

Volume mount

The volume mount command attaches the volume to the SVM's namespace at a junction path, making it visible and mountable by NFS clients. Without a junction path, the volume exists internally but cannot be accessed over NFS.

```
ONTAPSelectCluster::> volume show -volume Test_Vol -fields junction-path
 vserver     volume     junction-path
 ----------     ---------     -------------
 SVM_data Test_Vol   -
```

```
volume mount –volume <vol> -junction-path <junction-path>
```

```
ONTAPSelectCluster::> volume mount -volume Test_Vol -junction-path /Test_Vol
```

```
ONTAPSelectCluster::> volume show -volume Test_Vol -fields junction-path
 vserver     volume     junction-path
 ----------     ---------     -------------
 SVM_data Test_Vol   /Test_Vol
```

Check Export Policy access:

```
export-policy check-access -vserver <dst_svm> -volume <vol> -client-ip <compute_ip> -
authentication-method sys –protocol nfs3 –access-type read-write
```

```
ONTAPSelectCluster::> export-policy check-access -vserver SVM_data -volume Test_Vol -
client-ip 10.12.96.8 -authentication-method sys -protocol nfs3 -access-type read-write
                                Policy    Policy     Rule            Security
Path                Policy      Owner     Owner Type Index Access    Style
------------------- ---------- --------- ---------- ---- ---------- ----------
/                   default    SVM_data_root volume    3 read       unix
/Test_Vol           Test_Expolicy Test_Vol volume     4 read-write mixed
2 entries were displayed.
```

Test Read-Only Mount on Vultr Compute

Verification that Vultr compute nodes can access the replicated dataset over NFS. Ensures the end-to-end path - from **on-prem → SnapMirror → NetApp ONTAP(Destination) → Vultr compute** - is fully operational.

When we mount an NFS export from NetApp ONTAP in Vultr:

- We are mounting the **active filesystem** of the destination DP volume
- SnapMirror always maintains the **complete, consistent copy** of the source volume's data

- Clients see the **entire directory structure and files** exactly as they exist on-prem (at the time of last update)

The destination is read-only (DP mirror) unless we perform a **SnapMirror break**. But the data is **100% complete and usable** for analytics/AI workloads.

CLI (from compute instance on Vultr Cloud)

```
mount -t nfs <ontap_select_ip>:/<junction-path> <compute node mount point>
```

```
root@nvidia-gpu-compute:~# mount -t nfs 10.12.96.52:/Test_Vol /mnt/test
```

```
root@nvidia-gpu-compute:~# mount -t nfs 10.12.96.52:/Test_Vol /mnt/test
```

```
ls /mnt/data
```

```
root@nvidia-gpu-compute:/mnt/test# ls -l
total 1052720
-rw-r--r-- 1 root root 1073741824 Dec  4 11:35 1GB_testfile.img
-rw-r--r-- 1 root root       1599 Dec  5 10:34 file2.txt
-rw-r--r-- 1 root root        203 Dec  5 10:06 Test_file.txt
root@nvidia-gpu-compute:/mnt/test#
```

```
root@nvidia-gpu-compute:/mnt/test# ls -l
total 1052720
-rw-r--r-- 1 root root 1073741824 Dec  4 11:35 1GB_testfile.img
-rw-r--r-- 1 root root       1599 Dec  5 10:34 file2.txt
-rw-r--r-- 1 root root        203 Dec  5 10:06 Test_file.txt
root@nvidia-gpu-compute:/mnt/test#
```

Testing Read Access

```
cat /mnt/data/<file>
```

```
root@nvidia-gpu-compute:/mnt/test# cat Test_file.txt
Compute-vm has been created in Vultr cloud.

vCPU/s: 2 vCPUs
RAM: 4096.00 MB
Storage: 128 GB NVMe
OS: Ubuntu 24.04 LTS x64

Public IP Address: 104.238.182.xxxx
Username: root
Password: g3H.,toroot@nvidia-gpu-compute:/mnt/test#
```

```
root@nvidia-gpu-compute:/mnt/test# cat Test_file.txt
```

```
Compute-vm has been created in Vultr cloud.
vCPU/s: 2 vCPUs
RAM: 4096.00 MB
Storage: 128 GB NVMe
OS: Ubuntu 24.04 LTS x64
Public IP Address: 104.238.182.xxxx
Username: root
Password: g3H.,to
root@nvidia-gpu-compute:/mnt/test#
```

Testing Write Access (should fail as read-only mount point)

```
root@nvidia-gpu-compute:/mnt/test# echo "Vultr Cloud" >> Test_file.txt
-bash: Test_file.txt: Permission denied
```

```
root@nvidia-gpu-compute:/mnt/test# echo "Vultr Cloud" >> Test_file.txt

-bash: Test_file.txt: Permission denied
```

```
root@nvidia-gpu-compute:/mnt/test# cp Test_file.txt Test_file2.txt
cp: cannot create regular file 'Test_file2.txt': Read-only file system
```

```
root@nvidia-gpu-compute:/mnt/test# cp Test_file.txt Test_file2.txt

cp: cannot create regular file 'Test_file2.txt': Read-only file system
```

Expected Output

- Mount is successful
- Read operations succeed
- Write operations fail (read-only file system)

Validate Throughput and Performance

A performance test using read workloads to ensure ONTAP meets expected throughput for AI/analytics. Confirms the system can sustain the I/O patterns required by downstream GPU/compute pipelines.

CLI (compute-side example)

```
dd if=/mnt/Test/<bigfile> of=/dev/null bs=1M status=progress
```

```
root@nvidia-gpu-compute:/mnt/test# dd if=/mnt/test/1GB_testfile.img of=/dev/null bs=1M
status=progress
1051721728 bytes (1.1 GB, 1003 MiB) copied, 3 s, 350 MB/s
1024+0 records in
1024+0 records out
```

```
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 3.06939 s, 350 MB/s
```

```
root@nvidia-gpu-compute:/mnt/test# dd if=/mnt/test/1GB_testfile.img of=/dev/null bs=1M status=progress
1051721728 bytes (1.1 GB, 1003 MiB) copied, 3 s, 350 MB/s
1024+0 records in
1024+0 records out
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 3.06939 s, 350 MB/s
```

Expected Output

- Stable read throughput (e.g., 300–800 MB/s depending on VPC, storage, and VM size)
- No read stalls or NFS timeout messages

Step 4 - Production Ready Validation

A robust validation process ensures that the deployment is reliable, consistent, and production-ready. This confirms that replication works correctly under interruptions or planned failovers, and that performance meets expectations.

This section covers:

- Test SnapMirror update cycles
- Perform Replication Validation exercises:
    - SnapMirror break (switch to temporary read/write)
    - SnapMirror resync (restore mirror mode)
- Validate throughput and performance using representative workloads

Test SnapMirror Update Cycles

A manual SnapMirror update to confirm delta replication works. Validates ongoing synchronization for scheduled updates or event-driven refreshes.

CLI

```
snapmirror update -destination-path dst_svm:vol1
```

```
ONTAPSelectCluster::> snapmirror update -destination-path SVM_data:Test_Vol
Operation is queued: SnapMirror update of destination "SVM_data:Test_Vol".
```

```
snapmirror show -destination-path dst_svm:vol1
```

```
ONTAPSelectCluster::> snapmirror show
                                                                    Progress
Source            Destination Mirror  Relationship   Total            Last
Path        Type  Path        State   Status         Progress  Healthy Updated
----------- ---- ------------ ------- -------------- --------- ------- --------
sx:Test_cifs_01 XDP SVM_data:Test_Vol Snapmirrored Finalizing 21.19KB true 02/04 06:10:44
```

```
ONTAPSelectCluster::> snapmirror show
                                                                       Progress
 Source            Destination Mirror  Relationship   Total            Last
 Path        Type  Path        State   Status         Progress  Healthy Updated
 ----------- ---- ------------ ------- -------------- --------- ------- --------
 sx:Test_cifs_01 XDP SVM_data:Test_Vol Snapmirrored Idle -     true    -
```

Expected Output

- Update completes with no errors
- "Last transfer end timestamp" updates
- Status returns to Idle

Perform – SnapMirror Break

Before a SnapMirror **break**, the relationship must be **quiesced** to ensure all scheduled updates stop cleanly and no active transfers interfere with the transition to read/write mode.

Quiesce the SnapMirror Relationship

Quiesce stops future transfers and prevents new update jobs from starting.

CLI

```
snapmirror quiesce -destination-path dst:vol1
```

```
ONTAPSelectCluster::> snapmirror quiesce -destination-path SVM_data:Test_Vol
Operation succeeded: SnapMirror quiesce for destination "SVM_data:Test_Vol".
```

If the relationship is still transferring, run:

```
snapmirror show -destination-path dst:vol1
```

```
ONTAPSelectCluster::> snapmirror show
                                                                       Progress
 Source            Destination Mirror  Relationship   Total            Last
 Path        Type  Path        State   Status         Progress  Healthy Updated
 ----------- ---- ------------ ------- -------------- --------- ------- --------
 sx:Test_cifs_01 XDP SVM_data:Test_Vol Snapmirrored Quiesced - true    -
```

Once quiesced, you can safely make the destination volume writable.

Expected Output

- Relationship **state = Quiesced**
- No active transfers in progress
- No errors such as "Operation not permitted" or "Relationship is busy"

CLI

snapmirror break temporarily stops replication and makes the destination volume writable.

```
snapmirror break -destination-path dst:vol1
```

```
ONTAPSelectCluster::> snapmirror break -destination-path SVM_data:Test_Vol
Operation succeeded: SnapMirror break for destination "SVM_data:Test_Vol".
```

```
ONTAPSelectCluster::> snapmirror show
                                                                    Progress
Source            Destination Mirror  Relationship   Total            Last
Path        Type  Path        State   Status         Progress  Healthy Updated
----------- ---- ------------ ------- -------------- --------- ------- --------
sx:Test_cifs_01 XDP SVM_data:Test_Vol Broken-off Idle -       true    -
```

```
ONTAPSelectCluster::> vol show Test_Vol
Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- -----
SVM_data  Test_Vol     aggr_data    online     RW         25GB    24.99GB    0%
```

Expected Output

- Relationship state = **Broken-off**
- Volume becomes **read/write**
- No “cannot break” or access errors

Testing Write Access (should work as volume become read/write mount point)

We should be able to copy files or edit files in the mount volume

```
root@nvidia-gpu-compute:/mnt/test# cp Test_file.txt Test_file2.txt
root@nvidia-gpu-compute:/mnt/test#
```

```
root@nvidia-gpu-compute:/mnt/test# cp Test_file.txt Test_file2.txt

root@nvidia-gpu-compute:/mnt/test#
```

```
root@nvidia-gpu-compute:/mnt/test# ls -l
total 1052724
-rw-r--r-- 1 root root 1073741824 Dec  4 11:35 1GB_testfile.img
-rw-r--r-- 1 root root       1599 Dec  5 10:34 file2.txt
-rw-r--r-- 1 root root        203 Feb  4 06:24 Test_file2.txt
-rw-r--r-- 1 root root        203 Dec  5 10:06 Test_file.txt
root@nvidia-gpu-compute:/mnt/test#
```

```
root@nvidia-gpu-compute:/mnt/test# ls -l

total 1052724

-rw-r--r-- 1 root root 1073741824 Dec  4 11:35 1GB_testfile.img

-rw-r--r-- 1 root root       1599 Dec  5 10:34 file2.txt

-rw-r--r-- 1 root root        203 Feb  4 06:24 Test_file2.txt

-rw-r--r-- 1 root root        203 Dec  5 10:06 Test_file.txt

root@nvidia-gpu-compute:/mnt/test#
```

Perform – SnapMirror Resync

Resync reconnects the source and destination volumes and returns the system to mirror mode. Ensures the environment can return to normal, healthy replication after a failover test.

When a SnapMirror **resync** is initiated, the destination volume is reverted into a **mirror relationship** and ONTAP restores it to the state of the **source** based on the last common Snapshot copy. Any data that was written locally on the destination while it was in a **broken, read-write state** is discarded because SnapMirror must re-align the destination to match the source exactly. After resync completes, the destination becomes a **read-only mirror** again, fully synchronized with the primary system.

CLI

```
snapmirror resync -destination-path dst:vol1
```

```
ONTAPSelectCluster::> snapmirror resync -destination-path SVM_data:Test_Vol

Warning: All data newer than snapshot snapmirror.8e57bbd9-ceae-11f0-8278-00a0b8ca4bc5_2156550278.2026-02-04_061031 on volume
        SVM_data:Test_Vol will be deleted.
Do you want to continue? {y|n}: y
Operation is queued: initiate SnapMirror resync to destination "SVM_data:Test_Vol".
```

```
ONTAPSelectCluster::> snapmirror show
                                                                    Progress
Source            Destination Mirror  Relationship   Total            Last
Path        Type  Path        State   Status         Progress  Healthy Updated
----------- ---- ------------ ------- -------------- --------- ------- --------
sx:Test_cifs_01 XDP SVM_data:Test_Vol Snapmirrored Idle -     true    -
```

```
ONTAPSelectCluster::> volume show Test_Vol
Vserver   Volume       Aggregate    State      Type       Size  Available Used%
--------- ------------ ------------ ---------- ---- ---------- ---------- -----
SVM_data  Test_Vol     aggr_data    online     DP         25GB    24.99GB    0%
```

The mount point doesn't have the previous created Test_file3.txt after resync.

```
root@nvidia-gpu-compute:/mnt/test# ls -l
total 1052720
-rw-r--r-- 1 root root 1073741824 Dec  4 11:35 1GB_testfile.img
-rw-r--r-- 1 root root       1599 Dec  5 10:34 file2.txt
-rw-r--r-- 1 root root        203 Dec  5 10:06 Test_file.txt
root@nvidia-gpu-compute:/mnt/test#
```


```
root@nvidia-gpu-compute:/mnt/test# ls -l
total 1052720
-rw-r--r-- 1 root root 1073741824 Dec  4 11:35 1GB_testfile.img
-rw-r--r-- 1 root root       1599 Dec  5 10:34 file2.txt
-rw-r--r-- 1 root root        203 Dec  5 10:06 Test_file.txt
root@nvidia-gpu-compute:/mnt/test#
```


Expected Output

- State = Resyncing → **Snapmirrored**
- No conflict or snapshot mismatch errors
- Volume shows **DP**
- Replication resumes normally
- Destination volume will be
- File created on mount point (when snap-mirror was broken and read-write was enabled) will be lost

SnapMirror Health Validation

This section explains how to validate that SnapMirror replication is functioning correctly across all regions and between on-premises ONTAP systems and ONTAP in the cloud. It provides the essential commands and checks required to confirm relationship status, data transfer health, lag times, and peering connectivity. Proper health validation ensures consistent, predictable replication and prevents silent failures that could impact AI, analytics, or DR workflows.

This section covers

- Checking overall SnapMirror relationship state and transfer status
- Validating last transfer success and identifying any replication errors
- Measuring lag time for asynchronous policies
- Testing intercluster LIF connectivity between sites
- Confirming cluster and SVM peering health
- Reviewing transfer history and event logs for issues
- Verifying destination volumes are online and in DP (Data Protection) mode

Check Overall SnapMirror Relationship Status

```
snapmirror show
```

You should see:

- State: Snapmirrored
- Status: Idle or Transferring
- Last Transfer: successful
- Last Transfer Size: non-zero during updates

If the state is “Uninitialized,” “Error,” or “Broken-off,” replication is not healthy.

Validate the Most Recent Transfer

```
snapmirror show -fields last-transfer-type,last-transfer-end-timestamp,last-transfer-error
```

Confirm:

- last-transfer-error = "" (empty)
- last-transfer-end-timestamp = recent timestamp

Check Lag Time (for async policies)

```
snapmirror show -fields lag-time
```

Expected:

- Lag time matches your SnapMirror schedule
- Large or increasing lag may indicate bandwidth issues or blocked transfers

Validate Intercluster LIF Connectivity

From source cluster:

```
network ping -lif <source_intercluster_lif> -vserver <src_svm> -destination
<dest_intercluster_lif>
```

Expected:

- Successful ICMP replies
- Low latency for smooth transfers

Validate SVM and Cluster Peering

Cluster peering:

```
cluster peer show
```

Look for:

- Availability = Available
- Authentication Status = ok

SVM peering:

```
vserver peer show
```

Look for:

- Peer state = **peered**

Review Transfer History

```
snapmirror show-history -destination-path <dst_svm:dst_vol>
```

You should see a list of successful updates.

Validate Volume Status

```
volume show -vserver <dst_svm> -volume <volname>
```

Look for:

- State = online
- Type = DP (Data Protection)

Check for SnapMirror Errors Globally

```
event log show -severity error -message-name *snapmirror*
```

Expected:

- No recent SnapMirror-related errors

Troubleshooting

A structured approach to isolate problems across networking, routing, SnapMirror policy, and export access.

This section covers

- Peering problems, including failed intercluster or SVM peer relationships
- SnapMirror transfer failures caused by policy, scheduling, permissions, or connectivity issues
- Routing or firewall misconfigurations that block SnapMirror ports or intercluster LIF communication
- NetApp ONTAP connectivity issues related to VPC, VPN, or LIF configuration errors
- NFS/SMB export issues impacting read-only mount access from compute nodes

Peering Problems

Check cluster peering status

```
cluster peer show
```

Expected:

- Availability = Available
- Authentication = ok

- No error messages
- Remote cluster name populated

Check SVM peering

```
vserver peer show
```

Expected:

- Peer state = peered
- Applications = snapmirror
- No conflicts or pending states

Detailed peer status

```
cluster peer show -instance
```

Expected:

- No blocked ports
- No authentication failures
- Connection status = operational

SnapMirror Transfer Failures

Overall SnapMirror status

```
snapmirror show
```

Expected:

- State = Snapmirrored
- Status = Idle (or transferring during sync)
- Relationship healthy, no errors

Last transfer errors

```
snapmirror show -fields last-transfer-error
```

Expected:

- last-transfer-error = "" (empty)

Last update timestamps

```
snapmirror show -fields last-transfer-end-timestamp,last-transfer-type
```

Expected:

- Timestamp matches schedule window
- last-transfer-type = update

Transfer history

```
snapmirror show-history -destination-path <dst_svm:vol>
```

Expected:

- All entries show Success
- No "Transfer failed" events

Force update

```
snapmirror update -destination-path <dst_svm:vol>
```

Expected:

- Command completes without errors
- snapmirror show updates "Last Transfer Size" and timestamp

Routing or Firewall Misconfiguration

Ping intercluster LIFs

```
network ping -lif <src_ic_lif> -vserver <src_svm> -destination <dst_ic_lif>
```

Expected:

- Successful ping replies
- Low latency (single digit ms ideal)

Routing table

```
network route show
```

Expected:

- Route exists to destination intercluster network
- Gateway reachable
- No incorrect or overlapping routes

Firewall policy check

```
system services firewall policy show
```

Expected:

- SnapMirror service allowed
- No deny rules blocking intercluster traffic
- Relevant policies applied to the correct LIFs

Verify SnapMirror ports

```
network connections active show -service snapmirror
```

Expected:

- Active connections visible on ports 11104 and 11105
- State = Established

NetApp ONTAP Connectivity (VPC/VPN/LIF)

List all LIFs

```
network interface show
```

Expected:

- All LIFs in up/up state
- Correct roles: data / intercluster / mgmt

Home-node and home-port correctness

```
network interface show -fields home-node,home-port,is-home
```

Expected:

- is-home = true (unless failover active)

Cluster internal connectivity

```
cluster ping-cluster -node *
```

Expected:

- All nodes reachable
- No packet loss

Test VPN reachability

```
network ping -vserver <svm> -destination <onprem_vpn_gateway_ip>
```

Expected:

- Ping successful
- Round-trip latency consistent

NFS/SMB Export Issues

Export policies

```
vserver export-policy rule show
```

Expected:

- Client IP or subnet allowed
- Read-only permissions granted
- NFS version allowed (e.g., v3/v4)

Volume export path

```
volume show -fields junction-path,export-policy
```

Expected:

- Junction path = valid (e.g., /vol1)
- Export policy assigned correctly

List SMB shares

```
vserver cifs share show
```

Expected:

- Share listed (if SMB used)
- Correct permissions

Check client access

```
export-policy check-access -vserver <svm> -volume <vol> -client-ip <compute_ip> -
authentication-method sys
```

Expected:

- Access = read or read-only
- No “Access denied”

NFS service status

```
vserver nfs status
```

Expected:

- NFS enabled = true
- Active NFS versions listed