Learn about integrating ONTAP storage with KVM virtualization environments

NetApp virtualization solutions

NetApp
August 03, 2025

Table of Contents

Learn about integrating ONTAP storage with KVM virtualization environments

Enhance performance, data protection, and operational efficiency by integrating ONTAP storage with KVM virtualization environments using Libvirt. Discover how ONTAP's enterprise-grade storage features support both KVM host infrastructure and guest virtual machine storage requirements through flexible NFS, iSCSI, and Fibre Channel protocols.

Shared storage in KVM hosts reduces the time for VM live migration, and makes for a better target for backups and consistent templates across the environment. ONTAP storage can serve the needs of KVM host environments as well as for guest file, block and object storage demands.

KVM hosts need to have FC, Ethernet, or other supported interfaces cabled to switches and have communication to ONTAP logical interfaces. Always check Interoperability Matrix Tool for supported configurations.

High-level ONTAP Features

Common features

- Scale out Cluster
- Secure Authentication and RBAC support
- Zero trust multi admin support
- Secure Multitenancy
- Replicate data with SnapMirror.
- Point in time copies with Snapshots.
- Space efficient clones.
- Storage efficiency features like dedupe, compression, etc.
- Trident CSI support for Kubernetes
- Snaplock
- Tamperproof Snapshot copy locking
- Encryption support
- FabricPool to tier cold data to object store.
- BlueXP and Data Infrastructure Insights Integration.
- Microsoft offloaded data transfer (ODX)

NAS

- FlexGroup volumes are a scale out NAS container, providing high performance along with load distribution and scalability.
- FlexCache allows data to be distributed globally and still provides local read and write access to the data.
- Multiprotocol support enables the same data to be accessible via SMB, as well as NFS.
- NFS nConnect allows multiple TCP sessions per TCP connection increasing network throughput. This increases utilization of high speed nics available on modern servers.

- NFS session trunking provides increased data transfer speeds, high availability and fault tolerance.
- pNFS for optimized data path connection.
- SMB multichannel provides increased data transfer speed, high availability and fault tolerance.
- Integration with Active directory/LDAP for file permissions.
- Secure connection with NFS over TLS.
- NFS Kerberos support.
- NFS over RDMA.
- Name mapping between Windows and Unix identities.
- Autonomous ransomware protection.
- File System Analytics.

SAN

- Stretch cluster across fault domains with SnapMirror active sync.
- ASA models provide active/active multipathing and fast path failover.
- Support for FC, iSCSI, NVMe-oF protocols.
- Support for iSCSI CHAP mutual authentication.
- Selective LUN Map and Portset.

Libvirt with ONTAP Storage

Libvirt can be used to manage virtual machines that leverage NetApp ONTAP storage for their disk images and data. This integration allows you to benefit from ONTAP's advanced storage features, such as data protection, storage efficiency, and performance optimization, within your Libvirt-based virtualization environment.
Here's how Libvirt interacts with ONTAP and what you can do:

1. Storage Pool Management:
   - Define ONTAP storage as a Libvirt storage pool: You can configure Libvirt storage pools to point to ONTAP volumes or LUNs via protocols like NFS, iSCSI, or Fibre Channel.
   - Libvirt manages volumes within the pool: Once the storage pool is defined, Libvirt can manage the creation, deletion, cloning, and snapshotting of volumes within that pool, which correspond to ONTAP LUNs or files.
     - Example: NFS storage pool: If your Libvirt hosts mount an NFS share from ONTAP, you can define an NFS-based storage pool in Libvirt, and it will list the files in the share as volumes that can be used for VM disks.
2. Virtual Machine Disk Storage:
   - Store VM disk images on ONTAP: You can create virtual machine disk images (e.g., qcow2, raw) within the Libvirt storage pools that are backed by ONTAP storage.
   - Benefit from ONTAP's storage features: When VM disks are stored on ONTAP volumes, they automatically benefit from ONTAP's data protection (Snapshots, SnapMirror, SnapVault), storage efficiency (deduplication, compression), and performance features.
3. Data Protection:
   - Automated data protection: ONTAP offers automated data protection with features like Snapshots and SnapMirror, which can protect your valuable data by replicating it to other ONTAP storage, whether on-

premises, at a remote site, or in the cloud.

  - RPO and RTO: You can achieve low Recovery Point Objectives (RPO) and fast Recovery Time Objectives (RTO) using ONTAP's data protection features.
  - MetroCluster/SnapMirror active sync: For automated zero-RPO (Recovery Point Objective) and site-to-site availability, you can use ONTAP MetroCluster or SMas, which enables to have stretch cluster between sites.

4. Performance and Efficiency:
   - Virtio drivers: Use Virtio network and disk device drivers in your guest VMs for improved performance. These drivers are designed to cooperate with the hypervisor and offer paravirtualization benefits.
   - Virtio-SCSI: For scalability and advanced storage features, use Virtio-SCSI, which provides the ability to connect directly to SCSI LUNs and handle a large number of devices.
   - Storage efficiency: ONTAP's storage efficiency features, such as deduplication, compression, and compaction, can help reduce the storage footprint of your VM disks, leading to cost savings.
5. ONTAP Select Integration:
   - ONTAP Select on KVM: ONTAP Select, NetApp's software-defined storage solution, can be deployed on KVM hosts, providing a flexible and scalable storage platform for your Libvirt-based VMs.
   - ONTAP Select Deploy: ONTAP Select Deploy is a tool used to create and manage ONTAP Select clusters. It can be run as a virtual machine on KVM or VMware ESXi.

In essence, using Libvirt with ONTAP allows you to combine the flexibility and scalability of Libvirt-based virtualization with the enterprise-class data management features of ONTAP, providing a robust and efficient solution for your virtualized environment.

File based Storage Pool (with SMB or NFS)

Storage pool of type dir and netfs are applicable for file based storage.

Storage Protocol	dir	fs	netfs	logical	disk	iscsi	iscsi-direct	mpath
SMB/CIFS	Yes	No	Yes	No	No	No	No	No
NFS	Yes	No	Yes	No	No	No	No	No

With netfs, libvirt will mount the filesystem and supported mount options are limited. With dir storage pool, the mounting of filesystem needs to be handled externally on the host. fstab or automounter can be utilized for that purpose. To utilize automounter, autofs package needs to be installed. Autofs is particularly useful for mounting network shares on demand, which can improve system performance and resource utilization compared to static mounts in fstab. It automatically unmounts shares after a period of inactivity.

Based on storage protocol used, validate required packages are installed on the host.

Storage Protocol	Fedora	Debian	pacman
SMB/CIFS	samba-client/cifs-utils	smbclient/cifs-utils	smbclient/cifs-utils
NFS	nfs-utils	nfs-common	nfs-utils

NFS is a popular choice due to its native support and performance in Linux, while SMB is a viable option for integrating with Microsoft environments. Always check the support matrix before using it on production.

Based on protocol of choice, follow the appropriate steps to create the SMB share or NFS export.
SMB Share creation
NFS Export creation

Include mount options in either fstab or automounter configuration file. For example, with autofs, we included the following line in /etc/auto.master to use direct mapping using files auto.kvmfs01 and auto.kvmsmb01

/- /etc/auto.kvmnfs01 --timeout=60
/- /etc/auto.kvmsmb01 --timeout=60 --ghost

and in /etc/auto.kvmnfs01 file, we had
/mnt/kvmnfs01 -trunkdiscovery,nconnect=4 172.21.35.11,172.21.36.11(100):/kvmnfs01

for smb, in /etc/auto.kvmsmb01, we had
/mnt/kvmsmb01 -fstype=cifs,credentials=/root/smbpass,multichannel,max_channels=8
://kvmfs01.sddc.netapp.com/kvmsmb01

Define the storage pool using virsh of pool type dir.

```
virsh pool-define-as --name kvmnfs01 --type dir --target /mnt/kvmnfs01
virsh pool-autostart kvmnfs01
virsh pool-start kvmnfs01
```

Any existing VM disks can be listed using the

```
virsh vol-list kvmnfs01
```

For optimizing the performance of a Libvirt storage pool based on an NFS mount, all three options Session Trunking, pNFS, and the nconnect mount option can play a role, but their effectiveness depends on your specific needs and environment.
Here’s a breakdown to help you choose the best approach:

1. nconnect:
    - Best for: Simple, direct optimization of the NFS mount itself by using multiple TCP connections.
    - How it works: The nconnect mount option allows you to specify the number of TCP connections the NFS client will establish with the NFS endpoint (server). This can significantly improve throughput for workloads that benefit from multiple concurrent connections.
    - Benefits:
        - Easy to configure: Simply add nconnect=<number_of_connections> to your NFS mount options.
        - Improves throughput: Increases the "pipe width" for NFS traffic.
        - Effective for various workloads: Useful for general-purpose virtual machine workloads.
    - Limitations:
        - Client/Server support: Requires support for nconnect on both the client (Linux kernel) and the NFS server (e.g., ONTAP).
        - Saturation: Setting a very high nconnect value might saturate your network line.
        - Per-mount setting: The nconnect value is set for the initial mount and all subsequent mounts to the

    same server and version inherit this value.

2. Session Trunking:
    - Best for: Enhancing throughput and providing a degree of resiliency by leveraging multiple network interfaces (LIFs) to the NFS server.
    - How it works: Session trunking allows NFS clients to open multiple connections to different LIFs on an NFS server, effectively aggregating the bandwidth of multiple network paths.
    - Benefits:
        - Increased data transfer speed: By utilizing multiple network paths.
        - Resiliency: If one network path fails, others can still be used, although ongoing operations on the failed path might hang until the connection is re-established.
    - Limitations:
      Still a single NFS session: While it uses multiple network paths, it doesn’t change the fundamental single-session nature of traditional NFS.
    - Configuration complexity: Requires configuring trunking groups and LIFs on the ONTAP server.
      Network setup: Requires a suitable network infrastructure to support multipathing.
    - With nConnect option: Only the first interface will have nConnect option applied. Rest of the interface will have single connection.
3. pNFS:
    - Best for: High-performance, scale-out workloads that can benefit from parallel data access and direct I/O to the storage devices.
    - How it works: pNFS separates metadata and data paths, allowing clients to access data directly from the storage, potentially bypassing the NFS server for data access.
    - Benefits:
        - Improved scalability and performance: For specific workloads like HPC and AI/ML that benefit from parallel I/O.
        - Direct data access: Reduces latency and improves performance by allowing clients to read/write data directly from the storage.
        - with nConnect option: All the connections will have nConnect applied to maximize the network bandwidth.
    - Limitations:
        - Complexity: pNFS is more complex to set up and manage than traditional NFS or nconnect.
        - Workload specific: Not all workloads benefit significantly from pNFS.
        - Client support: Requires support for pNFS on the client side.

Recommendation:
* For general-purpose Libvirt storage pools on NFS: Start with the nconnect mount option. It’s relatively easy to implement and can provide a good performance boost by increasing the number of connections.
* If you need higher throughput and resiliency: Consider Session Trunking in addition to or instead of nconnect. This can be beneficial in environments where you have multiple network interfaces between your Libvirt hosts and your ONTAP system.
* For demanding workloads that benefit from parallel I/O: If you’re running workloads like HPC or AI/ML that can take advantage of parallel data access, pNFS might be the best option for you. However, be prepared for increased complexity in setup and configuration.
Always test and monitor your NFS performance with different mount options and settings to determine the optimal configuration for your specific Libvirt storage pool and workload.

Block based Storage Pool (with iSCSI, FC or NVMe-oF)

A dir pool type is often used on top of cluster filesystem like OCFS2 or GFS2 on a shared LUN or namespace.

Validate the host has necessary packages installed based on storage protocol used.

Storage Protocol	Fedora	Debian	pacman
iSCSI	iscsi-initiator-utils,device-mapper-multipath,ocfs2-tools/gfs2-utils	open-iscsi,multipath-tools,ocfs2-tools/gfs2-utils	open-iscsi,multipath-tools,ocfs2-tools/gfs2-utils
FC	device-mapper-multipath,ocfs2-tools/gfs2-utils	multipath-tools,ocfs2-tools/gfs2-utils	multipath-tools,ocfs2-tools/gfs2-utils
NVMe-oF	nvme-cli,ocfs2-tools/gfs2-utils	nvme-cli,ocfs2-tools/gfs2-utils	nvme-cli,ocfs2-tools/gfs2-utils

Collect host iqn/wwpn/nqn.

```
# To view host iqn
cat /etc/iscsi/initiatorname.iscsi
# To view wwpn
systool -c fc_host -v
# or if you have ONTAP Linux Host Utility installed
sanlun fcp show adapter -v
# To view nqn
sudo nvme show-hostnqn
```

Refer appropriate section to create the LUN or namespace.

LUN creation for iSCSI hosts
LUN creation for FC hosts
Namespace create for NVMe-oF hosts

Ensure FC Zoning or ethernet devices are configured to communicate with ONTAP logical interfaces.

For iSCSI,

```
# Register the target portal
iscsiadm -m discovery -t st -p 172.21.37.14
# Login to all interfaces
iscsiadm -m node -L all
# Ensure iSCSI service is enabled
sudo systemctl enable iscsi.service
# Verify the multipath device info
multipath -ll
# OCFS2 configuration we used.
o2cb add-cluster kvmcl01
o2cb add-node kvm02.sddc.netapp.com
o2cb cluster-status
mkfs.ocfs2 -L vmdata -N 4  --cluster-name=kvmcl01 --cluster-stack=o2cb -F
/dev/mapper/3600a098038314c57312b58387638574f
mount -t ocfs2 /dev/mapper/3600a098038314c57312b58387638574f1
/mnt/kvmiscsi01/
mounted.ocfs2 -d
# For libvirt storage pool
virsh pool-define-as --name kvmiscsi01 --type dir --target /mnt/kvmiscsi01
virsh pool-autostart kvmiscsi01
virsh pool-start kvmiscsi01
```

For NVMe/TCP, we used

```
# Listing the NVMe discovery
cat /etc/nvme/discovery.conf
# Used for extracting default parameters for discovery
#
# Example:
# --transport=<trtype> --traddr=<traddr> --trsvcid=<trsvcid> --host
-traddr=<host-traddr> --host-iface=<host-iface>
-t tcp -l 1800 -a 172.21.37.16
-t tcp -l 1800 -a 172.21.37.17
-t tcp -l 1800 -a 172.21.38.19
-t tcp -l 1800 -a 172.21.38.20
# Login to all interfaces
nvme connect-all
nvme list
# Verify the multipath device info
nvme show-topology
# OCFS2 configuration we used.
o2cb add-cluster kvmcl01
o2cb add-node kvm02.sddc.netapp.com
o2cb cluster-status
mkfs.ocfs2 -L vmdata1 -N 4  --cluster-name=kvmcl01 --cluster-stack=o2cb -F
/dev/nvme2n1
mount -t ocfs2 /dev/nvme2n1 /mnt/kvmns01/
mounted.ocfs2 -d
# To change label
tunefs.ocfs2 -L tme /dev/nvme2n1
# For libvirt storage pool
virsh pool-define-as --name kvmns01 --type dir --target /mnt/kvmns01
virsh pool-autostart kvmns01
virsh pool-start kvmns01
```

For FC,

```
# Verify the multipath device info
multipath -ll
# OCFS2 configuration we used.
o2cb add-cluster kvmcl01
o2cb add-node kvm02.sddc.netapp.com
o2cb cluster-status
mkfs.ocfs2 -L vmdata2 -N 4  --cluster-name=kvmcl01 --cluster-stack=o2cb -F
/dev/mapper/3600a098038314c57312b583876385751
mount -t ocfs2 /dev/mapper/3600a098038314c57312b583876385751 /mnt/kvmfc01/
mounted.ocfs2 -d
# For libvirt storage pool
virsh pool-define-as --name kvmfc01 --type dir --target /mnt/kvmfc01
virsh pool-autostart kvmfc01
virsh pool-start kvmfc01
```

NOTE:
The device mount should be included in /etc/fstab or use automount map files.

Libvirt manages the virtual disks (files) on top of the clustered file system. It relies on the clustered file system (OCFS2 or GFS2) to handle the underlying shared block access and data integrity. OCFS2 or GFS2 act as a layer of abstraction between the Libvirt hosts and the shared block storage, providing the necessary locking and coordination to allow safe concurrent access to the virtual disk images stored on that shared storage.

Copyright information

Trademark information