The Nutanix Bible - Classic Edition

[PDF generated January 03 2024. For all recent updates please see the Nutanix Bible releases notes located at https://nutanixbible.com/release_notes.html. Disclaimer: Downloaded PDFs may not always contain the latest information.]

» Download Nutanix Bible Classic Version as PDF (opens in a new tab/window)

Welcome to the classic edition of the Nutanix Bible! The purpose of the Nutanix Bible is to provide in-depth technical information about the Nutanix platform architecture.

Part 1: Core

Book of Basics

» Download this section as PDF (opens in a new tab/window)

web·scale - /web ' skāl/ - noun - computing architecture

a new architectural approach to infrastructure and computing.

Nutanix leverages "Web-scale" principles throughout our software stack. Web-scale doesn't mean you need to be as big as Google, Facebook, Amazon, or Microsoft in order to leverage them. Web-scale principles are applicable and beneficial at any scale, whether 3-nodes or thousands of nodes.

There are a few key constructs used when talking about "Web-scale" infrastructure:

- Hyper-convergence
- Software defined intelligence
- Distributed autonomous systems
- Incremental and linear scale out

Other related items:

- API-based automation and rich analytics
- Security as a core tenant
- Self-healing

This book will cover these basics as well as the core architectural concepts.

Strategy and Vision

» Download this section as PDF (opens in a new tab/window)

When Nutanix was conceived it was focused on one goal:

Make infrastructure computing invisible, anywhere.

This simplicity was to be achieved by focus in three core areas:

1. Enable choice and portability (HCI/Cloud/Hypervisor)
2. Simplify the “stack” through convergence, abstraction and intelligent software (AOS)
3. Provide an intuitive user interface (UI) through focus on user experience (UX) and design (Prism)

HCI/Cloud/Hypervisor: “The Choice”

Though we started with a single hardware platform (NX) supporting a single hypervisor (ESXi), we’ve always known we were more than a single hypervisor/platform/cloud company. This was one of the reasons we chose to build our own UI from scratch vs. run as a plug-in in vCenter, run as a VM vs. natively in the kernel (a lot more reasons there), etc. Why you may ask? Choice.

Not one hypervisor, platform, or cloud will fit all customer’s needs. By supporting multiple under the same platform we give the customer choice and leverage. By giving them the ability to move between them, we give them flexibility. All delivered with the same experience since it’s all part of the Nutanix platform.

We now have support for over 12 different hardware platforms (direct/OEM/third-party), multiple hypervisors (AHV, ESXi, Hyper-V, etc.), and expanding integrations with all of the major cloud vendors (AWS, Azure, GCP). This allows the customer to choose what is best for them, as well as use this for negotiations purposes between vendors.

NOTE: Platform is one key word that is used throughout the section and in general. We’re not trying to build one-off products, we’re building a platform.

The following shows a high-level architecture of the Nutanix platform:

Nutanix Platform - Architecture

AOS + AHV/Hypervisor: “The Runtime”

We started this journey by simplifying storage with a feature called the Distributed Storage Fabric (DSF then known as the Nutanix Distributed Filesystem aka NDFS), which combined local storage resources with intelligent software to provide “centralized storage” like capabilities.

Over the years, we’ve added a great deal of features and capabilities. To simplify things these have been broken down into two core areas:

1. Core Services
    - Foundational services
2. Platform Services
    - Services building upon core services providing additional capabilities/services

The core provides the foundational services and components that facilitate the running of workloads (VMs/Containers) and other higher-level Nutanix services. In the beginning this was just the DSF product, however we continue to expand the platform’s capabilities to help simplify and abstract the stack.

The following shows a high-level view of the AOS core platform:

Nutanix Platform - AOS Core

Over the years this has expanded into things like abstracting virtualization (we believe this should be something transparent and part of the system) by introducing our own hypervisor (AHV), simplifying upgrades, and providing other essential services like security and encryption.

With these capabilities we solved for a lot of the infrastructure level issues, but we didn't stop there. People still needed additional services like file shares, object storage, or containers.

Rather than requiring customers to use other vendors and products for some services we figured which ones we should partner on and which ones we should build ourselves. For backup we partnered with vendors like Veeam and Hycu, for others like file and object services we built them as services into the platform.

The following shows a high-level view of the Nutanix platform services:

Nutanix Platform - Services

Prism: "The Interface"

Nutanix Platform - Prism

Simply put, apply some of the design principles fostered by companies like Apple focused on simplicity, consistency and intuitiveness. Since the beginning we've invested significant time and effort on the Nutanix product's "front-end". Rather than being an afterthought, the UI/UX and design teams have always been pushing the boundaries. Case in point, we were one of the first enterprise software companies (besides the SaaS players), to have the management UI be written in HTML5.

Another core item here is the focus on providing a single interface for the platform and keeping the experience consistent throughout that. Our goal is to converge UIs like we've converged infrastructure. We want Prism to be a single interface allowing you to manage and consume the Nutanix platform, whether that is managing virtualization in your datacenter, Desktops-as-a-Service in the cloud, or providing spend visibility.

This is important as we continue to expand the platform through feature / service creation and acquisition. Rather than bolting the new capabilities on, we'd rather spend the time to natively integrate them into the platform. It is a slower process, but in the long run it keeps the experience consistent and reduces risk.

Nutanix: The Platform

To summarize, our vision is simple: "one platform, any app, any location". Thanks to our marketing team for providing this image; it fits perfectly and states our purpose succinctly.

Nutanix Platform - Architecture

This has been our goal from close to the beginning. Testament to this, below is an image created circa 2014 to talk about the Nutanix platform architecture. As you can see not much has changed, we just continue expanding and working towards this goal.

Nutanix Platform - Circa 2014

Products and Platforms

» Download this section as PDF (opens in a new tab/window)

Products

Nutanix has come a long way after pioneering Hyper-Converged Infrastructure (HCI). Over the years, Nutanix capabilities and services have grown substantially providing a complete Cloud Platform.

Nutanix Cloud Platform

The Nutanix Cloud Platform is a secure, resilient, and self-healing platform for building your hybrid multicloud infrastructure to support all kinds of workloads and use cases across public and private clouds, multiple hypervisors and containers, with varied compute, storage, and network requirements.

Nutanix Cloud Platform

The building blocks of Nutanix Cloud Platform are Nutanix Cloud Infrastructure and Nutanix Cloud Manager. These are the products that fall under each of them to form a complete solution.

Nutanix Cloud Infrastructure (NCI)

- AOS Scale-Out Storage
  - This is the core of Nutanix Cloud Platform and provides a distributed, performant, and resilient storage platform that scales linearly.
- AHV Hypervisor
  - Native enterprise class virtualization, management and monitoring capabilities are provided by AHV. The Nutanix Cloud Platform also supports ESXi and Hyper-V.
- Virtual Networking
  - AHV comes with standard VLAN-backed virtual networking. You can also enable Flow Virtual Networking to provide virtual private cloud (VPC) and other advanced networking constructs in AHV for enhanced isolation, automation, and multi-tenancy.
- Disaster Recovery
  - Integrated disaster recovery that is simple to deploy and easy to manage, providing flexible RPO and RTO options on-prem and in the cloud
- Container Services
  - Nutanix Cloud platform provides compute and data for container services like OpenShift and provides an enterprise Kubernetes management solution (Nutanix Kubernetes Engine), to deliver and manage an end-to-end production ready Kubernetes environment.
- Data and Network Security
  - Comprehensive security for data using encryption with a built-in local key manager and software-based firewalls for network and applications with Flow Network Security.

Nutanix Cloud Manager (NCM)

- AI Operations (NCM Intelligent Operations)
  - Optimizes capacity, proactively detects performance anomalies and provides capability to automate operational tasks that can streamline infrastructure management

- Self-Service Infrastructure/App Lifecycle Management (NCM Self-Service)
  - Orchestration with the ability to manage, deploy and scale applications across hybrid clouds with self-service, automation, and centralized role-based governance.

- Cost Governance
  - Drive financial accountability with intelligent resource sizing and accurate visibility into cloud metering and chargeback.

- Security Central
  - Security dashboard that unifies cloud security operations, identifies workload vulnerabilities, manages microsegmentation, and adheres to regulatory compliance requirements for strategic initiatives like Zero Trust.

In addition to NCI and NCM, the Nutanix Cloud Platform also provides the following services.

Nutanix Unified Storage Services

- Files Storage
  - Simple, Secure, software-defined scale-out file storage and management

- Objects Storage
  - Simple, Secure and scale-out S3 compatible object storage at massive scale

- Volumes Block Storage
  - Enterprise class scale-out block storage that exposes storage resources directly to virtualized guest operating systems or physical hosts

- Mine Integrated Backup
  - Natively integrated turnkey backup solution

Nutanix Database Service

Simplified and automated database lifecycle management across hybrid clouds

Desktop Services

Deliver virtual desktops and applications on any cloud at scale.

Nutanix Cloud Platform can be deployed on-prem, in public clouds using Nutanix Cloud Clusters (NC2), in a colocation, or at edge as required. For licensing and software options please visit Nutanix.com Software Options Page.

Platform

Nutanix supports a variety of vendor platforms and configurations. Platforms can be Nutanix appliances, OEM platforms and third-party server vendors for on-prem platforms. Additionally, Nutanix software also runs in public cloud platforms with NC2 and service provider clouds. For a complete list, please visit the Nutanix.com Platforms Page.

Hyperconverged Platform

» Download this section as PDF (opens in a new tab/window)

For a video explanation you can watch the following video: LINK

There are a few core structs for hyperconverged systems:

- Must converge and collapse the computing stack (e.g. compute + storage)

- Must shard (distribute) data and services across nodes in the system
- Must appear and provide the same capabilities as centralized storage (e.g. HA, live-migration, etc.)
- Must keep data as close to the execution (compute) as possible (Importance of Latency)
- Should be hypervisor agnostic
- Should be hardware agnostic

The following figure shows an example of a typical 3-tier stack vs. hyperconverged:

3-Tier vs. HCI

As you can see, the hyperconverged system does the following:

- Virtualizes and moves the controllers to the host
- Provides core services and logic through software
- Distributes (shards) data across all nodes in the system
- Moves the storage local to the compute

The Nutanix solution is a converged storage + compute solution which leverages local components and creates a distributed platform for running workloads.

Each node runs an industry-standard hypervisor (ESXi, AHV, and Hyper-V) and the Nutanix Controller VM (CVM). The Nutanix CVM is what runs the Nutanix software and serves all of the I/O operations for the hypervisor and all VMs running on that host.

The following figure provides an example of what a typical node logically looks like:

Converged Platform

The Nutanix CVM is responsible for the core Nutanix platform logic and handles services like:

- Storage I/O & transforms (Deduplication, Compression, EC)
- UI / API
- Upgrades
- DR / Replication
- Etc.

NOTE: Some services / features will spawn additional helper VMs or use the Microservices Platform (MSP). For example, Nutanix Files will deploy additional VMs, whereas Nutanix Objects will deploy VMs for MSP and leverage those.

For the Nutanix units running VMware vSphere, the SCSI controller, which manages the SSD and HDD devices, is directly passed to the CVM leveraging VM-Direct Path (Intel VT-d). In the case of Hyper-V, the storage devices are passed through to the CVM.

Virtualizing the Controller

The key reasons for running the Nutanix controllers as VMs in user-space really come down to four core areas:

1. Mobility
2. Resiliency
3. Maintenance / Upgrades
4. Performance, yes really

Since the beginning we knew we were more than a single platform company. In that sense, choice has always been a big thing for us, whether it is with hardware, cloud or hypervisor vendors.

By running as a VM in user-space it decouples the Nutanix software from the underlying hypervisor and hardware platforms. This enabled us to rapidly add support for other hypervisors while keeping the core code base the same across all operating environments (on-premises & cloud). Additionally, it gave us flexibility to not be bound to vendor specific release cycles.

Due to the nature of running as a VM in user-space, we can elegantly handle things like upgrades or CVM "failures" as they are outside of the hypervisor. For example, if there is some catastrophic issue where a CVM goes down, the whole node still continues to operate with storage I/Os and services coming from other CVMs in the cluster. During a AOS (Nutanix Core Software) upgrade, we can reboot the CVM without any impact to the workloads running on that host.

But isn't being in the kernel is so much faster? **Simple answer, NO.**

A common discussion topic is the debate around being in the kernel vs. in user-space. It is recommended to read through the "User vs. Kernel Space" section; it covers what both actually are and the pros and cons of each.

To summarize, there are two areas of execution in an operating system (OS): the kernel (privileged core of the OS where drivers may sit) and user space (where applications/ processes sit). Traditionally moving between user-space and the kernel (aka context switch) can be expensive in terms of CPU and time (~1,000ns / context switch).

The debate is that being in the kernel is always better / faster. Which is false. No matter what there will always be context switches in the guest VM's OS.

Distributed System

» Download this section as PDF (opens in a new tab/window)

There are three core principles for distributed systems:

1. Must have no single points of failure (SPOF)
2. Must not have any bottlenecks at any scale (must be linearly scalable)
3. Must leverage concurrency (MapReduce)

Together, a group of Nutanix nodes forms a distributed system (Nutanix cluster) responsible for providing the Prism and AOS capabilities. All services and components are distributed across all CVMs in a cluster to provide for high-availability and linear performance at scale.

The following figure shows an example of how these Nutanix nodes form a Nutanix cluster:

Nutanix Cluster - Distributed System

These techniques are also applied to metadata and data alike. By ensuring metadata and data is distributed across all nodes and all disk devices we can ensure the highest possible performance during normal data ingest and re-protection.

This enables our MapReduce Framework (Curator) to leverage the full power of the cluster to perform activities concurrently. Sample activities include that of data re-protection, compression, erasure coding, deduplication, etc.

The Nutanix cluster is designed to accommodate and remediate failure. The system will transparently handle and remediate the failure, continuing to operate as expected. The user will be alerted, but rather than being a critical time-sensitive item, any remediation (e.g. replace a failed node) can be done on the admin's schedule.

If you need to add additional resources to your Nutanix cluster, you can scale out linearly simply by adding new nodes. With traditional 3-tier architecture, simply adding additional servers will not scale out your storage performance. However, with a hyperconverged platform like Nutanix, when you scale out with new node(s) you're scaling out:

- The number of hypervisor / compute nodes
- The number of storage controllers
- The compute and storage performance / capacity
- The number of nodes participating in cluster wide operations

The following figure shows how the % of work handled by each node drastically decreases as the cluster scales:

Work Distribution - Cluster Scale

Key point: As the number of nodes in a cluster increases (cluster scaling), certain activities actually become more efficient as each node is handling only a fraction of the work.

Software Defined

» Download this section as PDF (opens in a new tab/window)

There are four core principles for software definition systems:

- Must provide platform mobility (hardware, hypervisor)
- Must not be reliant on any custom hardware
- Must enable rapid speed of development (features, bug fixes, security patches)
- Must take advantage of Moore's Law

As mentioned above (likely numerous times), the Nutanix platform is a software-based solution which ships as a bundled software + hardware appliance. The controller VM is where the vast majority of the Nutanix software and logic sits and was designed from the beginning to be an extensible and pluggable architecture. A key benefit to being software-defined and not relying upon any hardware offloads or constructs is around extensibility. As with any product life cycle, advancements and new features will always be introduced.

By not relying on any custom ASIC/FPGA or hardware capabilities, Nutanix can develop and deploy these new features through a simple software update. This means that the deployment of a new feature (e.g., deduplication) can be deployed by upgrading the current version of the Nutanix software. This also allows newer generation features to be deployed on legacy hardware models. For example, say you're running a workload running an older version of Nutanix software on a prior generation hardware platform (e.g., 2400). The running software version doesn't provide deduplication capabilities which your workload could benefit greatly from. To get these features, you perform a rolling upgrade of the Nutanix software version while the workload is running, and you now have deduplication. It's really that easy.

Similar to features, the ability to create new "adapters" or interfaces into DSF is another key capability. When the product first shipped, it solely supported iSCSI for I/O from the hypervisor, this has now grown to include NFS and SMB. In the future, there is the ability to create new adapters for various workloads and hypervisors (HDFS, etc.). And again, all of this can be deployed via a software update. This is contrary to most legacy infrastructures, where a hardware upgrade or software purchase is normally required to get the "latest and greatest" features. With Nutanix, it's different. Since all features are deployed in software, they can run on any hardware platform, any hypervisor, and be deployed through simple software upgrades.

The following figure shows a logical representation of what this software-defined controller framework looks like:

Software-Defined Controller Framework

Cluster Components

» Download this section as PDF (opens in a new tab/window)

The user-facing Nutanix product is extremely simple to deploy and use. This is primarily possible through abstraction and a lot of automation / integration in the software.

The following is a detailed view of the main Nutanix Cluster components (don't worry, no need to memorize or know what everything does):

Cassandra

- Key Role: Distributed metadata store
- Description: Cassandra stores and manages all of the cluster metadata in a distributed ring-like manner based upon a heavily modified Apache Cassandra. The Paxos algorithm is utilized to enforce strict consistency. This service runs on every node in the cluster. The Cassandra is accessed via an interface called Medusa.

Zookeeper

- Key Role: Cluster configuration manager
- Description: Zookeeper stores all of the cluster configuration including hosts, IPs, state, etc. and is based upon Apache Zookeeper. This service runs on three nodes in the cluster, one of which is elected as a leader. The leader receives all requests and forwards them to its peers. If the leader fails to respond, a new leader is automatically elected. Zookeeper is accessed via an interface called Zeus.

Stargate

- Key Role: Data I/O manager
- Description: Stargate is responsible for all data management and I/O operations and is the main interface from the hypervisor (via NFS, iSCSI, or SMB). This service runs on every node in the cluster in order to serve localized I/O.

Curator

- Key Role: MapReduce cluster management and cleanup
- Description: Curator is responsible for managing and distributing tasks throughout the cluster, including disk balancing, proactive scrubbing, and many more items. Curator runs on every node and is controlled by an elected Curator Leader who is responsible for the task and job delegation. There are two scan types for Curator, a full scan which occurs around every 6 hours and a partial scan which occurs every hour.

Prism

- Key Role: UI and API
- Description: Prism is the management gateway for component and administrators to configure and monitor the Nutanix cluster. This includes Ncli, the HTML5 UI, and REST API. Prism runs on every node in the cluster and uses an elected leader like all components in the cluster.

Genesis

- Key Role: Cluster component & service manager
- Description: Genesis is a process which runs on each node and is responsible for any services interactions (start/stop/etc.) as well as for the initial configuration. Genesis is a process which runs independently of the cluster and does not require the cluster to be configured/

running. The only requirement for Genesis to be running is that Zookeeper is up and running. The cluster_init and cluster_status pages are displayed by the Genesis process.

Chronos

- Key Role: Job and task scheduler
- Description: Chronos is responsible for taking the jobs and tasks resulting from a Curator scan and scheduling/throttling tasks among nodes. Chronos runs on every node and is controlled by an elected Chronos Leader that is responsible for the task and job delegation and runs on the same node as the Curator Leader.

Cerebro

- Key Role: Replication/DR manager
- Description: Cerebro is responsible for the replication and DR capabilities of DSF. This includes the scheduling of snapshots, the replication to remote sites, and the site migration/failover. Cerebro runs on every node in the Nutanix cluster and all nodes participate in replication to remote clusters/sites.

Pithos

- Key Role: vDisk configuration manager
- Description: Pithos is responsible for vDisk (DSF file) configuration data. Pithos runs on every node and is built on top of Cassandra.

Nondisruptive Upgrades

» Download this section as PDF (opens in a new tab/window)

In the ‘Nutanix Software Upgrade’ and ‘Hypervisor Upgrade’ sections in the Book of Prism, we highlighted the steps used to perform an upgrade of AOS and hypervisor versions. This section will cover the techniques allowing us to perform different types of upgrades in a non-disruptive manner.

AOS Upgrades

For an AOS upgrade there are a few core steps that are performed:

1 - Pre-upgrade Checks

During the pre-upgrade checks, the following items are verified. NOTE: This must complete successfully before an upgrade can continue.

- Check version compatibility between AOS, hypervisor versions
- Check cluster health (cluster status, free space, and component checks (e.g. Medusa, Stargate, Zookeeper, etc.)
- Check network connectivity between all CVMs and Hypervisors

2 - Upload upgrade software to 2 nodes

Once the pre-upgrade checks have been completed, the system will upload the upgrade software binaries to two nodes in the cluster. This is done for fault-tolerance and to ensure if one CVM is rebooting the other is available for others to pull the software from.

3 - Stage Upgrade Software

Once the software has been uploaded to two CVMs, all CVMs will stage the upgrade in parallel.

The CVMs have two partitions for AOS versions:

- Active partition (the currently running version)
- Passive partition (where upgrades are staged)

When an AOS upgrade occurs, we perform the upgrade on the non-active partition. When the upgrade token is received it will mark the upgraded partition as the active partition and reboot the CVM into the upgraded version. This is similar to a bootbank / altbootbank.

NOTE: the upgrade token is passed between nodes iteratively. This ensures only one CVM reboots at a time. Once the CVM reboots and is stable (check service status and communication) the token can be passed to the next CVM until all CVMs have been upgraded.

Upgrade Error Handling

A common question is what happens if the upgrade is unsuccessful or has an issue partially through the process?

In the event some upgrade issue occurs we will stall the upgrade and not progress. NOTE: this is a very infrequent occurrence as pre-upgrade checks will find most issues before the upgrade actually begins. However, in the event the pre-upgrade checks succeed and some issue occurs during the actual upgrade, there will be **no impact to workloads and user I/O running on the cluster.**

The Nutanix software is designed to work indefinitely in a mixed mode between supported upgrade versions. For example, if the cluster is running x.y.foo and is upgrading to x.y.bar the system can run indefinitely with CVMs on both versions. This is actually what occurs during the upgrade process.

For example, if you have a 4 node cluster on x.y.foo and start the upgrade to x.y.bar, when the first node upgrades it will be running x.y.bar while the others are on x.y.foo. This process will continue and CVMs will reboot into x.y.bar as they receive the upgrade token.

Foundation (Imaging)

» Download this section as PDF (opens in a new tab/window)

Foundation Imaging Architecture

Foundation is a Nutanix provided tool leveraged for bootstrapping, imaging and deployment of Nutanix clusters. The imaging process will install the desired version of the AOS software as well as the hypervisor of choice.

By default Nutanix nodes ship with AHV pre-installed, to leverage a different hypervisor type you must use foundation to re-image the nodes with the desired hypervisor. NOTE: Some OEMs will ship directly from the factory with the desired hypervisor.

The figure shows a high level view of the Foundation architecture:

Foundation - Architecture

As of 4.5, Foundation is built in to the CVMs to simplify configuration. The installer store is a directory for storing uploaded images, these can be used for the initial imaging as well as cluster expansion when imaging is required.

The Foundation Discovery Applet (which can be found HERE) is responsible for discovering nodes and allowing the user to select a node to connect to. Once the user has selected a node to connect to, the applet will proxy localhost:9442 IPv4 to the CVM’s IPv6 link-local address on port 8000.

The figure shows a high level view of the applet architecture:

Foundation - Applet Architecture

NOTE: the discovery applet is merely a means of discovery and proxy to the Foundation service which runs on the nodes. All of the imaging and configuration is handled by the Foundation service, not the applet.

Pro tip

If you're on a different network (L2) than your target Nutanix nodes (e.g. over the WAN) you can connect directly to the Foundation service on the CVM if it has an IPv4 address assigned (instead of using the discovery applet).

To directly connect browse to <CVM_IP>:8000/gui/index.html

Inputs

The Foundation tool has the following configuration inputs (below). A typical deployment requires 3 IP addresses per node (hypervisor, CVM, remote management (e.g. IPMI, iDRAC, etc.)). In addition to the per node addresses, it is recommended to set a Cluster and Data Services IP addresses.

- Cluster
  - Name
  - IP*
  - NTP*
  - DNS*

- CVM
  - IP per CVM
  - Netmask
  - Gateway
  - Memory

- Hypervisor
  - IP per hypervisor host
  - Netmask
  - Gateway
  - DNS
  - *Hostname prefix*

- *IPMI*
  - IP per node
  - Netmask
  - Gateway

NOTE: Items marked with '*' are optional but highly advisable

System Imaging and Deployment

The first step is to connect to the Foundation UI which can be done via the discovery applet (if on same L2, node IPs unecessary):

Foundation - Discovery Applet

If you can't find the desired node, make sure you're on the same L2 network.

After connecting into the selected node's Foundation instance the main Foundation UI will appear:

Foundation - Discovery Page

This will show all of the discovered nodes and their chassis. Select the desired nodes to form the cluster and click 'Next'

Foundation - Node Selection

The next page prompts for the cluster and network inputs:

Foundation - Cluster Information

Foundation - Network Information

Once the details have been input, click ‘Next’

Next we’ll input the node details and IP addresses:

Foundation - Node Setup

You can manually override the hostname and IP addresses if necessary:

Foundation - Hostname and IP

Click ‘Validate Network’ to validate network configuration and proceed. This will check for IP address conflicts and ensure connectivity.

Foundation - Network Validation

Once network validation has completed successfully we’ll now proceed to selecting the desired images.

To upgrade AOS to a newer version than currently on the CVM, download it from the portal and upload the Tarball. Once we have the desired AOS image, we’ll select the hypervisor.

For AHV, the image is built-in to the AOS image. For others you must upload the desired hypervisor image. NOTE: make sure the AOS and hypervisor versions are on the compatibility matrix (LINK).

Once we have the desired images, click ‘Create’:

Foundation - Select Images

If imaging is not necessary you can also click ‘Skip’ to skip the imaging process. This will not re-image the hypervisor or Nutanix cluster, but just configure the cluster (e.g. IP addresses, etc.).

Foundation will then proceed with the imaging (if necessary) and cluster creation process.

Foundation - Cluster Creation Process

Once the creation is successful you’ll get a completion screen:

Foundation - Cluster Creation Complete

At this point you can now log into any CVM or the Cluster IP and start using the Nutanix platform!

Drive Breakdown

» Download this section as PDF (opens in a new tab/window)

In this section, I'll cover how the various storage devices (Performance (NVMe/SSD) / Capacity (SSD/HDD) are broken down, partitioned, and utilized by the Nutanix platform. NOTE: All of the capacities used are in Base2 Gibibyte (GiB) instead of the Base10 Gigabyte (GB). Formatting of the drives with a filesystem and associated overheads has also been taken into account.

Performance Disk Devices

Performance devices are the highest performance device in a node. These can be NVMe or a mix of NVMe and SSD devices. They store a few key items, as explained below

- Nutanix Home (CVM core)
- Metadata (Cassandra / AES storage)
- OpLog (persistent write buffer)
- Extent Store (persistent storage)

The following figure shows an example of the storage breakdown for a Nutanix node's performance device:

Performance Drive Breakdown

Graphics and proportions aren't drawn to scale. When evaluating the Remaining GiB capacities, do so from the top down. For example, the Remaining GiB to be used for the OpLog calculation would be after Nutanix Home and Cassandra have been subtracted from the formatted SSD capacity.

Nutanix Home is mirrored across the first two SSDs to ensure availability and has a 60GiB reservation for two devices.

As of 5.0 Cassandra is sharded across multiple SSDs in the node (currently up to 4) with an initial reservation of 15GiB per SSD (can leverage some Stargate SSD if metadata usage increases). In dual SSD systems, metadata will be mirrored between the SSDs. The metadata reservation per SSD is 15 GiB (30GiB for dual SSD, 60GiB for 4+ SSD).

Prior to 5.0, Cassandra was on the first SSD by default, if that SSD fails the CVM will be restarted and Cassandra storage will then be on the 2nd. In this case the metadata reservation per SSD is 30 GiB for the first two devices.

The OpLog is distributed among all SSD devices up to a max of 12 per node (Gflag: max_ssds_for_oplog). If NVMe devices are available, OpLog will be placed on those devices instead of SATA SSD.

The OpLog reservation per disk can be calculated using the following formula: MIN(((Max cluster RF/2)*400 GiB)/ numDevForOplog), ((Max cluster RF/2)*25%) x Remaining GiB). NOTE: The sizing for OpLog is done dynamically as of release 4.0.1 which will allow the extent store portion to grow dynamically. The values used are assuming a completely utilized OpLog.

For example, in a RF2 (FT1) cluster with 8 SSD devices that are 1TB the result would be:

- MIN(((2/2)*400 GiB)/ 8), ((2/2)*25%) x ~900GiB) == MIN(50, 225) == 50 GiB reserved for Oplog per device.

For a RF3 (FT2) cluster this would be:

- MIN(((3/2)*400 GiB)/ 8), ((3/2)*25%) x ~900GiB) == MIN(75, 337) == 75 GiB reserved for Oplog per device.

For a RF2 (FT1) cluster with 4 NVMe and 8 SSD devices that are 1TB the result would be:

- MIN(((2/2)*400 GiB)/ 4), ((2/2)*25%) x ~900GiB) == MIN(100, 225) == 100 GiB reserved for Oplog per device.

The Extent Store capacity would be the remaining capacity after all other reservations are accounted for.

HDD Devices

Since HDD devices are primarily used for bulk storage, their breakdown is much simpler:

- Curator Reservation (Curator storage)
- Extent Store (persistent storage)

HDD Drive Breakdown

Book of Prism

» Download this section as PDF (opens in a new tab/window)

prism - /'prizəm/ - noun - control plane

one-click management and interface for datacenter operations.

Building a beautiful, empathetic and intuitive product is core to the Nutanix platform and something we take very seriously. This section will cover our design methodologies and how we iterate on design.

You can download the Nutanix Visio stencils here: http://www.visiocafe.com/nutanix.htm

Prism Architecture

» Download this section as PDF (opens in a new tab/window)

Prism is a distributed resource management platform which allows users to manage and monitor objects and services across their Nutanix environment, whether hosted locally or in the cloud.

These capabilities are broken down into two key categories:

- Interfaces
  - HTML5 UI, REST API, CLI, PowerShell CMDlets, etc.

- Management Capabilities
  - Platform management, VM / Container CRUD, policy definition and compliance, service design and status, analytics and monitoring

The following figure illustrates the conceptual nature of Prism as part of the Nutanix platform:

High-Level Prism Architecture

Prism is broken down into two main components:

- Prism Central (PC)
  - Multi-cluster manager responsible for managing multiple Nutanix Clusters to provide a single, centralized management interface. Prism Central is an optional software appliance (VM) which can be deployed in addition to the AOS Cluster (can run on it).
  - 1-to-many cluster manager

- Prism Element (PE)
  - Localized cluster manager responsible for local cluster management and operations. Every Nutanix Cluster has Prism Element built-in.
  - 1-to-1 cluster manager

The figure shows an image illustrating the conceptual relationship between Prism Central and Prism Element:

Prism Architecture

Prism Services

A Prism service runs on every CVM with an elected Prism Leader which is responsible for handling HTTP requests. Similar to other components which have a Leader, if the Prism Leader fails, a new one will be elected. When a CVM which is not the Prism Leader gets a HTTP request it will permanently redirect the request to the current Prism Leader using HTTP response status code 301.

Here we show a conceptual view of the Prism services and how HTTP request(s) are handled:

Prism Services - Request Handling

Prism ports

Prism listens on ports 80 and 9440, if HTTP traffic comes in on port 80 it is redirected to HTTPS on port 9440.

When using the cluster external IP (recommended), it will always be hosted by the current Prism Leader. In the event of a Prism Leader failure the cluster IP will be assumed by the newly elected Prism Leader and a gratuitous ARP (gARP) will be used to clean any stale ARP cache entries. In this scenario any time the cluster IP is used to access Prism, no redirection is necessary as that will already be the Prism Leader.

Pro tip

You can determine the current Prism leader by running the following command on any CVM:

```
curl localhost:2019/prism/leader
```

Authentication and Access Control (RBAC)

Authentication

Prism currently supports integrations with the following authentication providers:

- Prism Element (PE)
  - Local
  - Active Directory
  - LDAP

- Prism Central (PC)
  - Local
  - Active Directory
  - LDAP
  - SAML Authn (IDP)

SAML / 2FA

SAML Authn allows Prism to integrate with external identity providers (IDP) that are SAML compliant (e.g. Okta, ADFS, etc.).

This also allows you to leverage the multi-factor authentication (MFA) / two-factor authentication (2FA) capabilities these providers support for users logging into Prism.

Access Control

Coming soon!

Navigation

» Download this section as PDF (opens in a new tab/window)

Prism is fairly straight forward and simple to use, however we'll cover some of the main pages and basic usage.

Prism Central (if deployed) can be accessed using the IP address specified during configuration or corresponding DNS entry. Prism Element can be accessed via Prism Central (by clicking on a specific cluster) or by navigating to any Nutanix CVM or cluster IP (preferred).

Once the page has been loaded you will be greeted with the Login page where you will use your Prism or Active Directory credentials to login.

Prism Login Page

Upon successful login you will be sent to the dashboard page which will provide overview information for managed cluster(s) in Prism Central or the local cluster in Prism Element.

Prism Central and Prism Element will be covered in more detail in the following sections.

Prism Central

The figure shows a sample Prism Central dashboard where multiple clusters can be monitored / managed:

Prism Central - Dashboard

From here you can monitor the overall status of your environment, and dive deeper if there are any alerts or items of interest.

Prism Central contains the following main pages (NOTE: Search is the preferred / recommended method to navigation):

- Home Page
  - Environment wide monitoring dashboard including detailed information on service status, capacity planning, performance, tasks, etc. To get further information on any of them you can click on the item of interest.
- Virtual Infrastructure
  - Virtual entities (e.g. VMs, containers, Images, categories, etc.)
- Policies
  - Policy management and creation (e.g. security (FLOW), Protection (Backup/Replication), Recovery (DR), NGT)
- Hardware
  - Physical devices management (e.g. clusters, hosts, disks, GPU)
- Activity
  - Environment wide alerts, events and tasks
- Operations
  - Operations dashboards, reporting and actions (X-Play)
- Administration
  - Environment construct management (e.g. users, groups, roles, availability zones)
- Services
  - Add-on service management (e.g. Calm, Karbon)
- Settings
  - Prism Central configuration

To access the menu click on the hamburger icon::

Prism Central - Hamburger

The menu expands to display the available options:

Prism Central - Menu Bar

Search

Search is now the primary mechanism for Navigating the Prism Central UI (menus are still available).

To use the search bar to navigate you can use the search bar in the top left corner next to the menu icon.

Prism Central - Search

Search Semantics

PC Search allows for a great deal to semantics to be leveraged, some examples include:

Rule	Example
Entity type	vms

Entity type + metric perspective (io, cpu, memory)	vms io
Entity type + alerts	vm alerts
Entity type + alerts + alert filters	vm alerts severity=critical
Entity type + events	vm events
Entity type + events + event filters	vm events classification=anomaly
Entity type + filters (both metric and attribute)	vm “power state”=on
Entity type + filters + metric perspective (io, cpu, memory)	vm “power state”=on io
Entity type + filters + alerts	vm “power state”=on alerts
Entity type + filters + alerts + (alert filters)	vm “power state”=on alerts severity=critical
Entity type + filters + events	vm “power state”=on events
Entity type + filters + events + event filters	vm “power state”=on events classification=anomaly
Entity instance (name, ip address, disk serial etc)	vm1, 10.1.3.4, BHTXSPWRM
Entity instance + Metric perspective (io, cpu, memory)	vm1 io
Entity instance + alerts	vm1 alerts
Entity instance + alerts + alert filters	vm1 alerts severity=critical
Entity instance + events	vm1 events
Entity instance + events + event filters	vm1 events classification=anomaly
Entity instance + pages	vm1 nics, c1 capacity
Parent instance + entity type	c1 vms
Alert title search	Disk bad alerts
Page name search	Analysis, tasks

The prior is just a small subset of the semantics, the best way to get familiar with them is to give it a shot!

Prism Element

Prism Element contains the following main pages:

- Home Page
  - Local cluster monitoring dashboard including detailed information on alerts, capacity, performance, health, tasks, etc. To get further information on any of them you can click on the item of interest.
- Health Page
  - Environment, hardware and managed object health and state information. Includes NCC health check status as well.
- VM Page
  - Full VM management, monitoring and CRUD (AOS)
- Storage Page
  - Container management, monitoring and CRUD
- Hardware
  - Server, disk and network management, monitoring and health. Includes cluster expansion as well as node and disk removal.
- Data Protection
  - DR, Cloud Connect and Metro Availability configuration. Management of PD objects, snapshots, replication and restore.
- Analysis
  - Detailed performance analysis for cluster and managed objects with event correlation
- Alerts
  - Local cluster and environment alerts

The home page will provide detailed information on alerts, service status, capacity, performance, tasks, and much more. To get further information on any of them you can click on the item of interest.

The figure shows a sample Prism Element dashboard where local cluster details are displayed:

Prism Element - Dashboard

Keyboard Shortcuts

Accessibility and ease of use is a critical construct in Prism. To simplify things for the end-user a set of shortcuts have been added to allow users to do everything from their keyboard.

The following characterizes some of the key shortcuts:

Change view (page context aware):

- O - Overview View
- D - Diagram View
- T - Table View

Activities and Events:

- A - Alerts
- P - Tasks

Drop down and Menus (Navigate selection using arrow keys):

- M - Menu drop-down
- S - Settings (gear icon)
- F - Search bar
- U - User drop down
- H - Help

Features and Usage

» Download this section as PDF (opens in a new tab/window)

In the following sections we'll cover some of the typical Prism uses as well as some common troubleshooting scenarios.

Nutanix Software Upgrade

Performing a Nutanix software upgrade is a very simple and non-disruptive process.

To begin, start by logging into Prism and clicking on the gear icon on the top right (settings) or by pressing 'S' and selecting 'Upgrade Software':

Prism - Settings - Upgrade Software

This will launch the ‘Upgrade Software’ dialog box and will show your current software version and if there are any upgrade versions available.
 It is also possible to manually upload a NOS binary file.

You can then download the upgrade version from the cloud or upload the version manually:

Upgrade Software - Main

Upload software from the CVM

In certain cases you may want to download the software and upload from the CVM itself. For example, this can be used to download builds locally to the CVM.

First SSH into a CVM and find the Prism leader:

```
curl localhost:2019/prism/leader
```

SSH to the Prism leader and download the software bundle and metadata JSON

Run the following command to "upload" the software to Prism:

```
ncli software upload file-path=PATH_TO_SOFTWARE meta-file-path=PATH_TO_METADATA_JSON software-type=SOFTWARE_TYPE
```

The following shows an example for Prism Central:

```
ncli software upload file-path=/home/nutanix/tmp/leader-prism_central.tar meta-file-path=/home/nutanix/tmp/leader-prism_central-metadata.json
```

It will then upload the upgrade software onto the Nutanix CVMs:

Upgrade Software - Upload

After the software is loaded click on ‘Upgrade’ to start the upgrade process:

Upgrade Software - Upgrade Validation

You’ll then be prompted with a confirmation box:

Upgrade Software - Confirm Upgrade

The upgrade will start with pre-upgrade checks then start upgrading the software in a rolling manner:

Upgrade Software - Execution

Once the upgrade is complete you’ll see an updated status and have access to all of the new features:

Upgrade Software - Complete

> **Note**
>
> Your Prism session will briefly disconnect during the upgrade when the current Prism Leader is upgraded. All VMs and services running remain unaffected.

Hypervisor Upgrade

Similar to Nutanix software upgrades, hypervisor upgrades can be fully automated in a rolling manner via Prism.

To begin follow the similar steps above to launch the ‘Upgrade Software’ dialogue box and select ‘Hypervisor’.

You can then download the hypervisor upgrade version from the cloud or upload the version manually:

Upgrade Hypervisor - Main

It will then load the upgrade software onto the Hypervisors. After the software is loaded click on ‘Upgrade’ to start the upgrade process:

Upgrade Hypervisor - Upgrade Validation

You’ll then be prompted with a confirmation box:

Upgrade Hypervisor - Confirm Upgrade

The system will then go through host pre-upgrade checks and upload the hypervisor upgrade to the cluster:

Upgrade Hypervisor - Pre-upgrade Checks

Once the pre-upgrade checks are complete the rolling hypervisor upgrade will then proceed:

Upgrade Hypervisor - Execution

Similar to the rolling nature of the Nutanix software upgrades, each host will be upgraded in a rolling manner with zero impact to running VMs. VMs will be live-migrated off the current host, the host will be upgraded, and then rebooted. This process will iterate through each host until all hosts in the cluster are upgraded.

Pro tip

You can also get cluster wide upgrade status from any Nutanix CVM by running 'host_upgrade --status'. The detailed per host status is logged to ~/data/logs/host_upgrade.out on each CVM.

Once the upgrade is complete you’ll see an updated status and have access to all of the new features:

Upgrade Hypervisor - Complete

Cluster Expansion (add node)

Cluster Expansion

The ability to dynamically scale the Nutanix cluster is core to its functionality. To scale an Nutanix cluster, rack / stack / cable the nodes and power them on. Once the nodes are powered up they will be discoverable by the current cluster using mDNS.

The figure shows an example 7 node cluster with 1 node which has been discovered:

Add Node - Discovery

Multiple nodes can be discovered and added to the cluster concurrently.

Once the nodes have been discovered you can begin the expansion by clicking ‘Expand Cluster’ on the upper right hand corner of the ‘Hardware’ page:

Hardware Page - Expand Cluster

You can also begin the cluster expansion process from any page by clicking on the gear icon:

Gear Menu - Expand Cluster

This launches the expand cluster menu where you can select the node(s) to add and specify IP addresses for the components:

Expand Cluster - Host Selection

After the hosts have been selected you’ll be prompted to upload a hypervisor image which will be used to image the nodes being added. For AHV or cases where the image already exists in the Foundation installer store, no upload is necessary.

Expand Cluster - Host Configuration

After the upload is completed you can click on ‘Expand Cluster’ to begin the imaging and expansion process:

Expand Cluster - Execution

The job will then be submitted and the corresponding task item will appear:

Expand Cluster - Execution

Detailed tasks status can be viewed by expanding the task(s):

Expand Cluster - Execution

After the imaging and add node process has been completed you’ll see the updated cluster size and resources:

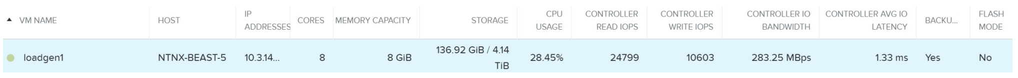
Expand Cluster - Execution

I/O Metrics

Identification of bottlenecks is a critical piece of the performance troubleshooting process. In order to aid in this process, Nutanix has introduced a new ‘I/O Metrics’ section to the VM page.

Latency is dependent on multitude of variables (queue depth, I/O size, system conditions, network speed, etc.). This page aims to offer insight on the I/O size, latency, source, and patterns.

To use the new section, go to the ‘VM’ page and select a desired VM from the table. Here we can see high level usage metrics:



▲ VM NAME	HOST	IP ADDRESSES	CORES	MEMORY CAPACITY	STORAGE	CPU USAGE	CONTROLLER READ IOPS	CONTROLLER WRITE IOPS	CONTROLLER IO BANDWIDTH	CONTROLLER AVG IO LATENCY	BACKU...	FLASH MODE
● loadgen1	NTNX-BEAST-5	10.3.14...	8	8 GiB	136.92 GiB / 4.14 TiB	28.45%	24799	10603	283.25 MBps	1.33 ms	Yes	No

VM Page - Details

The ‘I/O Metrics’ tab can be found in the section below the table:

VM Page - I/O Metrics Tab

Upon selecting the ‘I/O Metrics’ tab a detailed view will be shown. We will break this page down and how to use it in this section.

The first view is the ‘Avg I/O Latency’ section that shows average R/W latency for the past three hours. By default the latest reported values are shown with the corresponding detailed metrics below for that point in time.

You can also mouse over the plot to see the historical latency values and click on a time of the plot to view the detailed metrics below.

I/O Metrics - Latency Plot

This can be useful when a sudden spike is seen. If you see a spike and want to investigate further, click on the spike and evaluate the details below.

I/O Metrics - Latency Plot

If latency is all good, no need to dig any further.

The next section shows a histogram of I/O sizes for read and write I/Os:

I/O Metrics - I/O Size histogram

Here we can see our read I/Os range from 4K to 32K in size:

I/O Metrics - Read I/O Size histogram

Here we can see our write I/Os range from 16K to 64K with some up to 512K in size:

I/O Metrics - Write I/O Size histogram

Pro tip

If you see a spike in latency the first thing to check is the I/O size. Larger I/Os (64K up to 1MB) will typically see higher latencies than smaller I/Os (4K to 32K).

The next section shows a histogram of I/O latencies for read and write I/Os:

I/O Metrics - Latency histogram

Looking at the read latency histogram we can see the majority of read I/Os are sub-ms (<1ms) with some up to 2-5ms.

I/O Metrics - Read Latency histogram

Taking a look below at the ‘Read Source’ we can see most I/Os are being served from the SSD tier:

I/O Metrics - Read Source SSD

As data is read it will be pulled in to the Unified Cache realtime (Check the ‘I/O Path and Cache’ section to learn more). Here we can see the data has been pulled into the cache and is now being served from DRAM:

I/O Metrics - Read Source DRAM

We can now see basically all of our read I/Os are seeing sub-ms (<1ms) latency:

I/O Metrics - Read Latency histogram

Here we can see the majority of our write I/O are seeing <1-2ms latency:

I/O Metrics - Write Latency histogram

Pro tip

If you see a spike in read latency and the I/O sizes aren't large, check where the read I/Os are being served from. Any initial read from HDD will see higher latency than the DRAM cache; however, once it is in the cache all subsequent reads will hit DRAM and see an improvement in latency.

The last section shows the I/O patterns and how much is random vs. sequential:

I/O Metrics - RW Random vs. Sequential

Typically I/O patterns will vary by application or workload (e.g. VDI is mainly random, whereas Hadoop would primarily be sequential). Other workloads will be a mix of both. For example, a database might be random on inserts or some queries, however sequential during ETL.

Microservices Infrastructure

» Download this section as PDF (opens in a new tab/window)

Microservices Infrastructure (MSP) provides a common framework for delivering microservices associated with Prism Central-based components such as Flow Virtual Networking, Objects, and the Security Dashboard. MSP also provides services such as Identity and Access Management and internal service load balancing.

Before MSP, Prism Central was a monolithic application. With MSP enabled, certain services are migrated and spun up in a Kubernetes (K8s) cluster as pods. Over time, most of the Prism Central services will be migrated and converted to microservices leveraging the common platform. This will enable services to be upgraded independently through LCM, so particular services can be upgraded without having to upgrade the entire PC instance, resulting in faster upgrades and quick patch updates.

As of PC.2022.9, the Microservices Infrastructure is enabled by default. When you upgrade to the latest version of Prism Central, MSP will be automatically enabled. Refer to the Prism Central MSP documentation for the full list of prerequisites and considerations.

Network Configuration

When MSP is enabled, a Kubernetes cluster is created on Prism Central. This Kubernetes cluster is one node with a standalone Prism Central and three nodes with a scale-out Prism Central.

MSP uses the following subnets.

Subnet	Purpose
10.100.0.0/16	Reserved for K8s pod network
10.200.32.0/24	K8s Services (Flow Virtual Networking, IAM, etc)
10.200.0.0/16	Reserved for K8s Services network
10.100.0.0/24	K8s pod - PC1
10.100.1.0/24	K8s pod - PC2
10.100.2.0/24	K8s pod - PC3

If you already use these subnets for DNS or Active Directory and require different IP ranges, contact Nutanix Support.

On the firewall, bidirectional traffic should be allowed between Prism Central and:

- All of the Prism Element CVM IPs
- Prism Element Virtual IPs
- Prism Element Data Services IPs

On TCP ports:

- 3205 - iSCSI data plane connection
- 3260 - iSCSI control plane connection
- 9440 - Prism UI/API

Additionally, Prism Central should be able to ping all the Prism Element CVM IPs and Prism Element Virtual IPs.

During deployment, Prism Central must access several cloud services over port 443. For the most up-to-date list and diagrams, refer to the Ports & Protocols page on the Support Portal. There is also a dark site method of deployment for networks without Internet access, which is covered in the Prism Central MSP documentation.

When deploying Prism Central or enabling MSP, there are two options for the internal network configuration.

- Private Network (default): an internal private network is deployed using VXLAN for Kubernetes node communication. The default settings use 192.168.5.0/24. This requires no additional IPs from the physical network.

- VLAN Network: a managed or unmanaged VLAN network that has been configured on the Nutanix cluster. Five IPs are needed from the physical network for a single-node PC setup and 10 IPs are needed for a three-node scale-out PC setup. This option will add a second NIC to every PC VM for Kubernetes node communication.

Microservices Setup During Prism Central Deployment

Architecture

The diagrams below assume the default private VXLAN network.

An example architecture diagram for the Kubernetes node on a single-PC deployment looks like the following. Note that the example does not show every service running in the Kubernetes cluster.

In this example, 10.10.250.50 is the IP address assigned to the PC VM and 10.100.0.0/24 is used as the network for the Kubernetes pod.

Single Node PC

For scale-out PC, two additional Kubernetes nodes are provisioned and they use the 10.100.1.0/24 and 10.100.2.0/24 networks for their pods, respectively.

Scale-out PC

In a scale-out PC setup with VXLAN, pod-to-pod traffic across nodes uses direct routing and is encapsulated using the Prism Central VM IP.

Traffic from a pod on PC VM 1 to a pod on PC VM 3

Prism Central Backup & Restore

» Download this section as PDF (opens in a new tab/window)

Prism Central Backup and Restore enables the continuous backup of your Prism Central deployment and various service configurations and policies to Prism Element clusters managed by that same Prism Central. If the cluster hosting your Prism Central instance experiences a disaster, you can recover your PC deployment on another cluster.

Architecture

On Prism Element and Prism Central, there is a database called the Insights Data Fabric (IDF) that stores configuration, periodic performance, and metric data from various components and services of the infrastructure.

When Prism Central Backup & Restore is enabled, data from the Prism Central IDF instance is periodically replicated to the Prism Element IDF instance that has been designated for backup. You can select up to three Nutanix clusters, running AHV or ESXi, as PC backup targets. The replication happens every 30 minutes. Port 9440 is used for the replication.

When a restore operation occurs, Prism Central is rebuilt on one of the backup Prism Element clusters and re-seeded with the data from the IDF data backup.

In this architecture diagram, we have Prism Central managing three clusters with Prism Central hosted on Cluster 1. Note that these clusters could be in different physical locations. The clusters chosen for backup can be running either ESXi or AHV as long as they are running AOS 6.0 or later, are managed by Prism Central, and at least one cluster needs to be at least AOS 6.5.3.1. Additionally, NTP must be configured on the PC to synchronize time between the PC and the registered clusters.

When enabling Prism Central Backup & Restore, the IDF data from Prism Central gets periodically synced to the IDF database on the chosen cluster(s) over TCP port 9440. In this example, Prism Central is hosted on Cluster 1 and is replicating to both Cluster 2 and Cluster 3.

See the Prism Central Backup and Restore documentation for the full list of prerequisites along with which services and data are synced and restored.

If Cluster 1 is unavailable, Prism Central also becomes unavailable, and you can recover Prism Central from the Prism Element interface of either Cluster 2 or Cluster 3.

During the recovery process, ensure you use the new Prism Central deployment. If Cluster 1 comes back online, be sure to **shut down or delete** the original Prism Central. After recovery, replication continues to the same backup targets.

Book of APIs

» Download this section as PDF (opens in a new tab/window)

The HTML5 UI is a key part to Prism to provide a simple, easy to use management interface. However, another core ability are the APIs which are available for automation. All functionality exposed through the Prism UI is also exposed through a full set of REST APIs to allow for the ability to programmatically interface with the Nutanix platform. This allows customers and partners to enable automation, 3rd-party tools, or even create their own UI.

Core to any dynamic or “software-defined” environment, Nutanix provides a vast array of interfaces allowing for simple programmability and interfacing. Here are the main interfaces:

- REST API
- CLI - aCLI & nCLI
- Scripting interfaces

Nutanix.dev - Nutanix Developer Portal

To learn more about the available Nutanix Prism and product APIs, review sample code and go through self-paced labs, be sure to check out https://www.nutanix.dev!

Resources and Scripts:

- Nutanix.dev (Developer Portal): https://www.nutanix.dev
- Nutanix.dev Github - https://github.com/nutanixdev
- Nutanix.dev Code Samples - https://github.com/nutanixdev/code-samples
- Nutanix.dev Technical Articles - https://www.nutanix.dev/blog/
- Nutanix v4 API Documentation: https://developers.nutanix.com
- Nutanix Github - https://github.com/nutanix

Disclaimer: All code samples are © Nutanix, Inc., and are provided as-is under the MIT license (https://opensource.org/licenses/MIT).

REST APIs

» Download this section as PDF (opens in a new tab/window)

REST API Introduction

The REST API exposes every capability and data point of the Prism UI and allows for orchestration or automation tools to easily drive Nutanix action. This enables tools like Saltstack, Puppet, vRealize Operations, System Center Orchestrator, Ansible and more to easily create custom workflows for Nutanix. Also, this means that any third-party developer could create their own custom UI and pull in Nutanix data via REST.

The following figure shows a small snippet of the Nutanix REST API explorer which allows developers to interact with the API and see expected data formats.

REST API Explorer

The REST API Explorer is a way of reading API usage info while logged into Prism Element or Prism Central. The API endpoints shown will vary depending on the current login session: Prism Element or Prism Central.

For example, a user currently logged into Prism Central may need to send an API request that creates a virtual machine. The Prism Central v3 **vms** API reference can be quickly viewed in the REST API Explorer.

Prism Element REST API Explorer

Operations can be expanded to display details and examples of the REST call:

Prism Central REST API Explorer

REST API Authentication

Nutanix REST APIs support HTTP Basic Authentication for all endpoints. Authentication may be completed using two methods:

1. Read-only: To collect and inspect information exposed by API only.
2. Administrative: All actions available to Read-Only user accounts plus entity management, including but not exclusive to CRUD operations on virtual machines, NCM Self Service Blueprints and Nutanix Flow Network Security policies.

REST API Versions

Nutanix Prism Element and Prism Central currently offer two generally available APIs:

- v2.0, available in Prism Element only. Used for cluster-local activities such as storage container management operations.
- v3, available in Prism Central only. Used for multicluster activities such as virtual machine management, Nutanix product management including NCM Self Service (formerly Calm), Flow Networy Security.

For detailed information on the available API versions and the operations they expose, see API Versions on Nutanix.dev.

Nutanix Products Providing APIs

A non-exhaustive list of Nutanix products that can be fully or partially managed by API is as follows. Documentation is linked, where appropriate.

- Nutanix Prism Element
- Nutanix Prism Central

- NCM Self Service (formerly Calm)
- Nutanix Database Service (formerly Era)

For a complete list of all API references, see Nutanix.dev API Reference.

Prism Element v2.0 vs Prism Central v3 APIs

The chosen API will depend on the required operations.

Prism Element v2.0 is cluster-local and available through Prism Element only. It exposes operations specific to the local cluster entities such as storage containers, data-at-rest encryption, storage pools and hosts.

Prism Central v3 is multi-cluster and available through Prism Central only. It exposes operations that can impact multiple clusters such as distributed disk images, NCM Self Service (formerly Calm) blueprints and apps, marketplace items and network security rules.

Usage Examples: Prism API v2.0

Nutanix Prism API v2.0 is available through Prism Element only. All API requests are as constructed as follows.

```
METHOD https://CVM_OR_CLUSTER_IP:9440/api/nutanix/v2.0/API_NAME/VARIABLES
```

Example of a request to list all storage containers, assuming the cluster IP is 192.168.1.100.

```
GET https://192.168.1.100:9440/api/nutanix/v2.0/storage-containers
```

Example of VM creation:

```
POST https://192.168.1.100:9440/api/nutanix/v2.0/vms
```

Accompanied by VM configuration in JSON format. For example:

```
{
    "description": "VM created by v2.0 API",
    "memory_mb": 1024,
    "name": "vm_api_v2.0",
    "num_vcpus": 1,
    "num_cores_per_vcpu": 1,
    "vm_disks": [
        {
            "is_cdrom": false,
            "vm_disk_create": {
                "size": 128849018880,
                "storage_container_uuid": "b2fefc62-6274-4c84-8e6c-a61f5313ea0e"
            }
        }
    ]
}
```

Response

The response from this request contains a Prism task UUID that can be queried using the **tasks** API.

```
{
    "task_uuid": "eec7d743-b6c7-4c6d-b821-89b91b142f1d"
}
```

To use this example in your own environment, change **storage_container_uuid** to a value appropraite for your cluster.

For more information see Create a Virtual Machine in the Nutanix.dev API reference.

Note: The **/api/nutanix/v2.0** path is an alias for **/PrismGateway/services/rest/v2.0**. The **PrismGateway** path will often be referenced in the Prism API v2.0 documentation; both paths can be used interchangeably and will produce identical results.

Usage Examples: Prism API v3.0

Nutanix Prism API v3 is available through Prism Central only. All API requests are as constructed as follows.

```
METHOD https://PRISM_CENTRAL_IP:9440/api/nutanix/v3/API_NAME/VARIABLES
```

Example of a request to list all virtual machines, assuming the Prism Central IP is 192.168.1.110.

```
POST https://192.168.1.110:9440/api/nutanix/v3/vms/list
```

Accompanied by an appropriate payload in JSON format. In this example Prism Central is instructed to return all items of type **vm**.

```
{
    "kind": "vm"
}
```

Example of VM creation:

```
POST https://192.168.1.110:9440/api/nutanix/v3/vms
```

Accompanied by VM configuration in JSON format. For example:

```
{
    "spec": {
        "name": "vm_api_v3",
        "resources": {},
        "cluster_reference": {
            "uuid": "0005f2f7-eee7-1995-6145-ac1f6b35fe5e",
            "kind": "cluster"
        }
    },
    "metadata": {
        "kind": "vm"
    }
}
```

The response from this request contains the task status, currently **PENDING**, and task details including the cluster that owns the new VM and a **task_uuid** that can be queried using the **tasks** API.

To use this example in your own environment, change the cluster **uuid** to a value appropraite for your cluster.

For more information see, Create a new VM in the Nutanix.dev API reference.

CLI - nCLI and aCLI

» Download this section as PDF (opens in a new tab/window)

aCLI

The Acropolis CLI (aCLI) is the CLI for managing the Acropolis and AHV portion of the Nutanix product for tasks like AHV host, network, and VM management. These capabilities were enabled in releases after AOS 4.1.2 and are available on Nutanix AHV cluster CVMs only. **aCLI** is not supported on Prism Central.

The current aCLI command reference can be found on the Nutanix Portal.

Enter aCLI Shell

Description: Enter aCLI shell.

```
acli
```

OR

Description: Execute aCLI command via Linux shell

```
acli <command>
```

Output aCLI Response in JSON Format

Description: Enter aCLI shell. Any responses to commands will be in JSON format.

```
acli -o json
```

List AHV Hosts

Description: Lists AHV nodes in the cluster.

```
host.list
```

This screenshot shows the output from ‘host.list’ in both tabulated and JSON format. The obfuscated information are cluster specific serial numbers and IP address details.

Tabulated vs JSON output from aCLI

Create Network

Description: Create network based on VLAN

```
net.create NAME TYPE.ID[.VSWITCH] ip_config=A.B.C.D/NN vlan="VLAN"
```

Example:

```
net.create vlan.133 ip_config=10.1.1.1/24 vlan="133"
```

List Networks

Description: List networks

```
net.list
```

Create DHCP Scope

Description: Create dhcp scope

```
net.add_dhcp_pool NET NAME start=START IP A.B.C.D end=END IP W.X.Y.Z
```

Note: The last usable IP address in the network range is selected for the Acropolis DHCP server if an address for the DHCP server wasn’t set during network creation.

Example:

```
net.add_dhcp_pool vlan.100 start=10.1.1.100 end=10.1.1.200 vlan="100"
```

Get Existing Network Details

Description: Get a network’s VMs and details including VM name / UUID, MAC address and IP

```
net.list_vms NETNAME
```

Example:

```
net.list_vms vlan.133
```

Configure DHCP DNS Servers for Network

Description: Set DHCP DNS

```
net.update_dhcp_dns NETNAME servers=COMMA SEPARATED DNS IPs domains=COMMA SEPARATED DOMAINS
```

Example:

```
net.update_dhcp_dns vlan.100 servers=10.1.1.1,10.1.1.2 domains=ntnxlab.local
```

Create Virtual Machine

Description: Create VM

```
vm.create COMMA SEPARATED VM NAMES memory=NUM MEM MB num_vcpus=NUM VCPU num_cores_per_vcpu=NUM CORES ha_priority=PRIORITY INT
```

Example:

```
vm.create testVM memory=2G num_vcpus=2
```

Bulk Create Virtual Machines

Description: Create bulk VMs

```
vm.create CLONEPREFIX[STARTING INT..END INT] memory=NUM MEM MB num_vcpus=NUM VCPU num_cores_per_vcpu=NUM CORES ha_priority=PRIORITY INT
```

Example:

```
vm.create testVM[000..999] memory=2G num_vcpus=2
```

Clone VM from Existing

Description: Create clone of existing VM

```
vm.clone CLONE NAME(S) clone_from_vm=SOURCE VM NAME
```

Example:

```
vm.clone testClone clone_from_vm=MYBASEVM
```

Bulk Clone VMs from Existing VM

Description: Create bulk clones of existing VM

```
vm.clone CLONEPREFIX[STARTING INT..END INT] clone_from_vm=SOURCE VM NAME
```

Example:

```
vm.clone testClone[001..999] clone_from_vm=MYBASEVM
```

Create Disk and Add to VM

Description: Create disk for OS

```
vm.disk_create VM NAME create_size=Size and qualifier, e.g. 500G container=CONTAINER NAME
```

Example:

```
vm.disk_create testVM create_size=500G container=default
```

Add NIC to VM

Description: Create and add NIC

```
vm.nic_create VM NAME network=NETWORK NAME model=MODEL
```

Example:

```
vm.nic_create testVM network=vlan.100
```

Set VM Boot Device to Disk

Description: Set a VM boot device

Set to boot from specific disk id

```
vm.update_boot_device VM NAME disk_addr=DISK BUS
```

Example:

```
vm.update_boot_device testVM disk_addr=scsi.0
```

Add CD-ROM to VM

```
vm.disk_create VM NAME cdrom="true" empty="true"
```

Example:

```
vm.disk_create testVM cdrom="true" empty="true"
```

Set VM Boot Device to CD-ROM

Set to boot from CD-ROM

```
vm.update_boot_device VM NAME disk_addr=CD-ROM BUS
```

Example:

```
vm.update_boot_device testVM disk_addr=ide.0
```

Mount ISO to CD-ROM

Description: Mount ISO to VM CD-ROM

Steps:

1. Upload ISOs to container
2. Enable whitelist for client IPs
3. Upload ISOs to share

Create CD-ROM with ISO

```
vm.disk_create VM NAME clone_from_afsf_file=/CONTAINER/ISO CD-ROM=true
```

Example:

```
vm.disk_create testVM clone_from_adfs_file=/default/myfile.iso CD-ROM=true
```

Detach ISO from CD-ROM

Description: Remove ISO from CD-ROM

```
vm.disk_update VM NAME CD-ROM BUS empty=true
```

Power On VM(s)

Description: Power on VM(s)

```
vm.on VM NAME(S)
```

Example:

```
vm.on testVM
```

Power on all VMs.

Example:

```
vm.on *
```

Power on all VMs matching a prefix.

Example:

```
vm.on testVM*
```

Power on range of VMs.

Example:

```
vm.on testVM[0-9][0-9]
```

nCLI

The current nCLI command reference can be found on the Nutanix Portal.

The Nutanix Command Line Interface (nCLI) allows you to run system administration commands against the Nutanix cluster. In contrast to aCLI, nCLI can be installed on your local machine. See the Nutanix Portal link above for installation details.

Display Nutanix Version

Description: Displays the current version of the Nutanix software

```
ncli cluster version
```

This screenshot shows the output from ‘ncli version’ as both a single-line command and from within an ncli “session”.

ncli command usage options

Add Subnet to NFS whitelist

Description: Adds a particular subnet to the NFS whitelist

```
ncli cluster add-to-nfs-whitelist ip-subnet-masks=10.2.0.0/255.255.0.0
```

List Storage Pools

Description: Displays the existing storage pools

```
ncli sp ls
```

Note: This example shows the use of shortened commands. “storagepool” becomes “sp”, “list” becomes “ls”.

List Storage Containers

Description: Displays the existing containers

```
ncli ctr ls
```

Create Storage Container

Description: Creates a new container

```
ncli ctr create name=NAME sp-name=SP NAME
```

List VMs

Description: Displays the existing VMs

```
ncli vm ls
```

List Public Keys

Description: Displays the existing public keys

Add Public Key

Description: Adds a public key for cluster access

SCP public key to CVM

Add public key to cluster

```
ncli cluster add-public-key name=myPK file-path=~/mykey.pub
```

Remove Public Key

Description: Removes a public key for cluster access

```
ncli cluster remove-public-keys name=myPK
```

Create Protection Domain

Description: Creates a protection domain

```
ncli pd create name=NAME
```

Create Remote Site

Description: Create a remote site for replication

```
ncli remote-site create name=NAME address-list=Remote Cluster IP
```

Create Protection Domain For All VMs In Storage Container

Description: Protect all VMs in the specified container

```
ncli pd protect name=PD NAME ctr-id=Container ID cg-name=NAME
```

Create Protection Domain With Specified VMs

Description: Protect the VMs specified

```
ncli pd protect name=PD NAME vm-names=VM Name(s) cg-name=NAME
```

Create Protection Domain for AOS files (aka vDisk)

Description: Protect the DSF Files specified

```
ncli pd protect name=PD NAME files=File Name(s) cg-name=NAME
```

Create Protection Domain Snapshot

Description: Create a one-time snapshot of the protection domain

Create Snapshot and Replication Schedule to Remote Site

Description: Create a recurring snapshot schedule and replication to n remote sites

```
ncli pd set-schedule name=PD NAME interval=seconds retention-policy=POLICY remote-sites=REMOTE SITE NAME
```

List Replication Status

Description: Monitor replication status

```
ncli pd list-replication-status
```

Migrate Protection Domain to Remote Site

Description: Fail-over a protection domain to a remote site

```
ncli pd migrate name=PD NAME remote-site=REMOTE SITE NAME
```

Activate Protection Domain

Description: Activate a protection domain at a remote site

```
ncli pd activate name=PD NAME
```

Check Cluster Resiliency Status

```
# Node status
ncli cluster get-domain-fault-tolerance-status type=node
```

```
# Block status
ncli cluster get-domain-fault-tolerance-status type=rackable_unit
```

PowerShell Cmdlets

» Download this section as PDF (opens in a new tab/window)

PowerShell Cmdlets

To get started with Nutanix PowerShell Cmdlets, see PowerShell Cmdlets Reference on the Nutanix Support Portal.

The below will cover the Nutanix PowerShell Cmdlets, how to use them and some general background on Windows PowerShell.

PowerShell Versions

At the time of writing, the latest available version is PowerShell 7.3.4

Commands outlined in this Nutanix Bible chapter may not apply to earlier PowerShell versions.

Nutanix Cmdlets Versions

At the time of writing, the latest available version is Nutanix Cmdlets v2.0

Commands outlined in this Nutanix Bible chapter may not apply to other versions of the Nutanix Cmdlets.

PowerShell Basics

Windows PowerShell is a powerful shell and scripting language built on the .NET framework. It is a very simple to use language and is built to be intuitive and interactive. Within PowerShell there are a few key constructs/Items:

Cmdlets

Cmdlets are commands or .NET classes which perform a particular operation. They are usually conformed to the Getter/Setter methodology and typically use a Verb-Noun based structure. For example: Get-Process, Set-Partition, etc.

Piping Or Pipelining

Piping is an important construct in PowerShell (similar to its use in Linux) and can greatly simplify things when used correctly. With piping you're essentially taking the output of one section of the pipeline and using that as input to the next section of the pipeline. The pipeline can be as long as required (assuming there remains output which is being fed to the next section of the pipe). A very simple example could be getting the current processes, finding those that match a particular trait or filter and then sorting them:

```
Get-Service | where {$_.Status -eq "Running"} | Sort-Object Name
```

Piping can also be used in place of for-each, for example:

```
# For each item in my array
$myArray | %{
  # Do something
}
```

Key Object Types

Below are a few of the key object types in PowerShell. You can easily get the object type by using the .getType() method, for example: $someVariable.getType() will return the objects type.

Variables

```
$myVariable = "foo"
```

Note: You can also set a variable to the output of a series or pipeline of commands:

```
$myVar2 = (Get-Service | where {$_.Status -eq "Running"})
```

```
$myVar3 = (Get-Process | where {$_.ProcessName -eq "Chrome"})
```

In this example the commands inside the parentheses will be evaluated first then variable will be the outcome of that.

Array

```
$myArray = @("Value","Value")
```

Note: You can also have an array of arrays, hash tables or custom objects

Hash Table

```
$myHash = @{"Key" = "Value";"Key" = "Value"}
```

Useful Commands

Get the help content for a particular Cmdlet (similar to a man page in Linux)

```
Get-Help Cmdlet Name
```

Example:

```
Get-Help Get-Process
```

List properties and methods of a command or object

```
<expression or object> | Get-Member
```

Example:

```
$someObject | Get-Member
```

Core Nutanix Cmdlets And Usage

The Nutanix Cmdlets can be downloaded directly from the Prism Central UI and can be found on the drop down in the upper right hand corner:

Nutanix Cmdlets v2.0 - Usage Examples

The current Nutanix Cmdlet v2.0 reference can be found on the Nutanix Portal.

List Nutanix Cmdlets

```
Get-Command -Module Nutanix.Prism.Common
```

```
Get-Command -Module Nutanix.Prism.Ps.Cmds
```

Connect To Prism Central

Prompt for all connection details and accept valid SSL certificates only:

```
Connect-PrismCentral
```

Supply connection details (excluding password), accept valid and invalid SSL certificates:

```
Connect-PrismCentral -AcceptInvalidSSLCerts -Server <prism_central_ip_addres> -UserName <username>
```

List All Virtual Machines

```
Get-VM
```

Get Nutanix VMs Matching A Certain String

Assign to variable:

```
$searchString = "my-vm-name"
$vms = Get-VM | where {$_.vmName -match $searchString}
```

Interactive:

```
$myVm = Get-VM | where {$_.vmName -match "myVmName"}
```

Interactive and formatted:

```
$myVm = Get-VM | where {$_.vmName -match "myVmName"} | ft
```

List All Storage Containers

```
Get-StorageContainer
```

Get Storage Container With Similar Names

```
Get-StorageContainer | where {$_.name -like "Nutanix*"}
```

List All Networks

```
Get-Network
```

Get Specific Network By Name

```
Get-Network | where ${_.name -eq "Default"}
```

Disconnect From Prism Central

```
Disconnect-PrismCentral
```

Nutanix Cmdlets v1.0 - Usage Examples

The Nutanix Cmdlet v1.0 reference can be found on the Nutanix Portal.

Load Nutanix Snapin

Check if snapin is loaded and if not, load

```
if ( (Get-PSSnapin -Name NutanixCmdletsPSSnapin -ErrorAction SilentlyContinue) -eq $null )
{
    Add-PsSnapin NutanixCmdletsPSSnapin
}
```

List Nutanix Cmdlets

```
Get-Command | Where-Object{$_.PSSnapin.Name -eq "NutanixCmdletsPSSnapin"}
```

Connect To A Nutanix cluster

```
Connect-NutanixCluster -Server $server -UserName "myuser" -Password (Read-Host "Password: " -AsSecureString) -AcceptInvalidSSLCerts
```

Get Nutanix VMs Matching A Certain Search String

Set to variable

```
$searchString = "myVM"
$vms = Get-NTNXVM | where {$_.vmName -match $searchString}
```

Interactive

```
Get-NTNXVM | where {$_.vmName -match "myString"}
```

Interactive and formatted

```
Get-NTNXVM | where {$_.vmName -match "myString"} | ft
```

Get Nutanix vDisks

Set to variable

```
$vdisks = Get-NTNXVDisk
```

Interactive

```
Get-NTNXVDisk
```

Interactive and formatted

```
Get-NTNXVDisk | ft
```

Get Nutanix Storage Containers

Set to variable

```
$containers = Get-NTNXContainer
```

Interactive

```
Get-NTNXContainer
```

Interactive and formatted

```
Get-NTNXContainer | ft
```

Get Nutanix Protection Domains

Assign to variable

```
$pds = Get-NTNXProtectionDomain
```

Interactive

```
Get-NTNXProtectionDomain
```

Interactive and formatted

```
Get-NTNXProtectionDomain | ft
```

Get Nutanix Consistency Groups

Set to variable

```
$cgs = Get-NTNXProtectionDomainConsistencyGroup
```

Interactive

```
Get-NTNXProtectionDomainConsistencyGroup
```

Interactive and formatted

```
Get-NTNXProtectionDomainConsistencyGroup | ft
```

Book of AOS

» Download this section as PDF (opens in a new tab/window)

a·crop·o·lis - /ə ‘ kräpəlis/ - noun - data plane

storage, compute and virtualization platform.

The Acropolis Operating System (AOS) is the core software stack that provides the abstraction layer between the hypervisor (running on-premises or in the cloud) and the workloads running. It provides functionality such as storage services, security, backup and disaster recovery, and much more. This book will cover this functionality, as well as the architecture of AOS.

Architecture

» Download this section as PDF (opens in a new tab/window)

The Acropolis Operating System (AOS) provides the core functionality leveraged by workloads and services running on the platform. This includes, but isn’t limited to, things like storage services, upgrades, etc.

The figure highlights an image illustrating the conceptual nature of AOS at various layers:

High-level AOS Architecture

Building upon the distributed nature of everything Nutanix does, we’re expanding this into the virtualization and resource management space. AOS is a back-end service that allows for workload and resource management, provisioning, and operations. Its goal is to abstract the facilitating resource (e.g., hypervisor, on-premises, cloud, etc.) from the workloads running, while providing a single “platform” to operate.

This gives workloads the ability to seamlessly move between hypervisors, cloud providers, and platforms.

Supported Hypervisors for VM Management

As of 4.7, AHV and ESXi are the supported hypervisors for VM management, however this may expand in the future. The Volumes API and read-only operations are still supported on all.

Acropolis Services

An Acropolis Worker runs on every CVM with an elected Acropolis Leader which is responsible for task scheduling, execution, IPAM, etc. Similar to other components which have a Leader, if the Acropolis Leader fails, a new one will be elected.

The role breakdown for each can be seen below:

- Acropolis Leader
  - Task scheduling & execution
  - Stat collection / publishing
  - Network Controller (for hypervisor)
  - VNC proxy (for hypervisor)
  - HA (for hypervisor)

- Acropolis Worker
    - Stat collection / publishing
    - VNC proxy (for hypervisor)

Here we show a conceptual view of the Acropolis Leader / Worker relationship:

Acropolis Services

Dynamic Scheduler

Efficient scheduling of resources is critical to ensure resources are effectively consumed. The AOS Dynamic Scheduler extends the traditional means of scheduling that relies upon compute utilization (CPU/MEM) to make placement decisions. It leverages compute, as well as storage and others to drive VM and volume (ABS) placement decisions. This ensures that resources are effectively consumed and end-user performance is optimal.

Resource scheduling can be broken down into two key areas:

- Initial placement
    - Where an item is scheduled at power-on

- Runtime Optimization
    - Movement of workloads based upon runtime metrics

The original AOS Scheduler had taken care of the initial placement decisions since its release. With its release in AOS 5.0, the AOS Dynamic Scheduler expands upon this to provide runtime resources optimization.

The figure shows a high-level view of the scheduler architecture:

AOS Dynamic Scheduler

The dynamic scheduler runs consistently throughout the day to optimize placement (currently every 15 minutes	Gflag: lazan_anomaly_detection_period_secs). Estimated demand is calculated using historical utilization values and fed into a smoothing algorithm. This estimated demand is what is used to determine movement, which ensures a sudden spike will not skew decisions.

A different approach towards resource optimization

When you look at existing scheduling / optimization platforms (VMware DRS, Microsoft PRO) they are all focused on balancing workloads / VMs evenly across cluster resources. NOTE: how aggressively it tries to eliminate skew is determined by the balancing configuration (e.g. manual -> none, conservative -> some, aggressive -> more).

For example, say we had 3 hosts in a cluster, each of which is utilized 50%, 5%, 5% respectively. Typical solutions would try to re-balance workloads to get each hosts utilization ~20%. But why?

What we're really trying to do is eliminate / negate any contention for resources, not eliminate skew. Unless there is contention for resources there is no positive gain from "balancing" workloads. In fact by forcing unnecessary movement we cause additional requisite work (e.g. memory transfer, cache re-localization, etc.), all of which consumes resources.

The AOS Dynamic Scheduler does just this, it will only invoke workload movement if there is expected contention for resources, not because of skew. NOTE: DSF works in a different way and works to ensure uniform distribution of data throughout the cluster to eliminate hot spots and speed up rebuilds. To learn more of DSF, check out the 'disk balancing' section.

At power-on ADS will balance VM initial placement throughout the cluster.

At power-on ADS will balance VM initial placement throughout the cluster.

Placement Decisions

Placement decisions are based upon the following items:

- Compute utilization
  -

We monitor each individual node’s compute utilization. In the event where a node’s expected CPU allocation breaches its threshold (currently 85% of host CPU	Gflag: lazan_host_cpu_usage_threshold_fraction) we will migrate VMs off those host(s) to re-balance the workload. A key thing to mention here is a migration will only be performed when there is contention. If there is skew in utilization between nodes (e.g. 3 nodes at 10% and 1 at 50%) we will not perform a migration as there is no benefit from doing so until there is contention for resources.

- Storage performance
  -

- [Anti-]Affinity rules
    - Affinity or Anti-affinity constraints determine where certain resources are scheduled based upon other resources in the environment. In certain cases you want VMs to run on the same node for licensing reasons. In this case the VMs would be affined to the same host. In other cases you might want to ensure VMs run on different nodes for availability purposes. In this case the VMs would be anti-affined.

The scheduler will make its best effort to optimize workload placement based upon the prior items. The system places a penalty on movement to ensure not too many migrations are taking place. This is a key item as we want to make sure the movement doesn't have any negative impacts on the workload.

After a migration the system will judge its "effectiveness" and see what the actual benefit is. This learning model can self-optimize to ensure there is a valid basis for any migration decision.

Security

» Download this section as PDF (opens in a new tab/window)

Security is a core part of the Nutanix platform and was kept in mind from day one. The Nutanix Security Development Lifecycle (SecDL) incorporates security into every step of the development process. The Nutanix controlled parts of the platform is secure out of the box, rather than being an afterthought requiring end-users to "harden" the platform.

When we think about security we're really trying to achieve 3 core things (aptly named the CIA triad):

1. Confidentially
    - Protect and secure data by preventing unauthorized access
2. Integrity
    - Ensure the consistency and accuracy of data by preventing unauthorized alteration
3. Availability
    - Ensure authorized users get access to data through resiliency and redundancy

This can be simplified down to a simple statement: enable users to do their jobs while keeping the bad people out. When we're designing for security we need to look at a few core areas of interest which is highlighted in the following diagram:

Security Layers

We will break down each section in the prior graphic in the following sections.

Systems & Configuration

Traditionally people refer to system (OS + App) security using a method called “hardening”. This is the process to which you would secure the system by configuring things to a certain standard called a baseline.

The DoD’s IT org (DISA) has a sample hardening guide which they call the STIG (more details in the SCMA section following). This includes things like directory permissions, user account management, password complexity, firewalls and a slew of other configuration settings.

At a glance

- Patch and remove known vulnerabilities
- Enforce strong passwords and remove default accounts
- Configure permissions and user privileges
- Close unused ports / protocols
- Use automation to ensure baselines

Once a system is configured to that standard it is considered “secure” however that is just the beginning of the process. System security is something that must be maintained throughout its lifespan. For example, to ensure that standard hardening baseline is met, configuration automation tools should be employed. This ensures the system is always meeting your baseline “desired state”.

Nutanix ensures this for its CVM and AHV hypervisor using a tool we’ve developed called SCMA which is covered later in this section.

Data

Data is at the core of any business and is arguably the company’s most valuable asset. When thinking of security we need to focus on ensuring data accessibility, quality, and theft avoidance.

On the concept of accessibility, we constantly need access to systems and data to make decisions. One recent method of attack called ‘Ransomware’ threatens the ability to access data by encrypting the data and then ransoming the user to get access back. This can be avoided in a variety of methods, but also highlights to importance of backups.

At a glance:

- Secure access control to data
- Always take backups
- Encrypt data and secure keys

Data quality is also a critical item since a lot of decisions or actions are depending on this. For example, an attacker could get access to a system and place malicious orders or update shipping addresses diverting goods to his location. This is where logging and checksumming can be very critical to ensure data remains clean.

Last but not least is how do we secure or harden the data. This is typically done using encryption which renders the data useless if they don’t have the keys to decrypt the data. In this case if someone were to steal an encrypted file or disk device, they would be unable to get access to the underlying data.

Network

The network is the typically communication vector attackers use to gain access to systems. This includes things like perimeter security (e.g. external firewalls) and internal intrusion prevention / detection.

Like any good design there should always be layers of security; the same holds true with the network. We need to segment our high-security networks from our trusted networks and secure those from our untrusted networks (e.g. business / wifi networks). It is never safe to assume your local network in the office is secure.

At a glance:

- Segment trusted/untrusted networks
- Firewall at the perimeter and between segments
- Leverage an IDPS to detect anomalies

By having multiple layers of the network we can ensure someone who gains access our most untrusted network has a more difficult time working towards our secure networks. During this process a good IDPS system can detect access anomalies or scanning tools like nmap.

Authentication and Authorization

Authentication is all about authenticating a users identity against a trusted source of truth like Active Directory or any other IDP (Identity provider). Tools like MFA (multi-factor authentication) or 2FA add additional assurance the user is who they’re trying to authenticate as.

Once the identity has been verified the next piece is to determine what they are authorized to do or what they can access; this is the authorization piece. User foo is authorized to perform x,y on bar and y,z on bas.

At a glance:

- Use MFA/2FA where possible
- Use granular permissions

Compliance & Monitoring

Compliance is typically something people refer to when looking at certain accreditations like PCI, HIPAA, etc. However this extends further into ensure compliance with any hardening guide or standards that have been set. For example, the STIG is a sample hardening baseline, however each company may have additional policies / rules in place. In order to ensure a secure system, we must make sure our systems meet these policies and are in a compliant state.

At a glance:

- Compliance is a continuous activity
- Monitor and look for anomalies

Traditionally compliance is checked retroactively and is a fairly manual process. **I believe this is absolutely the wrong approach.** Compliance is something we must constantly ensure as that's the only way we can make sure we limit any potential threat vectors, or close any that may have been opened.

Tools that handle configuration management automation (aka desired state configuration - DSC) are a critical piece here. These will ensure our configuration / settings is always set to our baseline or desired state.

Monitoring and penetration testing are critical to validate and ensure this compliance. Tools like Nessus, Nmap or metasploit can be used to to test the security of a system. During these tests monitoring and detection systems should detect these and alert.

People

In any system, the people are traditionally the weakest link. In order to ensure users aren't prone to phishing attacks or social manipulation, training and education is critical. We must ensure that users know what to look for, and to escalate to a known resource if they are unsure.

At a glance:

- Educate, educate, educate
- Enforce strong practices and habits (e.g. locking computer)

One method of education is actually simulating phishing attacks so they can start to question things and learn what to look for. We must also enforce other policies like not leaving their computer unlocked or writing down their passwords.

Certifications & Accreditations

Nutanix has the following security certifications / qualifications across portions of the stack (on and off premise):

- Common Criteria*
  - Common Criteria was produced predominantly so that companies selling computer products for the government market (mainly for Defense or Intelligence use) would only need to have them evaluated against one set of standards. The CC was developed by the governments of Canada, France, Germany, the Netherlands, the UK, and the U.S.
  - *This is currently under re-certification as of March 2020
- Security Technical Implementation Guides (STIGs)
  - Configuration standards for DOD IA and IA-enabled devices/systems. Since 1998, DISA Field Security Operations (FSO) has played a critical role enhancing the security posture of DoD's (Dept of Defense) security systems by providing the Security Technical Implementation Guides. The STIGs contain technical guidance to "lock down" information systems/software that might otherwise be vulnerable to a malicious computer attack.
- FIPS 140-2
  - FIPS 140-2 standard is an information technology security accreditation program for cryptographic modules produced by private sector vendors who seek to have their products certified for use in government departments and regulated industries (such as financial and health-care institutions) that collect, store, transfer, share and disseminate sensitive but unclassified (SBU) information.
- NIST 800-53
- NIST 800-131a
- ISO 27001
- ISO 27017
- ISO 27018

Security Configuration Management Automation (SCMA)

Nutanix Security engineering now provides customers with the ability to evolve from point-in-time security baseline checking to a continuous monitoring/self-remediating baseline to ensure all CVM/AHV hosts in a cluster remain baseline compliant throughout the deployment lifecycle. This new innovation checks all components of the documented security baselines (STIGs) , and if found to be non-compliant, sets it back to the supported security settings without customer intervention. SCMA is enabled by default so no action is necessary to enable.

Ad-hoc SCMA execution

The SCMA will run on the configured schedule (Default: HOURLY), however it is also possible to run this on-demand. To run the SCMA tool you can execute the following command from the CVM(s):

Run on a single CVM

```
sudo salt-call state.highstate
```

Run on all CVMs

```
allssh "sudo salt-call state.highstate"
```

The Nutanix Command Line Interface (NCLI) allows customers to control various configuration settings to enable more stringent security requirements.

CVM Security Settings

The following commands have been added to NCLI to support cluster-wide configuration of the SCMA policy. The list below gives all commands and functions:

Get CVM security settings

```
ncli cluster get-cvm-security-config
```

This command outputs the current cluster configuration. The default output will display:

```
Enable Aide : false
Enable Core : false
Enable High Strength P... : false
Enable Banner : false
Enable SNMPv3 Only : false
Schedule : DAILY
```

Each of these is defined below:

- Aide
  - Enables the ‘Advanced Intrusion Detection Environment’ to periodically run.
- Core
  - Generates stack traces when there’s an issue or SCMA is unable to remediate.
- High Strength Passwords
  - Enforces high strength passwords (minlen=15,difok=8,remember=24)
- Banner
  - Enables a custom login banner
- SNMPv3 Only
  - Forces SNMPv3 instead of v2

Set CVM login banner

This command enables or disables the Department of Defense (DoD) knowledge of consent login banner when logging in to any Nutanix CVM.

```
ncli cluster edit-cvm-security-params enable-banner=[yes|no] #Default:no
```

Custom login banner

By default the DoD knowledge of consent login banner is used. To utilize a custom banner follow the following steps (run as the Nutanix user on any CVM):

1. Create backup of existing banner
   - sudo cp -a /srv/salt/security/KVM/sshd/DODbanner /srv/salt/security/KVM/sshd/DODbannerbak

2. Use vi to modify existing banner
   - sudo vi /srv/salt/security/KVM/sshd/DODbanner

2. Repeat steps on every CVM or SCP modified banner to all other CVMs
3. Enable banner using command above

Set CVM password strength

This command enables or disables high-strength password policies (minlen=15,difok=8,remember=24).

```
ncli cluster edit-cvm-security-params enable-high-strength-password=[yes|no] #Default:no
```

Set Advanced Intrusion Detection Environment (AIDE)

This command enables or disables the AIDE service to run weekly.

```
ncli cluster edit-cvm-security-params enable-aide=true=[yes|no] #Default:no
```

Set SNMPv3 only

This command enables or disables SNMPv3 only traps.

```
ncli cluster edit-cvm-security-params enable-snmpv3-only=[true|false] #Default:false
```

Set SCMA schedule

This command sets the frequency at which SCMA runs.

```
ncli cluster edit-cvm-security-params schedule=[HOURLY|DAILY|WEEKLY|MONTHLY] #Default:HOURLY
```

Hypervisor Security Settings

The following commands have been added to NCLI to support cluster-wide configuration of the SCMA policy. The list below gives all commands and functions:

Get hypervisor security settings

```
ncli cluster get-hypervisor-security-config
```

This command outputs the current cluster configuration. The default output will display:

```
Enable Aide : false
Enable Core : false
Enable High Strength P... : false
Enable Banner : false
Schedule : DAILY
```

Set hypervisor login banner

This command enables or disables the Department of Defense (DoD) knowledge of consent login banner when loging in to any Nutanix hypervisor.

```
ncli cluster edit-hypervisor-security-params enable-banner=[yes|no] #Default:no
```

Set hypervisor password strength

This command enables or disables high-strength password policies (minlen=15,difok=8,remember=24).

```
ncli cluster edit-hypervisor-security-params enable-high-strength-password=[yes|no] #Default:no
```

Set Advanced Intrusion Detection Environment (AIDE)

This command enables or disables the AIDE service to run weekly.

```
ncli cluster edit-hypervisor-security-params enable-aide=true=[yes|no] #Default:no
```

Set SCMA schedule

This command sets the frequency at which SCMA runs.

```
ncli cluster edit-hypervisor-security-params schedule=[HOURLY|DAILY|WEEKLY|MONTHLY] #Default:HOURLY
```

Cluster Lockdown

Cluster lockdown is the ability to disable password based CVM access and/or only allow key based access.

The cluster lockdown configuration can be found in Prism under the gear menu:

Cluster Lockdown Menu

This will show the current configuration and allow you to add/remove SSH keys for access:

Cluster Lockdown Page

To add a new key click on the ‘New Public Key’ button and enter the public key details:

Cluster Lockdown - Add Key

Working with SSH keys

To generate a SSH key, run the following command:

```
ssh-keygen -t rsa -b 2048
```

This will generate the key pair which creates two files:

- id_rsa (private key)
- id_rsa.pub (public key - this one is used when adding a key to the cluster)

Once you’ve added some key(s) and have validated access with them, you can disable password based login, by un-checking ‘Enable Remote Login with Password.’ A popup will appear to confirm the action, click ‘Ok’ to proceed with lockdown.

Data Encryption and Key Management

Data encryption is a method that allows parties to encode data in a manner that only those who are authorized can make sense of the data, making it unintelligible for anyone who is unauthorized.

For example, if a message needs to be sent to one person and one person only, the message (plaintext) can be encrypted with a cipher (key) and send them the encrypted message (ciphertext). If this message is stolen or intercepted the attacker can only see the ciphertext which is mostly useless without having the cipher to decipher the message. Once the desired party has received the message they can decrypt the message using the key we have given them.

There are a few main methods of encrypting data:

- Symmetric Encryption (private key encryption):
  - The same key is used to both encrypt and decrypt data
  - Examples: AES, PGP*, Blowfish, Twofish, etc.

- Asymmetric Encryption (public key encryption):
  - One key is used for encryption (public key), another is used for decryption (private key)
  - Examples: RSA, PGP*, etc.

NOTE: PGP (or GPG) uses both a symmetric and asymmetric key.

When data encryption is talked about it is normally done so in two main contexts:

- In-transit: data that is in transit between two parties (e.g. sending data over the network)
- At-rest: static data (e.g. data that is stored on a device)

With Native software-based encryption (with or without SEDs) Nutanix solves for both in-transit* and at-rest encryption. With SED only based encryption Nutanix solves for at-rest data encryption. *NOTE: In-transit encryption is currently applicable within a Nutanix cluster for data RF.

The following sections will describe how Nutanix manages data encryption and its key management options.

Data Encryption

Nutanix provides data encryption via three main options:

- Native software-based encryption (FIPS-140-2 Level-1) *released in 5.5
- Using self-encrypting drives (SED) (FIPS-140-2 Level-2)
- Software + hardware encryption

This encryption is configured at either the cluster or container level, and is dependent on the hypervisor type:

- Cluster level encryption:
  - AHV, ESXi, Hyper-V

- Container level encryption:
  - ESXi, Hyper-V

NOTE: for deployments using SED based encryption, this will be cluster level as the physical devices are encrypted themselves.

You can view the encryption state of the cluster by navigating to ‘Data-at-Rest Encryption’ in the settings menu (gear icon). This will provide the current status and allow you to configure encryption (if not currently enabled).

In this example we can see that encryption is enabled at the cluster level:

Data Encryption - Enabled (cluster level)

In this example encryption is enabled for particular containers which are listed:

Data Encryption - Enabled (container level)

You can enable / modify the configuration by clicking the ‘edit configuration’ button. This will bring up the menu to configure the KMS used for encryption or the type of KMS currently being leveraged:

Data Encryption - Configure

For external KMS the menus will guide your through the CSR request process which you can then give to your CA for signing.

Native Software-based Encryption

Nutanix software encryption provides native AES-256 data-at-rest encryption. This can either interact with any KMIP or TCG compliant external KMS server (Vormetric, SafeNet, etc.) or the Nutanix native KMS introduced in 5.8 (more on this below). For encryption / decryption the system leverages the Intel AES-NI acceleration to minimize any potential performance impact of doing this in software.

As data is written (OpLog and Extent Store) the data is encrypted before it is written to disk at the checksum boundary. This also means that data is encrypted locally and then the encrypted data is replicated to the remote CVM(s) for RF.

Encryption is the last transform applied to data before it is written to disk:

Data Encryption - Transform Application

Encryption and Data Efficiency

Since we encrypt the data after we've applied any deduplication or compression, we ensure that all space savings from those methods are maintained. Put simply, deduplication and compression ratios will be the exact same for encrypted or non-encrypted data.

When data is read we will read the encrypted data from disk at the checksum boundary, decrypt and return the data to the guest. By doing [de/en]cryption at the checksum boundary we ensure no read amplification occurs. Given we are leveraging the Intel AES NI offload, we’ve seen very little impact to performance / latency.

SED Based Encryption

The figure shows a high-level overview of the architecture:

Data Encryption - SED

SED encryption works by splitting the storage device into “data bands” which can be in an secured or un-secured state. In the case of Nutanix, the boot and Nutanix Home partitions are trivially encrypted. All data devices and bands are heavily encrypted with big keys to level-2 standards.

When the cluster starts it will call out to the KMS server to get the keys to unlock the drives. In order to ensure security no keys are cached on the cluster. In the event of a cold boot and IPMI reset, the node will need to make a call back to the KMS server to unlock the drives. Soft reboots of the CVM will not force this to occur.

Key Management (KMS)

Nutanix provides native key management (local key manager - LKM) and storage capabilities (introduced in 5.8) as an alternative to other dedicated KMS solutions. This was introduced to negate the need for a dedicated KMS solution and simplify the environment, however external KMS are still supported.

As mentioned in the prior section, key management is a very crucial piece of any data encryption solution. Multiple keys are used throughout the stack to provide a very secure key management solution.

There are three types of keys used in the solution:

- Data Encryption Key (DEK)
  - Key used to encrypt the data

- Key Encryption Key (KEK)
  - Encryption key used to encrypt the DEK

- Master Encryption Key (MEK)
  - Encryption key used to encrypt the KEK
  - Only applicable when using the Local Key Manager

The following figure shows the relationships between the various keys and KMS options:

Data Encryption - Key Management

The local key manager (LKM) service is distributed among every Nutanix node and runs natively on each CVM. The service uses a FIPS 140-2 Crypto module (under certification), and key management is transparent to the end-user besides doing any key management activities (e.g. re-key, backup keys, etc.).

When configuring data encryption, the native KMS can be leveraged by selecting ‘Cluster’s local KMS’:

Data Encryption - Configure

The master key (MEK) is split and stored across all nodes in the cluster leveraging Shamir’s Secret Sharing algorithm to allow for resiliency and security. A minimum of ROUNDUP(N/2) nodes must be available to re-construct the keys, where N = number of nodes in the cluster.

Key Backups and Key Rotation

Once encryption has been enabled, it is recommended to take a backup of the data encryption key(s) (DEK). If a backup is taken, it must be secured with a strong password and stored in a secure location.

The system provides the ability to rotate (re-key) both the KEK and MEK. It automatically rotates the master key (MEK) every year, however, this operation can also be done on demand. In the event of a node add/remove, we also rotate the master key.

Storage

» Download this section as PDF (opens in a new tab/window)

Nutanix AOS storage is the scale-out storage technology that appears to the hypervisor like any centralized storage array, however all of the I/Os are handled locally to provide the highest performance. More detail on how these nodes form a distributed system can be found in the next section.

Data Structure

Nutanix AOS storage is composed of the following high-level struct:

Storage Pool

- Key Role: Group of physical devices
- Description: A storage pool is a group of physical storage devices including PCIe SSD, SSD, and HDD devices for the cluster. The storage pool can span multiple Nutanix nodes and is expanded as the cluster scales. In most configurations, only a single storage pool is leveraged.

Container

- Key Role: Group of VMs/files
- Description: A container is a logical segmentation of the Storage Pool and contains a group of VM or files (vDisks). Some configuration options (e.g., RF) are configured at the container level, however are applied at the individual VM/file level. Containers typically have a 1 to 1 mapping with a datastore (in the case of NFS/SMB).

vDisk

- Key Role: vDisk
- Description: A vDisk is any file over 512KB on AOS including .vmdks and VM hard disks. vDisks are logically composed of vBlocks which make up the 'block map.'

Maximum AOS vDisk Size

No artificial limits are imposed on the vdisk size on the AOS/stargate side. As of 4.6, the vdisk size is stored as a 64 bit signed integer that stores the size in bytes. This means the theoretical maximum vDisk size can be 2^63-1 or 9E18 (9 Exabytes). Any limits below this value would be due to limitations on the client side, such as the maximum vmdk size on ESXi.

The following figure shows how these map between AOS and the hypervisor:

High-level Filesystem Breakdown

vBlock

- Key Role: 1MB chunk of vDisk address space
- Description: A vBlock is a 1MB chunk of virtual address space composing a vDisk. For example, a vDisk of 100MB will have 100 x 1MB vBlocks, vBlock 0 would be for 0-1MB, vBlock 1 would be from 1-2MB, and so forth. These vBlocks map to extents which are stored as files on disk as extent groups.

Extent

- Key Role: Logically contiguous data
- Description: An extent is a 1MB piece of logically contiguous data which consists of n number of contiguous blocks (varies depending on guest OS block size). Extents are written/read/modified on a sub-extent basis (aka slice) for granularity and efficiency. An extent’s slice may be trimmed when moving into the cache depending on the amount of data being read/cached.

Extent Group

- Key Role: Physically contiguous stored data
- Description: An extent group is a 1MB or 4MB piece of physically contiguous stored data. This data is stored as a file on the storage device owned by the CVM. Extents are dynamically distributed among extent groups to provide data striping across nodes/disks to improve performance. NOTE: as of 4.0, extent groups can now be either 1MB or 4MB depending on deduplication.

The following figure shows how these structs relate between the various file systems:

Low-level Filesystem Breakdown

Here is another graphical representation of how these units are related:

Graphical Filesystem Breakdown

I/O Path and Cache

For a visual explanation, you can watch the following video: LINK

The typical hyperconverged storage I/O path can be characterized into the following core layers:

1. Guest OS (UVM) to virtual disk(s)
   - This remains unchanged with Nutanix. Depending on the hypervisor the guest OS will use a device driver to talk to a virtual disk device. Depending on the hypervisor this could be virtio-scsi (AHV), pv-scsi (ESXi), etc. The virtual disks will also vary based upon the hypervisor (e.g. vmdk, vhd, etc.)

2. Hypervisor to AOS (via CVM)
   - Communication between the hypervisor and Nutanix occurs via standard storage protocols (e.g. iSCSI, NFS, SMBv3) over the local interface of the CVM and hypervisor. At this point all communication has been local to the host (there are scenarios where I/O will be remote (e.g. local CVM down, etc.).

3. Nutanix I/O path
   - This is all transparent to the hypervisor and UVMs and it native to the Nutanix platform.

The following image shows a high-level overview of these layers:

High-level I/O Path - Traditional

Communication and I/O

Within the CVM the Stargate process is responsible for handling all storage I/O requests and interaction with other CVMs / physical devices. Storage device controllers are passed through directly to the CVM so all storage I/O bypasses the hypervisor.

The following image shows a high-level overview of the traditional I/O path:

High-level I/O Path

Nutanix BlockStore (shipped in AOS 5.18) is an AOS capability which creates an extensible filesystem and block management layer all handled in user space. This eliminates the filesystem from the devices and removes the invoking of any filesystem kernel driver. The introduction of newer storage media (e.g. NVMe), devices now come with user space libraries to handle device I/O directly (e.g. SPDK) eliminating the need to make any system calls (context switches). With the combination of BlockStore + SPDK all Stargate device interaction has moved into user space eliminating any context switching or kernel driver invocation.

Stargate - Device I/O Path

The following image shows a high-level overview of the updated I/O path with BlockStore + SPDK:

High-level I/O Path - BlockStore

To perform data replication the CVMs communicate over the network. With the default stack this will invoke kernel level drivers to do so.

However, with RDMA these NICs are passed through to the CVM, bypassing the hypervisor and reducing interrupts. Also, within the CVM all network traffic using RDMA only uses a kernel level driver for the control path, then all actual data I/O is done in user-space without any context switches.

The following image shows a high-level overview of the I/O path with RDMA:

High-level I/O Path - RDMA

To summarize, the following enhancements optimize with the following:

1. PCI passthrough bypasses the hypervisor for device I/O
2. SPDK + Blockstore eliminates kernel storage driver interactions and moves them to user-space
3. RDMA bypasses the hypervisor and all data transfer is done in CVM user-space

Stargate I/O Logic

Within the CVM the Stargate process is responsible for handling all I/O coming from user VMs (UVMs) and persistence (RF, etc.). When a write request comes to Stargate, there is a write characterizer which will determine if the write gets persisted to the OpLog for bursty random writes, or to Extent Store for sustained random and sequential writes. Read requests are satisfied from Oplog or Extent Store depending on where the data is residing when it is requested.

The Nutanix I/O path is composed of the following high-level components:

AOS I/O Path

*As of AOS 5.10, the Autonomous Extent Store (AES) can be used to handle sustained random workloads when requisite conditions are met.

^:

1. In all-flash node configurations (All NVMe SSD, All SATA/SAS SSD, NVMe+SATA/SAS SSD) the Extent Store will only consist of SSD devices and no tier ILM will occur as only a single flash tier exists.
2. In cases where Optane is used (e.g. Intel Optane + NVMe/SATA SSD) the highest performance media will be Tier 0 and the lower performance media will be Tier 1.
3. For hybrid scenarios with flash and HDD, the flash would be Tier 0 with HDD being Tier 1.
4. OpLog is always on SSDs in Hybrid Clusters and on both Optane and NVMe SSDs in Optane+NVMe clusters.
5. Data is moved between tiers by Intelligent Lifecycle Management (ILM) based on access patterns.

OpLog

- Key Role: Persistent write buffer
- Description: The OpLog is similar to a filesystem journal and is built as a staging area to handle bursts of random writes, coalesce them, and then sequentially drain the data to the extent store. Upon a write, the OpLog is synchronously replicated to another n number of CVM's OpLog before the write is acknowledged for data availability purposes. All CVM OpLogs partake in the replication and are dynamically chosen based upon load. The OpLog is stored on the SSD tier on the CVM to provide extremely fast write I/O performance, especially for random I/O workloads. All SSD devices participate and handle a portion of OpLog storage. For sequential workloads, the OpLog is bypassed and the writes go directly to the extent store. If data is currently sitting in the OpLog and has not been drained, all read requests will be directly fulfilled from the OpLog until they have been drained, where they would then be served by the extent store/unified cache. For containers where fingerprinting (aka Dedupe) has been enabled, all write I/Os will be fingerprinted using a hashing scheme allowing them to be deduplicated based upon fingerprint in the unified cache.

Dynamic OpLog Sizing

The OpLog is a shared resource and, however allocation is done on a per-vDisk basis to ensure each vDisk has an equal opportunity to leverage. Prior to AOS 5.19, Oplog size was capped to 6GB per vdisk. Starting with AOS 5.19, OpLog for individual vdisks can keep growing beyond 6GB if required based on IO patterns until a cap is reached for OpLog index memory used per node. This design decision allows flexibility such that if there are VMs with fewer vdisks that are more active from an I/O perspective, they can keep growing their OpLog as required at the expense of other vdisks that are not as active.

Extent Store

- Key Role: Persistent data storage
- Description: The Extent Store is the persistent bulk storage of AOS and spans all device tiers (Optane SSD, PCIe SSD, SATA SSD, HDD) and is extensible to facilitate additional devices/tiers. Data entering the extent store is either being A) drained from the OpLog or B) is sequential/sustained in nature and has bypassed the OpLog directly. Nutanix ILM will determine tier placement dynamically based upon I/O patterns, number of accesses of data and weight given to individual tiers and will move data between tiers.

Autonomous Extent Store (AES)

- Key Role: Persistent data storage
- Description: The Autonomous Extent Store (AES) is a new method for writing / storing data in the Extent Store introduced in AOS 5.10. It leverages a mix of primarily local + global metadata (more detail in the ‘Scalable Metadata’ section following) allowing for much more efficient sustained performance due to metadata locality. For sustained random write workloads, these will bypass the OpLog and be written directly to the Extent Store using AES. For bursty random workloads these will take the typical OpLog I/O path then drain to the Extent Store using AES where possible. As of AOS 5.20 (LTS), AES is enabled by default for new containers on All Flash Clusters and as of AOS 6.1 (STS) if requirements are met, AES is enabled on new containers created on Hybrid(SSD+HDD) Clusters.

Unified Cache

- Key Role: Dynamic read cache
- Description: The Unified Cache is a read cache which is used for data, metadata and deduplication and stored in the CVM’s memory. Upon a read request of data not in the cache (or based upon a particular fingerprint), the data will be read from the extent store and will also be placed into the single-touch pool of the Unified Cache which completely sits in memory, where it will use LRU (least recently used) until it is evicted from the cache. Any subsequent read request will “move” (no data is actually moved, just cache metadata) the data into the multi-touch pool. Any read request for data in the multi-touch pool will cause the data to go to the peak of the multi-touch pool where it will be given a new LRU counter. Cache size can be calculated using the following formula: ((CVM Memory - 12 GB) * 0.45). For example a 32GB CVM would have the following cache size: ((32 - 12)*0.45) == 9GB.

The following figure shows a high-level overview of the Unified Cache:

AOS Unified Cache

Cache Granularity and Logic

Data is brought into the cache at a 4K granularity and all caching is done real-time (e.g. no delay or batch process data to pull data into the cache).

Each CVM has its own local cache that it manages for the vDisk(s) it is hosting (e.g. VM(s) running on the same node). When a vDisk is cloned (e.g. new clones, snapshots, etc.) each new vDisk has its own block map and the original vDisk is marked as immutable. This allows us to ensure that each CVM can have it's own cached copy of the base vDisk with cache coherency.

Single vDisk Sharding

AOS was designed and architected to deliver performance for applications at scale. Inside Stargate, I/O is processed by threads for every vdisk created by something called vdisk controller. Every vdisk gets its own vdisk controller inside Stargate responsible for I/O for that vdisk. The expectation was that workloads and applications would have multiple vdisks each having its own vdisk controller thread capable of driving high performance the system is capable of delivering.

This architecture worked well except in cases of traditional applications and workloads that had VMs with single large vdisk. These VMs were not able to leverage the capabilities of AOS to its fullest. As of AOS 6.1, enhancement was done to vdisk controller such that requests to a single vdisk are now distributed to multiple vdisk controllers by creating shards of the controller each having its own thread. I/O distribution to multiple controllers is done by a primary controller so for external interaction this still looks like a single vdisk. This results in effectively sharding the single vdisk making it multi-threaded. This enhancement alongwith other technologies talked above like Blockstore, AES allows AOS to deliver consistent high performance at scale even for traditional applications that use a single vdisk.

Scalable Metadata

For a visual explanation, you can watch the following YouTube video: Tech TopX by Nutanix University: Scalable Metadata

Metadata is at the core of any intelligent system and is even more critical for any filesystem or storage array. For those unsure about the term 'metadata'; essentially metadata is 'data about data'. In terms of AOS, there are a few key principles that are critical for its success:

- Must be right 100% of the time (known as "strictly consistent")
- Must be ACID compliant
- Must have unlimited scalability
- Must not have any bottlenecks at any scale (must be linearly scalable)

As of AOS 5.10 metadata is broken into two areas: global vs. local metadata (prior all metadata was global). The motivation for this is to optimize for "metadata locality" and limit the network traffic on the system for metadata lookups.

The basis for this change is that not all data needs to be global. For example, every CVM doesn't need to know which physical disk a particular extent sits on, they just need to know which node holds that data, and only that node needs to know which disk has the data.

By doing this we can limit the amount of metadata stored by the system (eliminate metadata RF for local only data), and optimize for “metadata locality.”

The following image shows the differentiation between global vs. local metadata:

Global vs. Local Metadata

Local Metadata

- Description:
  - Local metadata store per CVM containing information only needed by the local node. This is leveraged by the Autonomous Extent Store (AES) introduced in 5.10.
- Storage Mechanism:
  - AES DB (based on Rocksdb)
- Types of data stored:
  - Physical extent / extent group placement (e.g. egroup to disk mappings), etc.

Global Metadata

- Description:
  - Metadata that is globally available to any CVM and sharded across CVMs in the cluster. All metadata prior to 5.10.
- Storage Mechanism:
  - Medusa Store (based on Cassandra)
- Types of data stored:
  - vDisk block maps, extent to node mappings, time series stats, configurations, etc.

The section below covers how global metadata is managed:

As mentioned in the architecture section above, AOS utilizes a “ring-like” structure as a key-value store which stores essential global metadata as well as other platform data (e.g., stats, etc.). In order to ensure global metadata availability and redundancy a replication factor (RF) is utilized among an odd amount of nodes (e.g., 3, 5, etc.). Upon a global metadata write or update, the row is written to a node in the ring that owns that key and then replicated to n number of peers (where n is dependent on cluster size). A majority of nodes must agree before anything is committed, which is enforced using the Paxos algorithm. This ensures strict consistency for all data and global metadata stored as part of the platform.

The following figure shows an example of a global metadata insert/update for a 4 node cluster:

Cassandra Ring Structure

Performance at scale is also another important struct for AOS global metadata. Contrary to traditional dual-controller or “leader/worker” models, each Nutanix node is responsible for a subset of the overall platform’s metadata. This eliminates the traditional bottlenecks by allowing global metadata to be served and manipulated by all nodes in the cluster. A consistent hashing scheme is utilized for key partitioning to minimize the redistribution of keys during cluster size modifications (also known as “add/remove node”). When the cluster scales (e.g., from 4 to 8 nodes), the nodes are inserted throughout the ring between nodes for “block awareness” and reliability.

The following figure shows an example of the global metadata “ring” and how it scales:

Cassandra Scale Out

Data Protection

For a visual explanation, you can watch the following video: LINK

The Nutanix platform currently uses a resiliency factor, also known as a replication factor (RF), and checksum to ensure data redundancy and availability in the case of a node or disk failure or corruption. As explained above, the OpLog acts as a staging area to absorb incoming writes onto a low-latency SSD tier. Upon being written to the local OpLog, the data is synchronously replicated to another one or two Nutanix CVM’s OpLog (dependent on RF) before being acknowledged (Ack) as a successful write to the host. This ensures that the data exists in at least two or three independent locations and is fault tolerant. NOTE: For RF3, a minimum of 5 nodes is required since metadata will be RF5.

OpLog peers are chosen for every episode (1GB of vDisk data) and all nodes actively participate. Multiple factors play into which peers are chosen (e.g. response time, business, capacity utilization, etc). This eliminates any fragmentation and ensures every CVM/OpLog can be used concurrently.

Data RF is configured via Prism and is done at the container level. All nodes participate in OpLog replication to eliminate any “hot nodes”, ensuring linear performance at scale. While the data is being written, a checksum is computed and stored as part of its metadata. Data is then asynchronously drained to the extent store where the RF is implicitly maintained. In the case of a node or disk failure, the data is then re-

replicated among all nodes in the cluster to maintain the RF. Any time the data is read, the checksum is computed to ensure the data is valid. In the event where the checksum and data don't match, the replica of the data will be read and will replace the non-valid copy.

Data is also consistently monitored to ensure integrity even when active I/O isn't occurring. Stargate's scrubber operation will consistently scan through extent groups and perform checksum validation when disks aren't heavily utilized. This protects against things like bit rot or corrupted sectors.

The following figure shows an example of what this logically looks like:

AOS Data Protection

Availability Domains

For a visual explanation, you can watch the following video: LINK

Availability Domains (aka node/block/rack awareness) is a key struct for distributed systems to abide by for determining component and data placement. Nutanix refers to a "block" as the chassis which contains either one, two, or four server "nodes" and a "rack" as a physical unit containing one or more "block". NOTE: A minimum of 3 blocks must be utilized for block awareness to be activated, otherwise node awareness will be used.

Nutanix currently supports the following levels or awareness:

- Disk (always)
- Node (always)
- Block (as of AOS 4.5)
- Rack (as of AOS 5.9)

It is recommended to utilize uniformly populated blocks / racks to ensure the awareness is enabled and no imbalance is possible. Common scenarios and the awareness level utilized can be found at the bottom of this section. The 3-block requirement is due to ensure quorum. For example, a 3450 would be a block which holds 4 nodes. The reason for distributing roles or data across blocks is to ensure if a block fails or needs maintenance the system can continue to run without interruption. NOTE: Within a block, the redundant PSU and fans are the only shared components.

NOTE: Rack awareness requires the administrator to define "racks" in which the blocks are placed.

The following shows how this is configured in Prism:

Rack Configuration

Awareness can be broken into a few key focus areas:

- Data (The VM data)
- Metadata (Cassandra)
- Configuration Data (Zookeeper)

Data

With AOS, data replicas will be written to other [blocks/racks] in the cluster to ensure that in the case of a [block/rack] failure or planned downtime, the data remains available. This is true for both RF2 and RF3 scenarios, as well as in the case of a [block/rack] failure. An easy comparison would be “node awareness”, where a replica would need to be replicated to another node which will provide protection in the case of a node failure. Block and rack awareness further enhances this by providing data availability assurances in the case of [block/rack] outages.

The following figure shows how the replica placement would work in a 3-block deployment:

Block/Rack Aware Replica Placement

In the case of a [block/rack] failure, [block/rack] awareness will be maintained (if possible) and the data will be replicated to other [blocks/racks] within the cluster:

Block/Rack Failure Replica Placement

Rack/Block Awareness vs. Metro clustering

A common question is can you span a cluster across two locations (rooms, buildings, etc.) and use block / rack awareness to provide resiliency around a location failure.

While theoretically possible this is not the recommended approach. Let's first think about what we're trying to achieve with this:

1. Low RPO
2. Low RTO (HA event instead of a DR event)

If we take the first case where we're trying to achieve an RPO ~0, it is preferred to leverage synchronous or near-synchronous replication. This will provide the same RPOs with less risk.

To minimize the RTO one can leverage a metro-cluster on top of synchronous replication and handle any failures as HA events instead of doing DR recoveries.

In summary it is preferred to leverage synchronous replication / metro clustering for the following reasons:

- The same end result can be achieved with sync rep / metro clustering, avoiding any risks and keeping isolated fault domains
- If network connectivity goes down between the two locations in a non-supported "stretched" deployment, one side will go down as quorum must be maintained (e.g. majority side will stay up). In the metro cluster scenario, both sides can continue to operate independently.
- Availability domain placement of data is best effort in skewed scenarios
- Additional Latency / reduced network bandwidth between both sites can impact performance in the "stretched" deployment

Awareness Conditions and Tolerance

Below we breakdown some common scenarios and the level of tolerance:

Desired Awareness Type	FT Level	EC Enabled?	Min. Units	Simultaneous failure tolerance
Node	1	No	3 Nodes	1 Node
Node	1	Yes	4 Nodes	1 Node
Node	2	No	5 Nodes	2 Node
Node	2	Yes	6 Nodes	2 Nodes
Block	1	No	3 Blocks	1 Block
Block	1	Yes	4 Blocks	1 Block
Block	2	No	5 Blocks	2 Blocks
Block	2	Yes	6 Blocks	2 Blocks
Rack	1	No	3 Racks	1 Rack
Rack	1	Yes	4 Racks	1 Rack

Rack	2	No	5 Racks	2 Racks
Rack	2	Yes	6 Racks	2 Racks

As of AOS base software version 4.5 and later block awareness is best effort and doesn't have strict requirements for enabling. This was done to ensure clusters with skewed storage resources (e.g. storage heavy nodes) don't disable the feature. With that stated, it is however still a best practice to have uniform blocks to minimize any storage skew.

Prior to 4.5 the following conditions must be met for block awareness:

- If SSD**or** HDD tier variance between blocks is > max variance:**NODE** awareness
- If SSD and HDD tier variance between blocks is < max variance:**BLOCK + NODE** awareness

Max tier variance is calculated as: 100 / (RF+1)

- E.g., 33% for RF2 or 25% for RF3

Metadata

As mentioned in the Scalable Metadata section above, Nutanix leverages a heavily modified Cassandra platform to store metadata and other essential information. Cassandra leverages a ring-like structure and replicates to n number of peers within the ring to ensure data consistency and availability.

The following figure shows an example of the Cassandra's ring for a 12-node cluster:

12 Node Cassandra Ring

Cassandra peer replication iterates through nodes in a clockwise manner throughout the ring. With [block/rack] awareness, the peers are distributed among the [blocks/racks] to ensure no two peers are on the same [block/rack].

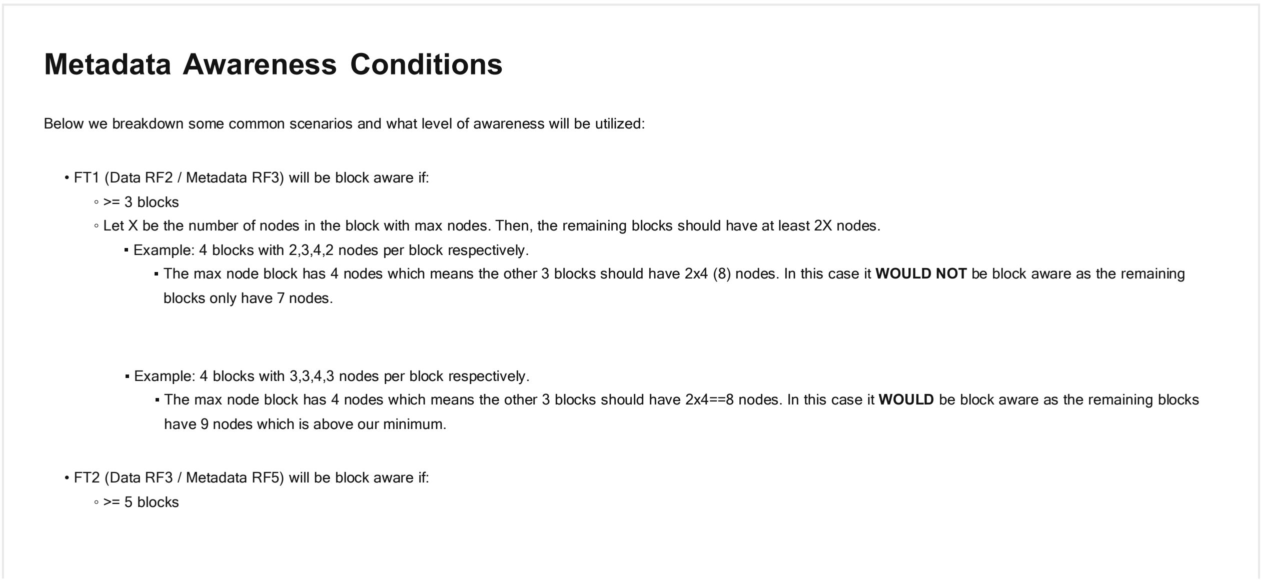
The following figure shows an example node layout translating the ring above into the [block/rack] based layout:

Cassandra Node Block/Rack Aware Placement

With this [block/rack]-aware nature, in the event of a [block/rack] failure there will still be at least two copies of the data (with Metadata RF3 – In larger clusters RF5 can be leveraged).

The following figure shows an example of all of the nodes replication topology to form the ring (yes – it's a little busy):

Full Cassandra Node Block/Rack Aware Placement



Metadata Awareness Conditions

Below we breakdown some common scenarios and what level of awareness will be utilized:

- FT1 (Data RF2 / Metadata RF3) will be block aware if:
  - >= 3 blocks
  - Let X be the number of nodes in the block with max nodes. Then, the remaining blocks should have at least 2X nodes.
    - Example: 4 blocks with 2,3,4,2 nodes per block respectively.
      - The max node block has 4 nodes which means the other 3 blocks should have 2x4 (8) nodes. In this case it **WOULD NOT** be block aware as the remaining blocks only have 7 nodes.
    - Example: 4 blocks with 3,3,4,3 nodes per block respectively.
      - The max node block has 4 nodes which means the other 3 blocks should have 2x4==8 nodes. In this case it **WOULD** be block aware as the remaining blocks have 9 nodes which is above our minimum.
- FT2 (Data RF3 / Metadata RF5) will be block aware if:
  - >= 5 blocks

Configuration Data

Nutanix leverages Zookeeper to store essential configuration data for the cluster. This role is also distributed in a [block/rack]-aware manner to ensure availability in the case of a [block/rack] failure.

The following figure shows an example layout showing 3 Zookeeper nodes distributed in a [block/rack]-aware manner:

Zookeeper Block/Rack Aware Placement

In the event of a [block/rack] outage, meaning one of the Zookeeper nodes will be gone, the Zookeeper role would be transferred to another node in the cluster as shown below:

Zookeeper Placement Block/Rack Failure

When the [block/rack] comes back online, the Zookeeper role would be transferred back to maintain [block/rack] awareness.

NOTE: Prior to 4.5, this migration was not automatic and must be done manually.

Data Path Resiliency

Reliability and resiliency are key, if not the most important concepts within AOS or any primary storage platform.

Contrary to traditional architectures which are built around the idea that hardware will be reliable, Nutanix takes a different approach: it expects hardware will eventually fail. By doing so, the system is designed to handle these failures in an elegant and non-disruptive manner.

NOTE: That doesn't mean the hardware quality isn't there, just a concept shift. The Nutanix hardware and QA teams undergo an exhaustive qualification and vetting process.

As mentioned in the prior sections metadata and data are protected using a RF which is based upon the cluster FT level. As of 5.0 supported FT levels are FT1 and FT2 which correspond to metadata RF3 and data RF2, or metadata RF5 and data RF3 respectively.

To learn more about how metadata is sharded refer to the prior 'Scalable Metadata' section. To learn more about how data is protected refer to the prior 'Data protection' section.

In a normal state, cluster data layout will look similar to the following:

Data Path Resiliency - Normal State

As you can see the VM/vDisk data has 2 or 3 copies on disk which are distributed among the nodes and associated storage devices.

Importance of Data Distribution

By ensuring metadata and data is distributed across all nodes and all disk devices we can ensure the highest possible performance during normal data ingest and re-protection.

As data is ingested into the system its primary and replica copies will be distributed across the local and all other remote nodes. By doing so we can eliminate any potential hot spots (e.g. a node or disk performing slowly) and ensure a consistent write performance.

In the event of a disk or node failure where data must be re-protected, the full power of the cluster can be used for the rebuild. In this event the scan of metadata (to find out the data on the failed device(s) and where the replicas exist) will be distributed evenly across all CVMs. Once the data replicas have been found all healthy CVMs, disk devices (SSD+HDD), and host network uplinks can be used concurrently to rebuild the data.

For example, in a 4 node cluster where a disk fails each CVM will handle 25% of the metadata scan and data rebuild. In a 10 node cluster, each CVM will handle 10% of the metadata scan and data rebuild. In a 50 node cluster, each CVM will handle 2% of the metadata scan and data rebuild.

Key point: With Nutanix and by ensuring uniform distribution of data we can ensure consistent write performance and far superior re-protection times. This also applies to any cluster wide activity (e.g. erasure coding, compression, deduplication, etc.)

Comparing this to other solutions where HA pairs are used or a single disk holds a full copy of the data, they will face frontend performance issues if the mirrored node/disk is under duress (facing heavy IO or resource constraints).

Also, in the event of a failure where data must be re-protected, they will be limited by a single controller, a single node's disk resources and a single node's network uplinks. When terabytes of data must be re-replicated this will be severely constrained by the local node's disk and network bandwidth, increasing the time the system is in a potential data loss state if another failure occurs.

Potential levels of failure

Being a distributed system, AOS is built to handle component, service, and CVM failures, which can be characterized on a few levels:

- Disk Failure

- CVM “Failure”
- Node Failure

When does a rebuild begin?

When there is an unplanned failure (in some cases we will proactively take things offline if they aren't working correctly) we begin the rebuild process **immediately**.

Unlike some other vendors which wait 60 minutes to start rebuilding and only maintain a single copy during that period (very risky and can lead to data loss if there's any sort of failure), we are not willing to take that risk at the sacrifice of potentially higher storage utilization.

We can do this because of a) the granularity of our metadata b) choose peers for write RF dynamically (while there is a failure, all new data (e.g. new writes / overwrites) maintain their configured redundancy) and c) we can handle things coming back online during a rebuild and re-admit the data once it has been validated. In this scenario data may be "over-replicated" in which a Curator scan will kick off and remove the over-replicated copies.

Disk Failure

A disk failure can be characterized as just that, a disk which has either been removed, encounters a failure, or one that is not responding or has I/O errors. When Stargate sees I/O errors or the device fails to respond within a certain threshold it will mark the disk offline. Once that has occurred Hades will run S.M.A.R.T. and check the status of the device. If the tests pass the disk will be marked online, if they fail it will remain offline. If Stargate marks a disk offline multiple times (currently 3 times in an hour), Hades will stop marking the disk online even if S.M.A.R.T. tests pass.

VM impact:

- HA event:**No**
- Failed I/Os:**No**
- Latency:**No impact**

In the event of a disk failure, a Curator scan (MapReduce Framework) will occur immediately. It will scan the metadata (Cassandra) to find the data previously hosted on the failed disk and the nodes / disks hosting the replicas.

Once it has found that data that needs to be “re-replicated”, it will distribute the replication tasks to the nodes throughout the cluster.

During this process a Drive Self Test (DST) is started for the bad disk and SMART logs are monitored for errors.

The following figure shows an example disk failure and re-protection:

Data Path Resiliency - Disk Failure

An important thing to highlight here is given how Nutanix distributes data and replicas across all nodes / CVMs / disks; all nodes / CVMs / disks will participate in the re-replication.

This substantially reduces the time required for re-protection, as the power of the full cluster can be utilized; the larger the cluster, the faster the re-protection.

Node Failure

VM Impact:

- HA event:**Yes**
- Failed I/Os:**No**
- Latency:**No impact**

In the event of a node failure, a VM HA event will occur restarting the VMs on other nodes throughout the virtualization cluster. Once restarted, the VMs will continue to perform I/Os as usual which will be handled by their local CVMs.

Similar to the case of a disk failure above, a Curator scan will find the data previously hosted on the node and its respective replicas. Once the replicas are found all nodes will participate in the reprotection.

Data Path Resiliency - Node Failure

In the event where the node remains down for a prolonged period of time (30 minutes as of 4.6), the down CVM will be removed from the metadata ring. It will be joined back into the ring after it has been up and stable for a duration of time.

Pro tip

Data resiliency state will be shown in Prism on the dashboard page.

You can also check data resiliency state via the cli:

Node Status

```
ncli cluster get-domain-fault-tolerance-status type=node
```

Block Status

```
ncli cluster get-domain-fault-tolerance-status type=rackable_unit
```

These should always be up to date, however to refresh the data you can kick off a Curator partial scan.

CVM “Failure”

A CVM “failure” can be characterized as a CVM power action causing the CVM to be temporarily unavailable. The system is designed to transparently handle these gracefully. In the event of a failure, I/Os will be re-directed to other CVMs within the cluster. The mechanism for this will vary by hypervisor.

The rolling upgrade process actually leverages this capability as it will upgrade one CVM at a time, iterating through the cluster.

VM impact:

- HA event:**No**
- Failed I/Os:**No**

- Latency:**Potentially higher given I/Os over the network**

In the event of a CVM “failure” the I/O which was previously being served from the down CVM, will be forwarded to other CVMs throughout the cluster. ESXi and Hyper-V handle this via a process called CVM Autopathing, which leverages HA.py (like “happy”), where it will modify the routes to forward traffic going to the internal address (192.168.5.2) to the external IP of other CVMs throughout the cluster. This enables the datastore to remain intact, just the CVM responsible for serving the I/Os is remote.

Once the local CVM comes back up and is stable, the route would be removed and the local CVM would take over all new I/Os.

In the case of AHV, iSCSI multi-pathing is leveraged where the primary path is the local CVM and the two other paths would be remote. In the event where the primary path fails, one of the other paths will become active.

Similar to Autopathing with ESXi and Hyper-V, when the local CVM comes back online, it’ll take over as the primary path.

Resilient Capacity

Some refresher on terms used:

- Failure/Availability Domain(FD): Logical grouping of entities across which replicas are placed. Node is default domain on Nutanix clusters for node sizes > 3. Refer to Availability Domain section.
- Replication Factor(RF): Number of copies of data to be maintained on cluster for data availability.
- Fault Tolerance(FT): Number of failure(s) of entities at configured FD that can be handled by cluster, ensuring data on failed entity is rebuilt completely.

Resilient capacity is storage capacity in a cluster that can be consumed at the lowest availability/failure domain while maintaining the cluster’s ability to self-heal and recover to desired replication factor (RF) after FT failure(s) at the configured availability/failure domain. So in simple terms, Resilient Capacity = Total Cluster Capacity - Capacity needed to rebuild from FT failure(s).

Resilient Capacity Examples

Homogeneous Cluster Capacity

- Configured Availability Domain: Node
- Lowest Availability Domain: Node
- FT=1
- RF=2
- Node Capacities: 10TB,10TB,10TB,10TB
- Resilient Capacity = (40-10)*0.95 = 28.5TB. 95% is threshold at which Stargate goes read-only mode not taking any more user writes

Non-Homogeneous Cluster Capacity

- Configured Availability Domain: Block(1 node/block)
- Lowest Availability Domain: Node
- FT=1
- RF=2
- Node Capacities: 10TB,10TB,40TB,40TB
- Resilient Capacity = 40*0.95 = 38TB. 95% is threshold at which Stargate goes read-only mode not taking any more user writes

The resilient capacity in this case is 40TB and not 60TB because after losing the 40TB block, the cluster has node availability domain. At this level to maintain 2 data copies, the capacity available is 40TB which makes resilient capacity in this case to be 40TB overall.

It is recommended to keep clusters uniform and homogenous from capacity and failure domain perspective.

Starting with AOS 5.18, the resilient capacity is displayed in Prism Element storage summary widget with a gray line. Thresholds can be set to warn end users when cluster usage is reaching resilient capacity. By default that is set to 75%.

Prism can also show detailed storage utilizations on a per node basis which helps administrators understand resiliency on a per node basis. This is useful in clusters which have a skewed storage distribution.

When cluster usage is greater than resilient capacity for that cluster, the cluster might not be able to tolerate and recover from failures anymore. Cluster can possibly still recover and tolerate failure at a lower failure domain as resilient capacity is for configured failure domain. For example, a cluster with a node failure domain may still be able to self-heal and recover from a disk failure but cannot self-heal and recover from a node failure.

It is highly recommeded to not exceed the resilient capacity of a cluster in any circumstances to ensure proper functioning of cluster and maintain it’s ability to self-heal and recover from failures.

Capacity Optimization

The Nutanix platform incorporates a wide range of storage optimization technologies that work in concert to make efficient use of available capacity for any workload. These technologies are intelligent and adaptive to workload characteristics, eliminating the need for manual configuration and fine-tuning.

The following optimizations are leveraged:

- Erasure Coding (EC-X)
- Compression
- Deduplication

More detail on how each of these features can be found in the following sections.

The table describes which optimizations are applicable to workloads at a high-level:

Data Transform	**Best suited Application(s)**	**Comments**
Erasure Coding (EC-X)	Most, Ideal for Nutanix Files/Objects	Provides higher availability with reduced overheads than traditional RF. No impact to normal write or read I/O performance. Does have some read overhead in the case of a disk / node / block failure where data must be decoded.
Inline Compression	All	No impact to random I/O, helps increase storage tier utilization. Benefits large or sequential I/O performance by reducing data to replicate and read from disk.
Offline Compression	None	Given inline compression will compress only large or sequential writes inline and do random or small I/Os post-process, that should be used instead.
Dedupe	Full Clones in VDI, Persistent Desktops, P2V/V2V, Hyper-V (ODX)	Greater overall efficiency for data which wasn't cloned or created using efficient AOS clones

Erasure Coding

The Nutanix platform leverages a replication factor (RF) for data protection and availability. This method provides the highest degree of availability because it does not require reading from more than one storage location or data re-computation on failure. However, this does come at the cost of storage resources as full copies are required.

To provide a balance between availability while reducing the amount of storage required, AOS provides the ability to encode data using erasure codes (EC).

Similar to the concept of RAID (levels 4, 5, 6, etc.) where parity is calculated, EC encodes a strip of data blocks on different nodes and calculates parity. In the event of a host and/or disk failure, the parity can be leveraged to calculate any missing data blocks (decoding). In the case of AOS, the data block is an extent group. Based upon the read nature of the data (read cold vs. read hot), the system will determine placement of the blocks in the strip.

For data that is read cold, we will prefer to distribute the data blocks from the same vDisk across nodes to form the strip (same-vDisk strip). This simplifies garbage collection (GC) as the full strip can be removed in the event the vDisk is deleted. For read hot data we will prefer to keep the vDisk data blocks local to the node and compose the strip with data from different vDisks (cross-vDisk strip). This minimizes remote reads as the local vDisk's data blocks can be local and other VMs/vDisks can compose the other data blocks in the strip. In the event a read cold strip becomes hot, AOS will try to recompute the strip and localize the data blocks.

The number of data and parity blocks in a strip is configurable based upon the desired failures to tolerate. The configuration is commonly referred to as the number of <number of data blocks>/<number of parity blocks>.

For example, "RF2 like" availability (N+1) could consist of 3 or 4 data blocks and 1 parity block in a strip (3/1 or 4/1). "RF3 like" availability (N+2) could consist of 3 or 4 data blocks and 2 parity blocks in a strip (3/2 or 4/2). The default strip sizes are 4/1 for RF2 like availability and 4/2 for RF3 like availability. These can be overriden using nCLI is desired.

```
ncli container [create/edit] ... erasure-code=<N>/<K>
```

where N is number of data blocks and K is number of parity blocks

EC + Block Awareness

As of 5.8, EC can place data and parity blocks in a block aware manner (prior to 5.8 this was done at a node level).

Pre-existing EC containers will not immediately change to block aware placement after being upgraded to 5.8. If there are enough blocks (strip size (k+n) + 1) available in the cluster these previously node aware strips will move to block aware. New EC containers will build block aware EC strips.

The expected overhead can be calculated as <# parity blocks> / <# data blocks>. For example, a 4/1 strip has a 25% overhead or 1.25X compared to the 2X of RF2. A 4/2 strip has a 50% overhead or 1.5X compared to the 3X of RF3.

The following table characterizes the encoded strip sizes and example overheads:

	FT1 (RF2 equiv.)		**FT2 (RF3 equiv.)**	
Cluster Size (nodes)	EC Strip Size (data/parity blocks)	EC Overhead (vs. 2X of RF2)	EC Strip Size (data/parity)	EC Overhead (vs. 3X of RF3)
4	2/1	1.5X	N/A	N/A
5	3/1	1.33X	N/A	N/A
6	4/1	1.25X	2/2	2X
7	4/1	1.25X	3/2	1.6X
8+	4/1	1.25X	4/2	1.5X

The encoding is done post-process and leverages the Curator MapReduce framework for task distribution. Since this is a post-process framework, the traditional write I/O path is unaffected.

A normal environment using RF would look like the following:

Typical AOS RF Data Layout

In this scenario, we have RF2 data whose primary copies are local and replicas are distributed to other nodes throughout the cluster.

When a Curator full scan runs, it will find eligible extent groups which are available to become encoded. Eligible extent groups must be “write-cold” meaning they haven’t been written to for awhile. This is controlled with the following Curator Gflag: curator_erasure_code_threshold_seconds. After the eligible candidates are found, the encoding tasks will be distributed and throttled via Chronos.

The following figure shows an example 3/1 strip:

AOS Encoded Strip - Pre-savings

Once the data has been successfully encoded (strips and parity calculation), the replica extent groups are then removed.

The following figure shows the environment after EC has run with the storage savings:

AOS Encoded Strip - Post-savings

Starting with AOS 5.18, inline erasure coding can be enabled on containers using nCLI. Inline erasure coding encodes and creates coding strips inline by erasure coding the data without waiting for it to become write-cold. In inline erasure coding, the erasure coded strips are created on the same vDisk data by default. With AOS 6.6 there is also an option to create strips from cross vdisks preserving data locality if that is desired. Due to this, it is only recommended for workloads that do not require data locality or perform numerous overwrites. Objects deployment have inline EC-X turned on by default.

Pro tip

Erasure Coding pairs perfectly with inline compression which will add to the storage savings.

Compression

For a visual explanation, you can watch the following video: LINK

The Nutanix Capacity Optimization Engine (COE) is responsible for performing data transformations to increase data efficiency on disk. Currently compression is one of the key features of the COE to perform data optimization. AOS provides both inline and offline flavors of compression to best suit the customer's needs and type of data. As of 5.1, offline compression is enabled by default.

Inline compression will compress sequential streams of data or large I/O sizes (>64K) when written to the Extent Store (SSD + HDD). This includes data draining from OpLog as well as sequential data skipping it.

OpLog Compression

As of 5.0, the OpLog will now compress all incoming writes >4K that show good compression (Gflag: vdisk_distributed_oplog_enable_compression). This will allow for a more efficient utilization of the OpLog capacity and help drive sustained performance.

When drained from OpLog to the Extent Store the data will be decompressed, aligned and then re-compressed at a 32K aligned unit size (as of 5.1).

This feature is on by default and no user configuration is necessary.

Offline compression will initially write the data as normal (in an un-compressed state) and then leverage the Curator framework to compress the data cluster wide. When inline compression is enabled but the I/Os are random in nature, the data will be written un-compressed in the OpLog, coalesced, and then compressed in memory before being written to the Extent Store.

Nutanix leverages LZ4 and LZ4HC for data compression with AOS 5.0 and beyond. Prior to AOS 5.0 the Google Snappy compression library is leveraged which provides good compression ratios with minimal computational overhead and extremely fast compression / decompression rates.

Normal data will be compressed using LZ4 which provides a very good blend between compression and performance. For cold data, LZ4HC will be leveraged to provide an improved compression ratio.

Cold data is characterized into two main categories:

- Regular data: No R/W access for 3 days (Gflag: curator_medium_compress_mutable_data_delay_secs)
- Immutable data (snapshots): No R/W access for 1 day (Gflag: curator_medium_compress_immutable_data_delay_secs)

The following figure shows an example of how inline compression interacts with the AOS write I/O path:

Inline Compression I/O Path

Pro tip

Almost always use inline compression (compression delay = 0) as it will only compress larger / sequential writes and not impact random write performance.

This will also increase the usable size of the SSD tier increasing effective performance and allowing more data to sit in the SSD tier. Also, for larger or sequential data that is written and compressed inline, the replication for RF will be shipping the compressed data, further increasing performance since it is sending less data across the wire.

Inline compression also pairs perfectly with erasure coding.

For offline compression, all new write I/O is written in an un-compressed state and follows the normal AOS I/O path. After the compression delay (configurable) is met, the data is eligible to become compressed. Compression can occur anywhere in the Extent Store. Offline compression uses the Curator MapReduce framework and all nodes will perform compression tasks. Compression tasks will be throttled by Chronos.

The following figure shows an example of how offline compression interacts with the AOS write I/O path:

Offline Compression I/O Path

For read I/O, the data is first decompressed in memory and then the I/O is served.

You can view the current compression rates via Prism on the Storage > Dashboard page.

Elastic Dedupe Engine

For a visual explanation, you can watch the following video: LINK

The Elastic Dedupe Engine is a software-based feature of AOS which allows for data deduplication in the capacity (Extent Store) tiers. Streams of data are fingerprinted during ingest at a 16KB granularity (Controlled by: stargate_dedup_fingerprint) within a 1MB extent. Prior to AOS 5.11, AOS used only SHA-1 hash to fingerprint and identify candidates for dedupe. Since AOS 5.11, AOS now uses logical checksums to select candidates for dedupe. This fingerprint is only done on data ingest and is then stored persistently as part of the written block's metadata.

Contrary to traditional approaches which utilize background scans requiring the data to be re-read, Nutanix performs the fingerprint inline on ingest. For duplicate data that can be deduplicated in the capacity tier, the data does not need to be scanned or re-read, essentially duplicate copies can be removed.

To make the metadata overhead more efficient, fingerprint refcounts are monitored to track dedupability. Fingerprints with low refcounts will be discarded to minimize the metadata overhead. To minimize fragmentation full extents will be preferred for deduplication.

Pro tip

Use deduplication on your base images (you can manually fingerprint them using vdisk_manipulator) to take advantage of the unified cache.

Fingerprinting is done during data ingest of data with an I/O size of 64K or greater (initial I/O or when draining from OpLog). AOS then looks at hashes/fingerprints of each 16KB chunk within a 1MB extent and if it finds duplicates for more than 40% of chunks, it dedupes the entire extent. That resulted in many dedupe extents with reference count of 1 (no other duplicates) within the 1MB extent from the remaining 60% of extent, that ended up using metadata.

With AOS 6.6, the algorithm was further enhanced such that within a 1MB extent, only chunks that have duplicates will be marked for deduplication instead of entire extent reducing the metadata required. With AOS 6.6, changes were also made with the way dedupe metadata was stored. Before AOS 6.6, dedupe metadata was stored in a top level vdisk block map which resulted in dedupe metadata to be copied when snapshots were taken. This resulted in a metadata bloat. With AOS 6.6, that metadata is now stored in extent group id map which is a level lower than vdisk block map. Now when snapshots are taken of the vdisk, it does not result in copying of dedupe metadata and prevents metadata bloat. Once the fingerprintng is done, a background process will remove the duplicate data using the AOS MapReduce framework (Curator). For data that is being read, the data will be pulled into the AOS Unified Cache which is a multi-tier/pool cache. Any subsequent requests for data having the same fingerprint will be pulled directly from the cache. To learn more about the Unified Cache and pool structure, please refer to the Unified Cache sub-section in the I/O path overview.

The following figure shows an example of how the Elastic Dedupe Engine interacts with the AOS I/O path:

EDE I/O Path

You can view the current deduplication rates via Prism on the Storage > Dashboard page.

Storage Tiering and Prioritization

The Disk Balancing section below talks about how storage capacity was pooled among all nodes in a Nutanix cluster and that ILM would be used to keep hot data local. A similar concept applies to disk tiering, in which the cluster's SSD and HDD tiers are cluster-wide and AOS ILM is responsible for triggering data movement events. A local node's SSD tier is always the highest priority tier for all I/O generated by VMs running on that node, however all of the cluster's SSD resources are made available to all nodes within the cluster. The SSD tier will always offer the highest performance and is a very important thing to manage for hybrid arrays.

The tier prioritization can be classified at a high-level by the following:

AOS Tier Prioritization

Specific types of resources (e.g. SSD, HDD, etc.) are pooled together and form a cluster wide storage tier. This means that any node within the cluster can leverage the full tier capacity, regardless if it is local or not.

The following figure shows a high level example of what this pooled tiering looks like:

AOS Cluster-wide Tiering

A common question is what happens when a local node’s SSD becomes full? As mentioned in the Disk Balancing section, a key concept is trying to keep uniform utilization of devices within disk tiers. In the case where a local node’s SSD utilization is high, disk balancing will kick in to move the coldest data on the local SSDs to the other SSDs throughout the cluster. This will free up space on the local SSD to allow the local node to write to SSD locally instead of going over the network. A key point to mention is that all CVMs and SSDs are used for this remote I/O to eliminate any potential bottlenecks and remediate some of the hit by performing I/O over the network.

AOS Cluster-wide Tier Balancing

The other case is when the overall tier utilization breaches a specific threshold [curator_tier_usage_ilm_threshold_percent (Default=75)] where AOS ILM will kick in and as part of a Curator job will down-migrate data from the SSD tier to the HDD tier. This will bring utilization within the threshold mentioned above or free up space by the following amount [curator_tier_free_up_percent_by_ilm (Default=15)], whichever is greater. The data for down-migration is chosen using last access time. In the case where the SSD tier utilization is 95%, 20% of the data in the SSD tier will be moved to the HDD tier (95% -> 75%).

However, if the utilization was 80%, only 15% of the data would be moved to the HDD tier using the minimum tier free up amount.

AOS Tier ILM

AOS ILM will constantly monitor the I/O patterns and (down/up) migrate data as necessary as well as bring the hottest data local regardless of tier. The logic for up-migration (or horizontal) follows the same as that defined for egroup locality: “3 touches for random or 10 touches for sequential within a 10 minute window where multiple reads every 10 second sampling count as a single touch”.

Disk Balancing

For a visual explanation, you can watch the following video: LINK

AOS is designed to be a very dynamic platform which can react to various workloads as well as allow heterogeneous node types: compute heavy (3050, etc.) and storage heavy (60X0, etc.) to be mixed in a single cluster. Ensuring uniform distribution of data is an important item when mixing nodes with larger storage capacities. AOS has a native feature, called disk balancing, which is used to ensure uniform distribution of data throughout the cluster. Disk balancing works on a node’s utilization of its local storage capacity and is integrated with AOS ILM. Its goal is to keep utilization uniform among nodes once the utilization has breached a certain threshold.

NOTE: Disk balancing jobs are handled by Curator which has different priority queues for primary I/O (UVM I/O) and background I/O (e.g. disk balancing). This is done to ensure disk balancing or any other background activity doesn’t impact front-end latency / performance. In this cases the job’s tasks will be given to Chronos who will throttle / control the execution of the tasks. Also, movement is done within the same tier for disk balancing. For example, if data is skewed in the HDD tier, it can be moved amongst nodes in the same tier.

The following figure shows an example of a mixed cluster (3050 + 6050) in an “unbalanced” state:

Disk Balancing - Unbalanced State

Disk balancing leverages the AOS Curator framework and is run as a scheduled process as well as when a threshold has been breached (e.g., local node capacity utilization > n %). In the case where the data is not balanced, Curator will determine which data needs to be moved and will distribute the tasks to nodes in the cluster. In the case where the node types are homogeneous (e.g., 3050), utilization should be fairly uniform. However, if there are certain VMs running on a node which are writing much more data than others, this can result in a skew in the per node capacity utilization. In this case, disk balancing would run and move the coldest data on that node to other nodes in the cluster. In the case where the node types are heterogeneous (e.g., 3050 + 6020/50/70), or where a node may be used in a “storage only” mode (not running any VMs), there will likely be a requirement to move data.

The following figure shows an example the mixed cluster after disk balancing has been run in a “balanced” state:

Disk Balancing - Balanced State

In some scenarios, customers might run some nodes in a “storage-only” state where only the CVM will run on the node whose primary purpose is bulk storage capacity. In this case, the full node’s memory can be added to the CVM to provide a much larger read cache.

The following figure shows an example of how a storage only node would look in a mixed cluster with disk balancing moving data to it from the active VM nodes:

Disk Balancing - Storage Only Node

Snapshots and Clones

For a visual explanation, you can watch the following video: LINK

AOS provides native support for offloaded snapshots and clones which can be leveraged via VAAI, ODX, ncli, REST, Prism, etc. Both the snapshots and clones leverage the redirect-on-write algorithm which is the most effective and efficient. As explained in the Data Structure section above, a virtual machine consists of files (vmdk/vhdx) which are vDisks on the Nutanix platform.

A vDisk is composed of extents which are logically contiguous chunks of data, which are stored within extent groups which are physically contiguous data stored as files on the storage devices. When a snapshot or clone is taken, the base vDisk is marked immutable and another vDisk is created as read/write. At this point, both vDisks have the same block map, which is a metadata mapping of the vDisk to its corresponding extents. Contrary to traditional approaches which require traversal of the snapshot chain (which can add read latency), each vDisk has its own block map. This eliminates any of the overhead normally seen by large snapshot chain depths and allows you to take continuous snapshots without any performance impact.

The following figure shows an example of how this works when a snapshot is taken (source: NTAP):

Example Snapshot Block Map

The same method applies when a snapshot or clone of a previously snapped or cloned vDisk is performed:

Multi-snap Block Map and New Write

The same methods are used for both snapshots and/or clones of a VM or vDisk(s). When a VM or vDisk is cloned, the current block map is locked and the clones are created. These updates are metadata only, so no I/O actually takes place. The same method applies for clones of clones; essentially the previously cloned VM acts as the “Base vDisk” and upon cloning, that block map is locked and two “clones” are created: one for the VM being cloned and another for the new clone. There is no imposed limit on the maximum number of clones.

They both inherit the prior block map and any new writes/updates would take place on their individual block maps.

Multi-Clone Block Maps

As mentioned previously, each VM/vDisk has its own individual block map. So in the above example, all of the clones from the base VM would now own their block map and any write/update would occur there.

In the event of an overwrite the data will go to a new extent / extent group. For example, if data exists at offset **o1** in extent **e1** that was being overwritten, Stargate would create a new extent **e2** and track that the new data was written in extent **e2** at offset **o2**. The Vblock map tracks this down to the byte level.

The following figure shows an example of what this looks like:

Clone Block Maps - New Write

Any subsequent clones or snapshots of a VM/vDisk would cause the original block map to be locked and would create a new one for R/W access.

Networking and I/O

The Nutanix platform does not leverage any backplane for inter-node communication and only relies on a standard 10GbE network. All storage I/O for VMs running on a Nutanix node is handled by the hypervisor on a dedicated private network. The I/O request will be handled by the hypervisor, which will then forward the request to the private IP on the local CVM. The CVM will then perform the remote replication with other Nutanix nodes using its external IP over the public 10GbE network. For all read requests, these will be served completely locally in most cases and never touch the 10GbE network. This means that the only traffic touching the public 10GbE network will be AOS remote replication traffic and VM network I/O. There will, however, be cases where the CVM will forward requests to other CVMs in the cluster in the case of a CVM being down or data being remote. Also, cluster-wide tasks, such as disk balancing, will temporarily generate I/O on the 10GbE network.

The following figure shows an example of how the VM's I/O path interacts with the private and public 10GbE network:

AOS Networking

Data Locality

Being a converged (compute+storage) platform, I/O and data locality are critical to cluster and VM performance with Nutanix. As explained above in the I/O path, all read/write IOs are served by the local Controller VM (CVM) which is on each hypervisor adjacent to normal VMs. A VM's data is served locally from the CVM and sits on local disks under the CVM's control. When a VM is moved from one hypervisor node to another (or during a HA event), the newly migrated VM's data will be served by the now local CVM. When reading old data (stored on the now remote node/CVM), the I/O will be forwarded by the local CVM to the remote CVM. All write I/Os will occur locally right away. AOS will detect the I/Os are occurring from a different node and will migrate the data locally in the background, allowing for all read I/Os to now be served locally. The data will only be migrated on a read as to not flood the network.

Data locality occurs in two main flavors:

- Cache Locality
  - Pulling remote data into the local Stargate's Unified Cache. This is done at a 4K granularity.
  - For instances where there are no local replicas, the requests will be forward to the Stargate(s) containing the replicas which will return the data and the local Stargate will store this locally then return the I/O. All subsequent requests for that data will be returned from the cache.

- Extent Group (egroup) Locality
  - Migrating the vDisk extent group(s) (egroups) to be stored in the local Stargate's Extent Store.
  - If a replica egroup is already local, no movement is necessary.
  - In this scenario the actual replica egroup will be re-localized after certain I/O thresholds are met. We don't automatically re-localize / migrate egroups to ensure we're leveraging the network efficiently.
  - For AES enabled egroups the same horizontal migration occurs for cases where replicas aren't local and the patterns are met.

The following figure shows an example of how data will "follow" the VM as it moves between hypervisor nodes:

Data Locality

Thresholds for Data Migration

Cache locality occurs in real time and will be determined based upon vDisk ownership. When a vDisk / VM moves from one node to another the "ownership" of those vDisk(s) will transfer to the now local CVM. Once the ownership has transferred the data can be cached locally in the Unified Cache. In the interim the cache will be wherever the ownership is held (the now remote host). The previously hosting Stargate will relinquish the vDisk token when it sees remote I/Os for 300+ seconds at which it will then be taken by the local Stargate. Cache coherence is enforced as ownership is required to cache the vDisk data.

Egroup locality is a sampled operation and an extent group will be migrated when the following occurs: "3 touches for random or 10 touches for sequential within a 10 minute window where multiple reads every 10 second sampling count as a single touch".

Shadow Clones

The Distributed Storage Fabric has a feature called 'Shadow Clones', which allows for distributed caching of particular vDisks or VM data which is in a 'multi-reader' scenario. A great example of this is during a VDI deployment many 'linked clones' will be forwarding read requests to a central master or 'Base VM'. In the case of VMware View, this is called the replica disk and is read by all linked clones, and in XenDesktop, this is called the MCS Master VM. This will also work in any scenario which may be a multi-reader scenario (e.g., deployment servers, repositories, etc.). Data or I/O locality is critical for the highest possible VM performance and a key struct of AOS.

With Shadow Clones, AOS will monitor vDisk access trends similar to what it does for data locality. However, in the case there are requests occurring from more than two remote CVMs (as well as the local CVM), and all of the requests are read I/O, the vDisk will be marked as immutable. Once the disk has been marked as immutable, the vDisk can then be cached locally by each CVM making read requests to it (aka Shadow Clones of the base vDisk). This will allow VMs on each node to read the Base VM's vDisk locally. In the case of VDI, this means the replica disk can be cached by each node and all read requests for the base will be served locally. NOTE: The data will only be migrated on a read as to not flood the network and allow for efficient cache utilization. In the case where the Base VM is modified, the Shadow Clones will be dropped and the process will start over. Shadow clones are enabled by default (as of 4.0.2) and can be enabled/disabled using the following NCLI command: ncli cluster edit-params enable-shadow-clones=<true/false>.

The following figure shows an example of how Shadow Clones work and allow for distributed caching:

Shadow Clones

Storage Layers and Monitoring

The Nutanix platform monitors storage at multiple layers throughout the stack, ranging from the VM/Guest OS all the way down to the physical disk devices. Knowing the various tiers and how these relate is important whenever monitoring the solution and allows you to get full visibility of how the ops relate. The following figure shows the various layers of where operations are monitored and the relative granularity which are explained below:

Storage Layers

Virtual Machine Layer

- Key Role: Metrics reported by the hypervisor for the VM
- Description: Virtual Machine or guest level metrics are pulled directly from the hypervisor and represent the performance the VM is seeing and is indicative of the I/O performance the application is seeing.
- When to use: When troubleshooting or looking for VM level detail

Hypervisor Layer

- Key Role: Metrics reported by the Hypervisor(s)
- Description: Hypervisor level metrics are pulled directly from the hypervisor and represent the most accurate metrics the hypervisor(s) are seeing. This data can be viewed for one of more hypervisor node(s) or the aggregate cluster. This layer will provide the most accurate data in terms of what performance the platform is seeing and should be leveraged in most cases. In certain scenarios the hypervisor may combine or split operations coming from VMs which can show the difference in metrics reported by the VM and hypervisor. These numbers will also include cache hits served by the Nutanix CVMs.
- When to use: Most common cases as this will provide the most detailed and valuable metrics.

Controller Layer

- Key Role: Metrics reported by the Nutanix Controller(s)
- Description: Controller level metrics are pulled directly from the Nutanix Controller VMs (e.g., Stargate 2009 page) and represent what the Nutanix front-end is seeing from NFS/SMB/iSCSI or any back-end operations (e.g., ILM, disk balancing, etc.). This data can be viewed for one of more Controller VM(s) or the aggregate cluster. The metrics seen by the Controller Layer should normally match those seen by the hypervisor layer, however will include any backend operations (e.g., ILM, disk balancing). These numbers will also include cache hits served by memory. In certain cases, metrics like (IOPS), might not match as the NFS / SMB / iSCSI client might split a large IO into multiple smaller IOPS. However, metrics like bandwidth should match.

- When to use: Similar to the hypervisor layer, can be used to show how much backend operation is taking place.

Disk Layer

- Key Role: Metrics reported by the Disk Device(s)
- Description: Disk level metrics are pulled directly from the physical disk devices (via the CVM) and represent what the back-end is seeing. This includes data hitting the OpLog or Extent Store where an I/O is performed on the disk. This data can be viewed for one of more disk(s), the disk(s) for a particular node, or the aggregate disks in the cluster. In common cases, it is expected that the disk ops should match the number of incoming writes as well as reads not served from the memory portion of the cache. Any reads being served by the memory portion of the cache will not be counted here as the op is not hitting the disk device.
- When to use: When looking to see how many ops are served from cache or hitting the disks.

Metric and Stat Retention

Metrics and time series data is stored locally for 90 days in Prism Element. For Prism Central and Insights, data can be stored indefinitely (assuming capacity is available).

Services

» Download this section as PDF (opens in a new tab/window)

Nutanix Guest Tools (NGT)

Nutanix Guest Tools (NGT) is a software-based in-guest agent framework that enables advanced VM management functionality through the Nutanix Platform.

The solution is composed of the NGT installer which is installed on the VMs and the Guest Tools Framework which is used for coordination between the agent and Nutanix platform.

The NGT installer contains the following components:

- Guest Agent Service
- Self-service Restore (SSR) aka File-level Restore (FLR) CLI
- VM Mobility Drivers (VirtIO drivers for AHV)
- VSS Agent and Hardware Provider for Windows VMs
- App Consistent snapshot support for Linux VMs (via scripts to quiesce)

This framework is composed of a few high-level components:

- Guest Tools Service
  - Gateway between the AOS and Nutanix services and the Guest Agent. Distributed across CVMs within the cluster with an elected NGT Leader which runs on the current Prism Leader (hosting cluster VIP)

- Guest Agent
  - Agent and associated services deployed in the VM's OS as part of the NGT installation process. Handles any local functions such as VSS and Self-service Restore (SSR) and interacts with the Guest Tools Service.

The following figure shows the high-level mapping of the components:

Guest Tools Mapping

Guest Tools Service

The Guest Tools Service is composed of two main roles:

- NGT Leader
    - Handles requests coming from NGT Proxy and interfaces with AOS components. A single NGT Leader is dynamically elected per cluster; in the event the current leader fails a new one will be elected. The service listens internally on port 2073.

- NGT Proxy
    - Runs on every CVM and will forward requests to the NGT Leader to perform the desired activity. The current VM acting as the Prism Leader (hosting the VIP) will be the active CVM handling communication from the Guest Agent. Listens externally on port 2074.

Current NGT Leader

You can find the IP of the CVM hosting the NGT Leader role with the following command (run on any CVM):

```
nutanix_guest_tools_cli get_leader_location
```

The figure shows the high-level mapping of the roles:

Guest Tools Service

Guest Agent

The Guest Agent is composed of the following high-level components as mentioned prior:

Guest Agent

Communication and Security

The Guest Agent Service communicates with the Guest Tools Service via a serial port connection from VM to the CVM. If the service is unable to reach the CVM via serial port it will fail back to a network connection to the Nutanix Cluster VIP using SSL. For deployments where the Nutanix cluster components and UVMs are on a different network (hopefully all), ensure that the following are possible to allow this fallback communication method:

- Ensure routed communication from UVM network(s) to Cluster VIP

OR

- Create a firewall rule (and associated NAT) from UVM network(s) allowing communication with the Cluster VIP on port 2074 (preferred)

The Guest Tools Service acts as a Certificate Authority (CA) and is responsible for generating certificate pairs for each NGT-enabled UVM. This certificate is embedded into the ISO which is configured for the UVM and used as part of the NGT deployment process. These certificates are installed inside the UVM as part of the installation process.

NGT Agent Installation

NGT Agent installation can be performed via Prism, install packages, or CLI and scripts such as nCLI, REST, or PowerShell. The install packages are available in .exe (Windows), .rpm and .deb (Linux) formats and are available on the Nutanix support portal. These installers can be installed manually, via scripts, or 3rd party tools. The 3rd party tool method enables NGT installation at scale utilizing existing tools.

Bulk NGT Deployment

Rather than installing NGT on each VM, you can use the install packages mentioned earlier or embed and deploy NGT in your base image. Use the NGT Bulk Install instructions for packaged deployment.

Use the following process to leverage NGT inside a base image:

1. Install NGT on a template VM and ensure communication with CVM
2. Clone VMs from the template VM
3. Mount NGT ISO on each clone (required to get new certificate pair)
    - Example: ncli ngt mount vm-id=<CLONE_ID> OR via Prism

4. Power on clones

When the cloned VM is booted, it will detect the new NGT ISO, copy relevant configuration files and new certificates, and start communicating with the Guest Tools Service.

OS Customization

Nutanix provides native OS customization capabilities leveraging CloudInit and Sysprep. CloudInit is a package which handles bootstrapping of Linux cloud servers. This allows for the early initialization and customization of a Linux instance. Sysprep is a OS customization for Windows.

Some typical uses include:

- Setting Hostname
- Installing packages
- Adding users / key management
- Custom scripts

Supported Configurations

The solution is applicable to Linux guests running on AHV, including versions below (list may be incomplete, refer to documentation for a fully supported list):

- Hypervisors:
  - AHV

- Operating Systems:
  - Linux - most modern distributions
  - Windows - most modern distributions

Pre-Requisites

In order for CloudInit to be used the following are necessary:

- CloudInit package must be installed in Linux image

Sysprep is available by default in Windows installations.

Package Installation

CloudInit can be installed (if not already) using the following commands:

Red Hat Based Systems (CentOS, RHEL)

```
yum -y install CloudInit
```

Debian Based Systems (Ubuntu)

```
apt-get -y update; apt-get -y install CloudInit
```

Sysprep is part of the base Windows installation.

Image Customization

To leverage a custom script for OS customization, a check box and inputs is available in Prism or the REST API. This option is specified during the VM creation or cloning process:

Custom Script - Input Options

Nutanix has a few options for specifying the custom script path:

- ADSF Path
  - Use a file which has been previously upload to ADSF

- Upload a file
  - Upload a file which will be used

- Type or paste script
  - CloudInit script or Unattend.xml text

Nutanix passes the user data script to CloudInit or Sysprep process during first boot by creating a CD-ROM which contains the script. Once the process is complete we will remove the CD-ROM.

Input formatting

The platform supports a good amount of user data input formats. Some of the key formats are outlined below:

User-Data Script (CloudInit - Linux)

A user-data script is a simple shell script that will be executed very late in the boot process (e.g. “rc.local-like”).

The scripts will begin similar to any bash script: “#!”.

Below we show an example user-data script:

```
#!/bin/bash
touch /tmp/fooTest
mkdir /tmp/barFolder
```

Include File (CloudInit - Linux)

The include file contains a list of urls (one per line). Each of the URLs will be read and they will be processed similar to any other script.

The scripts will begin with: “#include”.

Below we show an example include script:

```
#include
http://s3.amazonaws.com/path/to/script/1
http://s3.amazonaws.com/path/to/script/2
```

Cloud Config Data (CloudInit - Linux)

The cloud-config input type is the most common and specific to CloudInit.

The scripts will begin with: “#cloud-config”

Below we show an example cloud config data script:

```
#cloud-config

# Set hostname
hostname: foobar

# Add user(s)
users:
 - name: nutanix
 sudo: ['ALL=(ALL) NOPASSWD:ALL']
 ssh-authorized-keys:
 - ssh-rsa: PUBKEY
 lock-passwd: false
 passwd: PASSWORD

# Automatically update all of the packages
package_upgrade: true
package_reboot_if_required: true

# Install the LAMP stack
packages:
 - httpd
 - mariadb-server
 - php
 - php-pear
 - php-mysql

# Run Commands after execution
runcmd:
 - systemctl enable httpd
```

Validating CloudInit Execution

CloudInit log files can be found in /var/log/cloud-init.log and cloud-init-output.log.

Unattend.xml (Sysprep - Windows)

The unattend.xml file is the input file Sysprep uses for image customization on boot, you can read more here: LINK

The scripts will begin with: “<?xml version=”1.0” ?>”.

Attached is an example unattend.xml file

Karbon (Container Services)

Nutanix provides the ability to leverage persistent containers on the Nutanix platform using Kubernetes (currently). It was previously possible to run Docker on Nutanix platform; however, data persistence was an issue given the ephemeral nature of containers.

Container technologies like Docker are a different approach to hardware virtualization. With traditional virtualization each VM has its own Operating System (OS) but they share the underlying hardware. Containers, which include the application and all its dependencies, run as isolated processes that share the underlying Operating System (OS) kernel.

The following table shows a simple comparison between VMs and Containers:

Metric	Virtual Machines (VM)	Containers

Virtualization Type	Hardware-level virtualization	OS kernel virtualization
Overhead	Heavyweight	Lightweight
Provisioning Speed	Slower (seconds to minutes)	Real-time / fast (us to ms)
Performance Overhead	Limited performance	Native performance
Security	Fully isolated (more secure)	Process-level isolation (less secure)

Supported Configurations

The solution is applicable to the configurations below (list may be incomplete, refer to documentation for a fully supported list):

Hypervisor(s):

- AHV

Container System(s)*:

- Docker 1.13

*As of 4.7, the solution only supports storage integration with Docker based containers. However, any other container system can run as a VM on the Nutanix platform.

Container Services Constructs

The following entities compose Karbon Container Services:

- **Nutanix Docker Machine Driver:** Handles Docker container host provisioning via Docker Machine and the AOS Image Service
- **Nutanix Docker Volume Plugin:** Responsible for interfacing with AOS Volumes to create, mount, format and attach volumes to the desired container

The following entities compose Docker (note: not all are required):

- **Docker Image:** The basis and image for a container
- **Docker Registry:** Holding space for Docker Images
- **Docker Hub:** Online container marketplace (public Docker Registry)
- **Docker File:** Text file describing how to construct the Docker image
- **Docker Container:** Running instantiation of a Docker Image
- **Docker Engine:** Creates, ships and runs Docker containers
- **Docker Swarm:** Docker host clustering / scheduling platform
- **Docker Daemon:** Handles requests from Docker Client and does heavy lifting of building, running and distributing containers
- **Docker Store:** Marketplace for trusted and enterprise ready containers

Services Architecture

The Nutanix solution currently leverages Docker Engine running in VMs which are created using Docker Machine. These machines can run in conjunction with normal VMs on the platform.

Docker - High-level Architecture

Nutanix has developed a Docker Volume Plugin which will create, format and attach a volume to container(s) using the AOS Volumes feature. This allows the data to persist as a container is power cycled / moved.

Data persistence is achieved by using the Nutanix Volume Plugin which will leverage AOS Volumes to attach a volume to the host / container:

Docker - Volumes

Pre-Requisites

In order for Container Services to be used the following are necessary:

- Nutanix cluster must be AOS 4.7 or later
- A CentOS 7.0+ or a Rhel 7.2+ OS image with the iscsi-initiator-utils package installed must be downloaded and exist as an image in the AOS Image Service
- The Nutanix Data Services IP must be configured
- Docker Toolbox must be installed on the client machine used for configuration
- Nutanix Docker Machine Driver must be in client's PATH

Docker Host Creation

Assuming all pre-requisites have been met the first step is to provision the Nutanix Docker Hosts using Docker Machine:

```
docker-machine -D create -d nutanix \
--nutanix-username PRISM_USER --nutanix-password PRISM_PASSWORD \
--nutanix-endpoint CLUSTER_IP:9440 --nutanix-vm-image DOCKER_IMAGE_NAME \ --nutanix-vm-network NETWORK_NAME \
```

The following figure shows a high-level overview of the backend workflow:

Docker - Host Creation Workflow

The next step is to SSH into the newly provisioned Docker Host(s) via docker-machine ssh:

```
docker-machine ssh DOCKER_HOST_NAME
```

To install the Nutanix Docker Volume Plugin run:

```
docker plugin install ntnx/nutanix_volume_plugin PRISM_IP= DATASERVICES_IP= PRISM_PASSWORD= PRISM_USERNAME= DEFAULT_CONTAINER= --alias nutanix
```

After that runs you should now see the plugin enabled:

```
[root@DOCKER-NTNX-00 ~]# docker plugin ls
ID Name Description Enabled
37fba568078d nutanix:latest Nutanix volume plugin for docker true
```

Docker Container Creation

Once the Nutanix Docker Host(s) have been deployed and the volume plugin has been enabled, you can provision containers with persistent storage.

A volume using the AOS Volumes can be created using the typical Docker volume command structure and specifying the Nutanix volume driver. Example usage below:

```
docker volume create \
VOLUME_NAME --driver nutanix

Example:
docker volume create PGDataVol --driver nutanix
```

The following command structure can be used to create a container using the created volume. Example usage below:

```
docker run -d --name CONTAINER_NAME \
-p START_PORT:END_PORT --volume-driver nutanix \
-v VOL_NAME:VOL_MOUNT_POINT DOCKER_IMAGE_NAME

Example:
docker run -d --name postgresexample -p 5433:5433 --volume-driver nutanix -v PGDataVol:/var/lib/postgresql/data postgres:latest
```

The following figure shows a high-level overview of the backend workflow:

Docker - Container Creation Workflow

You now have a container running with persistent storage!

Backup and Disaster Recovery

» Download this section as PDF (opens in a new tab/window)

Nutanix provides native backup and disaster recovery (DR) capabilities allowing users to backup, restore and DR VM(s) and objects running on DSF to both on-premises or cloud environments (Xi). As of AOS 5.11 Nutanix released a feature called Leap which abstracts a lot of these concepts. For more information on Leap, refer to the 'Leap' chapter in the 'Book of Backup / DR Services'.

We will cover the following items in the following sections:

- Implementation Constructs
- Protecting Entities
- Backup and Restore
- Replication and DR

NOTE: Though Nutanix provides native options for backup and dr, traditional solutions (e.g. Commvault, Rubrik, etc.) can also be used, leveraging some of the native features the platform provides (VSS, snapshots, etc.).

Implementation Constructs

Within Nutanix Backup and DR, there are a few key constructs:

Protection Domain (PD)

- Key Role: Macro group of VMs and/or files to protect
- Description: A group of VMs and/or files to be replicated together on a desired schedule. A PD can protect a full container or you can select individual VMs and/or files

Pro tip

Create multiple PDs for various services tiers driven by a desired RPO/RTO. For file distribution (e.g. golden images, ISOs, etc.) you can create a PD with the files to replication.

Consistency Group (CG)

- Key Role: Subset of VMs/files in PD to be crash-consistent

- Description: VMs and/or files which are part of a Protection Domain which need to be snapshotted in a crash-consistent manner. This ensures that when VMs/files are recovered, they come up in a consistent state. A protection domain can have multiple consistency groups.

Pro tip

Group dependent application or service VMs in a consistency group to ensure they are recovered in a consistent state (e.g. App and DB)

Snapshot Schedule

- Key Role: Snapshot and replication schedule
- Description: Snapshot and replication schedule for VMs in a particular PD and CG

Pro tip

The snapshot schedule should be equal to your desired RPO

Retention Policy

- Key Role: Number of local and remote snapshots to keep
- Description: The retention policy defines the number of local and remote snapshots to retain. NOTE: A remote site must be configured for a remote retention/replication policy to be configured.

Pro tip

The retention policy should equal the number of restore points required per VM/file

Remote Site

- Key Role: A remote Nutanix cluster
- Description: A remote Nutanix cluster which can be leveraged as a target for backup or DR purposes.

Pro tip

Ensure the target site has ample capacity (compute/storage) to handle a full site failure. In certain cases replication/DR between racks within a single site can also make sense.

The following figure shows a logical representation of the relationship between a PD, CG, and VM/Files for a single site:

DR Construct Mapping

Protecting Entities

You can protect Entities (VMs, VGs, Files), using the following workflow:

From the Data Protection page, select + Protection Domain -> Async DR:

DR - Async PD

Specify a PD name and click ‘Create’

DR - Create PD

Select entities to protect:

DR - Async PD

Click ‘Protect Selected Entities’

DR - Protect Entities

The protect entities will now be displayed under ‘Protected Entities’

DR - Protected Entities

Click ‘Next’, then click ‘Next Schedule’ to create a snapshot and replication schedule

Enter the desired snapshot frequency, retention and any remote sites for replication

DR - Create Schedule

Click ‘Create Schedule’ to complete the schedule completion.

Multiple Schedules

It is possible to create multiple snapshot / replication schedules. For example, if you want to have a local backup schedule occurring hourly and another schedule which replicated to a remote site daily.

It is important to mention that a full container can be protected for simplicity. However, the platform provides the ability to protect down to the granularity of a single VM and/or file level.

Backup and Restore

Nutanix backup capabilities leverage the native DSF snapshot capabilities and are invoked by Cerebro and performed by Stargate. These snapshot capabilities are zero copy to ensure efficient storage utilization and low overhead. You can read more on Nutanix snapshots in the ‘Snapshots and Clones’ section.

Typical backup and restore operations include:

- Snapshot: Create a restore point and replicate (if necessary)
- Restore: Restore VM(s) / File(s) from a previous snapshot (replaces original objects)
- Clone: Similar to restore but does not replace original objects (creates new objects as desired snapshot)

From the Data Protection Page, you can see the protection domains (PD) previously created in the ‘Protecting Entities’ section.

DR - View PDs

Once you’re selected a target PD you can see the various options:

DR - PD Actions

If you click ‘Take Snapshot’ you can take an ad-hoc snapshot of the selected PD and replicate to a remote site if necessary:

DR - Take Snapshot

You can also ‘Migrate’ the PD which will fail over the entities to a remote site:

DR - Migrate

In the event of a migrate (controlled failover), the system will take a new snapshot, replicate then promote the other site with the newly created snap.

Pro tip

With AOS 5.0 and above you can now leverage a single node replication target data protection.

You can also view the PD snapshot(s) in the table below:

DR - Local Snapshots

From here you can restore or clone a PD snapshot:

DR - Restore Snapshot

If you choose to ‘Create new entities’ that will be like cloning the snapshot of the PD to new entities with the desired prefixes. Otherwise ‘Overwrite existing entities’ will replace the current entities with those at the time of the snapshot.

Storage only backup target

For backup / archival only purposes, it is possible to configure a storage only Nutanix cluster as a remote site which will act as a backup target. This will allow data to be replicated to / from the storage only cluster.

App Consistent Snapshots

Nutanix provides native VmQueisced Snapshot Service (VSS) capabilities for queiscing OS and application operations which ensure an application consistent snapshot is achieved.

VmQueisced Snapshot Service (VSS)

VSS is typically a Windows specific term for Volume Shadow Copy Service. However, since this solution applies to both Windows and Linux we've modified the term to VmQueisced Snapshot Service.

Supported Configurations

The solution is applicable to both Windows and Linux guests. Refer to “NGT Compatibility” in the Compatibility and Interoperability Matrix for a full list of supported guest OS’s: LINK

Pre-Requisites

In order for Nutanix VSS snapshots to be used the following are necessary:

- Nutanix Platform
  - Cluster Virtual IP (VIP) must be configured

- Guest OS / UVM
  - NGT must be installed
  - CVM VIP must be reachable on port 2074

- Disaster Recovery Configuration
  - UVM must be in PD with ‘Use application consistent snapshots’ enabled

Backup Architecture

As of 4.6 this is achieved using the native Nutanix Hardware VSS provider which is installed as part of the Nutanix Guest Tools package. You can read more on the guest tools in the ‘Nutanix Guest Tools’ section.

The following image shows a high-level view of the VSS architecture:

You can perform an application consistent snapshot by following the normal data protection workflow and selecting ‘Use application consistent snapshots’ when protecting the VM.

Enabling/Disabling Nutanix VSS

When NGT is enabled for a UVM, the Nutanix VSS snapshot capability is enabled by default. However, you can turn off this capability with the following command:

```
ncli ngt disable-applications application-names=vss_snapshot vm_id=VM_ID
```

Windows VSS Architecture

The Nutanix VSS solution is integrated with the Windows VSS framework. The following shows a high-level view of the architecture:

Nutanix VSS - Windows Architecture

Once NGT is installed you can see the NGT Agent and VSS Hardware Provider services:

VSS Hardware Provider

Linux VSS Architecture

The Linux solution works similar to the Windows solution, however scripts are leveraged instead of the Microsoft VSS framework as it doesn’t exist in Linux distros.

The Nutanix VSS solution is integrated with the Windows VSS framework. The following shows a high-level view of the architecture:

Nutanix VSS - Linux Architecture

The pre-freeze and post-thaw scripts are located in the following directories:

- Pre-freeze: /sbin/pre_freeze
- Post-thaw: /sbin/post-thaw

Eliminating ESXi Stun

ESXi has native app consistent snapshot support using VMware guest tools. However, during this process, delta disks are created and ESXi "stuns" the VM in order to remap the virtual disks to the new delta files which will handle the new write IO. Stuns will also occur when a VMware snapshot is deleted.

During this stun process the VM its OS cannot execute any operations and is essentially in a "stuck" state (e.g. pings will fail, no IO). The duration of the stun will depend on the number of vmdks and speed of datastore metadata operations (e.g. create new delta disks, etc.)

By using Nutanix VSS we completely bypass the VMware snapshot / stun process and have little to no impact to performance or VM / OS availability.

Replication and Disaster Recovery (DR)

For a visual explanation, you can watch the following video: LINK

Nutanix provides native DR and replication capabilities, which build upon the same features explained in the Snapshots & Clones section. Cerebro is the component responsible for managing the DR and replication in DSF. Cerebro runs on every node and a Cerebro leader is elected (similar to NFS leader) and is responsible for managing replication tasks. In the event the CVM acting as Cerebro leader fails, another is elected and assumes the role. The Cerebro page can be found on :2020. The DR function can be broken down into a few key focus areas:

- Replication Topologies
- Replication Lifecycle
- Global Deduplication

Replication Topologies

Traditionally, there are a few key replication topologies: Site to site, hub and spoke, and full and/or partial mesh. Contrary to traditional solutions which only allow for site to site or hub and spoke, Nutanix provides a fully mesh or flexible many-to-many model.

Example Replication Topologies

Essentially, this allows the admin to determine a replication capability that meets their company’s needs.

Replication Lifecycle

Nutanix replication leverages the Cerebro service mentioned above. The Cerebro service is broken into a “Cerebro Leader”, which is a dynamically elected CVM, and Cerebro Workers, which run on every CVM. In the event where the CVM acting as the “Cerebro Leader” fails, a new “Leader” is elected.

The Cerebro Leader is responsible for managing task delegation to the local Cerebro Workers as well as coordinating with remote Cerebro Leader(s) when remote replication is occurring.

During a replication, the Cerebro Leader will figure out which data needs to be replicated, and delegate the replication tasks to the Cerebro Workers which will then tell Stargate which data to replicate and to where.

Replicated data is protected at multiple layers throughout the process. Extent reads on the source are checksummed to ensure consistency for source data (similar to how any DSF read occurs) and the new extent(s) are checksummed at the target (similar to any DSF write). TCP provides consistency on the network layer.

The following figure shows a representation of this architecture:

Replication Architecture

It is also possible to configure a remote site with a proxy which will be used as a bridgehead for all coordination and replication traffic coming from a cluster.

Pro tip

When using a remote site configured with a proxy, always utilize the cluster IP as that will always be hosted by the Prism Leader and available, even if CVM(s) go down.

The following figure shows a representation of the replication architecture using a proxy:

Replication Architecture - Proxy

In certain scenarios, it is also possible to configure a remote site using a SSH tunnel where all traffic will flow between two CVMs.

The following figure shows a representation of the replication architecture using a SSH tunnel:

Replication Architecture - SSH Tunnel

Global Deduplication

As explained in the Elastic Deduplication Engine section above, DSF has the ability to deduplicate data by just updating metadata pointers. The same concept is applied to the DR and replication feature. Before sending data over the wire, DSF will query the remote site and check whether or not the fingerprint(s) already exist on the target (meaning the data already exists). If so, no data will be shipped over the wire and only a metadata update will occur. For data which doesn’t exist on the target, the data will be compressed and sent to the target site. At this point, the data existing on both sites is usable for deduplication.

The following figure shows an example three site deployment where each site contains one or more protection domains (PD):

Replication Deduplication

Note

Fingerprinting must be enabled on the source and target container / vstore for replication deduplication to occur.

NearSync

Building upon the traditional asynchronous (async) replication capabilities mentioned previously; Nutanix has introduced support for near synchronous replication (NearSync).

NearSync provides the best of both worlds: zero impact to primary I/O latency (like async replication) in addition to a very low RPO (like sync replication (metro)). This allows users have a very low RPO without having the overhead of requiring synchronous replication for writes.

This capability uses a new snapshot technology called light-weight snapshot (LWS). Unlike the traditional vDisk based snapshots used by async, this leverages markers and is completely OpLog based (vs. vDisk snapshots which are done in the Extent Store).

Mesos is a new service added to manage the snapshot layer and abstract the complexities of the full/incremental snapshots. Cerebro continues to manage the high-level constructs and policies (e.g. consistency groups, etc.) whereas Mesos is responsible for interacting with Stargate and controlling the LWS lifecycle.

The following figure shows an example of the communication between the NearSync components:

NearSync Component Interaction

When a user configures a snapshot frequency <= 15 minutes, NearSync is automatically leveraged. Upon this, an initial seed snapshot is taken then replicated to the remote site(s). Once this completes in < 60 minutes (can be the first or n later), another seed snapshot is immedatly taken and replicated in addition to LWS snapshot replication starting. Once the second seed snapshot finishes replication, all already replicated LWS snapshots become valid and the system is in stable NearSync.

The following figure shows an example timeline from enabling NearSync to execution:

NearSync Replication Lifecycle

During a steady run state vDisk snapshots are taken every hour. Rather than sending the snapshot over to the remote site in addition to the LWS, the remote site composes the vDisk snapshot based upon the prior vDisk snapshot and the LWS from that time.

In the event NearSync falls out of sync (e.g. network outage, WAN latency, etc.) causing the LWS replication to take > 60 minutes, the system will automatically switch back to vDisk based snapshots. When one of these completes in < 60 minutes, the system will take another snapshot immediately as well as start replicating LWS. Once the full snapshot completes, the LWS snapshots become valid and the system is in stable NearSync. This process is similar to the initial enabling of NearSync.

When a LWS based snap is restored (or cloned), the system will take a clone of the latest vDisk snapshot and apply the LWS incrementally until the desired LWS is reached.

The following figure shows an example of how a LWS based snapshot is restored:

vDisk Restore from LWS

Metro Availability

Nutanix provides native “stretch clustering” capabilities which allow for a compute and storage cluster to span multiple physical sites. In these deployments, the compute cluster spans two locations and has access to a shared pool of storage.

This expands the VM HA domain from a single site to between two sites providing a near 0 RTO and a RPO of 0.

In this deployment, each site has its own Nutanix cluster, however the containers are “stretched” by synchronously replicating to the remote site before acknowledging writes.

The following figure shows a high-level design of what this architecture looks like:

Metro Availability - Normal State

In the event of a site failure, an HA event will occur where the VMs can be restarted on the other site. The failover process is typically a manual process. With the AOS 5.0 release a Metro Witness can be configured which can automate the failover. The witness can be downloaded via the Portal and is configured via Prism.

The following figure shows an example site failure:

Metro Availability - Site Failure

In the event where there is a link failure between the two sites, each cluster will operate independently. Once the link comes back up, the sites will be re-synchronized (deltas-only) and synchronous replication will start occurring.

The following figure shows an example link failure:

Metro Availability - Link Failure

Administration

» Download this section as PDF (opens in a new tab/window)

Important Pages

These are advanced Nutanix pages besides the standard user interface that allow you to monitor detailed stats and metrics. The URLs are formatted in the following way: http://<Nutanix CVM IP/DNS>:<Port/path (mentioned below)> Example: http://MyCVM-A:2009 NOTE: if you're on a different subnet IPtables will need to be disabled on the CVM to access the pages.

2009 Page

This is a Stargate page used to monitor the back end storage system and should only be used by advanced users. I'll have a post that explains the 2009 pages and things to look for.

2009/latency Page

This is a Stargate page used to monitor the back end latency.

2009/vdisk_stats Page

This is a Stargate page used to show various vDisk stats including histograms of I/O sizes, latency, write hits (e.g., OpLog, eStore), read hits (cache, SSD, HDD, etc.) and more.

2009/h/traces Page

This is the Stargate page used to monitor activity traces for operations.

2009/h/vars Page

This is the Stargate page used to monitor various counters.

2010 Page

This is the Curator page which is used for monitoring Curator runs.

2010/master/control Page

This is the Curator control page which is used to manually start Curator jobs

2011 Page

This is the Chronos page which monitors jobs and tasks scheduled by Curator.

2020 Page

This is the Cerebro page which monitors the protection domains, replication status and DR.

2020/h/traces Page

This is the Cerebro page used to monitor activity traces for PD operations and replication.

2030 Page

This is the main Acropolis page and shows details about the environment hosts, any currently running tasks and networking details.

2030/sched Page

This is an Acropolis page used to show information about VM and resource scheduling used for placement decisions. This page shows the available host resources and VMs running on each host.

2030/tasks Page

This is an Acropolis page used to show information about Acropolis tasks and their state. You can click on the task UUID to get detailed JSON about the task.

2030/vms Page

This is an Acropolis page used to show information about Acropolis VMs and details about them. You can click on the VM Name to connect to the console.

Cluster Commands

Check cluster status

Description: Check cluster status from the CLI

```
cluster status
```

Check local CVM service status

Description: Check a single CVM's service status from the CLI

```
genesis status
```

Check upgrade status

```
upgrade_status
```

Perform manual / cli upgrade

```
download NUTANIXINSTALLERPACKAGE.tar.gz into ~/tmp
```

```
tar xzf NUTANIXINSTALLERPACKAGE.tar.gz
```

```
cd ~/tmp
```

```
./install/bin/cluster -i ./install upgrade
```

Node(s) upgrade Hypervisor

upgrade status

Description: Check hypervisor upgrade status from the CLI on any CVM

```
host_upgrade_status
```

Detailed logs (on every CVM)

```
~/data/logs/host_upgrade.out
```

Restart cluster service from CLI

Description: Restart a single cluster service from the CLI

Stop service

```
cluster stop ServiceName
```

Start stopped services

```
cluster start  #NOTE: This will start all stopped services
```

Start cluster service from CLI

Description: Start stopped cluster services from the CLI

Start stopped services

```
cluster start  #NOTE: This will start all stopped services
```

OR

Start single service

Start single service: cluster start ServiceName

Restart local service from CLI

Description: Restart a single cluster service from the CLI

Stop Service

```
genesis stop ServiceName
```

Start Service

```
cluster start
```

Start local service from CLI

Description: Start stopped cluster services from the CLI

```
cluster start #NOTE: This will start all stopped services
```

Cluster add node from cmdline

Description: Perform cluster add-node from CLI

ncli cluster discover-nodes | egrep “Uuid” | awk ‘{print $4}’ | xargs -I UUID ncli cluster add-node node-uuid=UUID </pre>

Find cluster id

Description: Find the cluster ID for the current cluster

```
zeus_config_printer | grep cluster_id
```

Open port

Description: Enable port through IPtables

```
sudo vi /etc/sysconfig/iptables -A INPUT -m state --state NEW -m tcp -p tcp --dport PORT -j ACCEPT
sudo service iptables restart
```

Check for Shadow Clones

Description: Displays the shadow clones in the following format: name#id@svm_id

```
vdisk_config_printer | grep '#'
```

Reset Latency Page Stats

Description: Reset the Latency Page (CVM IP):2009/latency) counters

```
allssh "wget 127.0.0.1:2009/latency/reset"
```

Find vDisk information

Description: Find vDisk information and details including name, id, size, iqn and others

```
vdisk_config_printer
```

Find Number of vDisks

Description: Find the current number of vDisks (files) on DSF

```
vdisk_config_printer | grep vdisk_id | wc -l
```

Get detailed vDisk information

Description: Displays a provided vDisks egroup IDs, size, transformation and savings, garbage and replica placement

```
vdisk_usage_printer -vdisk_id=VDISK_ID
```

Start Curator scan from CLI

Description: Starts a Curator scan from the CLI

Full Scan

```
allssh "wget -O - "http://localhost:2010/master/api/client/StartCuratorTasks?task_type=2";"
```

Partial Scan

```
allssh "wget -O - "http://localhost:2010/master/api/client/StartCuratorTasks?task_type=3";"
```

Refresh Usage

```
allssh "wget -O - "http://localhost:2010/master/api/client/RefreshStats";"
```

Check under replicated data via CLI

Description: Check for under replicated data using curator_cli

```
curator_cli get_under_replication_info summary=true
```

Compact ring

Description: Compact the metadata ring

```
allssh "nodetool -h localhost compact"
```

Find NOS version

Description: Find the NOS version (NOTE: can also be done using NCLI)

```
allssh "cat /etc/nutanix/release_version"
```

Find CVM version

Description: Find the CVM image version

Manually fingerprint vDisk(s)

Description: Create fingerprints for a particular vDisk (For dedupe) NOTE: dedupe must be enabled on the container

```
vdisk_manipulator –vdisk_id=vDiskID --operation=add_fingerprints
```

Manually fingerprint all vDisk(s)

Description: Create fingerprints for all vDisk(s) (For dedupe) NOTE: dedupe must be enabled on the container

```
for vdisk in `vdisk_config_printer | grep vdisk_id | awk {'print $2'}`; do vdisk_manipulator -vdisk_id=$vdisk --operation=add_fingerprints; done
```

Echo Factory_Config.json for all cluster nodes

Description: Echos the factory_config.jscon for all nodes in the cluster

```
allssh "cat /etc/nutanix/factory_config.json"
```

Upgrade a single Nutanix node’s NOS version

Description: Upgrade a single node’s NOS version to match that of the cluster

```
~/cluster/bin/cluster -u NEW_NODE_IP upgrade_node
```

List files (vDisk) on DSF

Description: List files and associated information for vDisks stored on DSF

```
Nfs_ls
```

Get help text

```
Nfs_ls --help
```

Install Nutanix Cluster Check (NCC)

Description: Installs the Nutanix Cluster Check (NCC) health script to test for potential issues and cluster health

Download NCC from the Nutanix Support Portal (portal.nutanix.com)

SCP .tar.gz to the /home/nutanix directory

Untar NCC .tar.gz

```
tar xzmf ncc.tar.gz --recursive-unlink
```

Run install script

```
./ncc/bin/install.sh -f ncc.tar.gz
```

Create links

```
source ~/ncc/ncc_completion.bash
echo "source ~/ncc/ncc_completion.bash" >> ~/.bashrc
```

Run Nutanix Cluster Check (NCC)

Description: Runs the Nutanix Cluster Check (NCC) health script to test for potential issues and cluster health. This is a great first step when troubleshooting any cluster issues.

Make sure NCC is installed (steps above)

Run NCC health checks

```
ncc health_checks run_all
```

List tasks using progress monitor cli

```
progress_monitor_cli -fetchall
```

Remove task using progress monitor cli

```
progress_monitor_cli --entity_id=ENTITY_ID --operation=OPERATION --entity_type=ENTITY_TYPE --delete
# NOTE: operation and entity_type should be all lowercase with k removed from the begining
```

Metrics and Thresholds

The following section will cover specific metrics and thresholds on the Nutanix back end. More updates to these coming shortly!

Gflags

More coming soon!

Troubleshooting & Advanced Administration

Find Acropolis logs

Description: Find Acropolis logs for the cluster

```
allssh "cat ~/data/logs/Acropolis.log"
```

Find cluster error logs

Description: Find ERROR logs for the cluster

```
allssh "cat ~/data/logs/COMPONENTNAME.ERROR"
```

Example for Stargate

```
allssh "cat ~/data/logs/Stargate.ERROR"
```

Find cluster fatal logs

Description: Find FATAL logs for the cluster

```
allssh "cat ~/data/logs/COMPONENTNAME.FATAL"
```

Example for Stargate

```
allssh "cat ~/data/logs/Stargate.FATAL"
```

Using the 2009 Page (Stargate)

In most cases Prism should be able to give you all of the information and data points you require. However, in certain scenarios, or if you want some more detailed data you can leverage the Stargate aka 2009 page. The 2009 page can be viewed by navigating to CVMIP:2009.

Accessing back-end pages

If you're on a different network segment (L2 subnet) you'll need to add a rule in IP tables to access any of the back-end pages.

At the top of the page is the overview details which show various details about the cluster:

2009 Page - Stargate Overview

In this section there are two key areas to look out for, the first being the I/O queues that shows the number of admitted / outstanding operations.

The figure shows the queues portion of the overview section:

2009 Page - Stargate Overview - Queues

The second portion is the unified cache details that shows information on cache sizes and hit rates.

The figure shows the unified cache portion of the overview section:

2009 Page - Stargate Overview - Unified Cache

NOTE: these values are per Stargate / CVM

The next section is the ‘Cluster State’ that shows details on the various Stargates in the cluster and their disk usages.

The figure shows the Stargates and disk utilization (available/total):

2009 Page - Cluster State - Disk Usage

The next section is the ‘NFS Worker’ section which will show various details and stats per vDisk.

The figure shows the vDisks and various I/O details:

2009 Page - NFS Worker - vDisk Stats

Pro tip

When looking at any potential performance issues the following metrics should be looked at:

1. Avg. latency
2. Avg. op size
3. Avg. outstanding

For more specific details the vdisk_stats page holds a plethora of information.

Using the 2009/vdisk_stats Page

The 2009 vdisk_stats page is a detailed page which provides even further data points per vDisk. This page includes details and a histogram of items like randomness, latency histograms, I/O sizes and working set details.

You can navigate to the vdisk_stats page by clicking on the ‘vDisk Id’ in the left hand column.

The figure shows the section and hyperlinked vDisk Id:

2009 Page - Hosted vDisks

This will bring you to the vdisk_stats page which will give you the detailed vDisk stats. NOTE: These values are real-time and can be updated by refreshing the page.

The first key area is the ‘Ops and Randomness’ section which will show a breakdown of whether the I/O patterns are random or sequential in nature.

The figure shows the ‘Ops and Randomness’ section:

2009 Page - vDisk Stats - Ops and Randomness

The next area shows a histogram of the frontend read and write I/O latency (aka the latency the VM / OS sees).

The figure shows the ‘Frontend Read Latency’ histogram:

2009 Page - vDisk Stats - Frontend Read Latency

The figure shows the ‘Frontend Write Latency’ histogram:

2009 Page - vDisk Stats - Frontend Write Latency

The next key area is the I/O size distribution that shows a histogram of the read and write I/O sizes.

The figure shows the ‘Read Size Distribution’ histogram:

2009 Page - vDisk Stats - Read I/O Size

The figure shows the ‘Write Size Distribution’ histogram:

2009 Page - vDisk Stats - Write I/O Size

The next key area is the ‘Working Set Size’ section which provides insight on working set sizes for the last 2 minutes and 1 hour. This is broken down for both read and write I/O.

The figure shows the ‘Working Set Sizes’ table:

2009 Page - vDisk Stats - Working Set

The ‘Read Source’ provides details on which tier or location the read I/Os are being served from.

The figure shows the ‘Read Source’ details:

2009 Page - vDisk Stats - Read Source

Pro tip

If you're seeing high read latency take a look at the read source for the vDisk and take a look where the I/Os are being served from. In most cases high latency could be caused by reads coming from HDD (Estore HDD).

The ‘Write Destination’ section will show where the new write I/O are coming in to.

The figure shows the ‘Write Destination’ table:

2009 Page - vDisk Stats - Write Destination

Pro tip

Random I/Os will be written to the Oplog, sequential I/Os will bypass the Oplog and be directly written to the Extent Store (Estore).

Another interesting data point is what data is being up-migrated from HDD to SSD via ILM. The ‘Extent Group Up-Migration’ table shows data that has been up-migrated in the last 300, 3,600 and 86,400 seconds.

The figure shows the ‘Extent Group Up-Migration’ table:

2009 Page - vDisk Stats - Extent Group Up-Migration

Using the 2010 Page (Curator)

The 2010 page is a detailed page for monitoring the Curator MapReduce framework. This page provides details on jobs, scans, and associated tasks.

You can navigate to the Curator page by navigating to http://:2010. NOTE: if you're not on the Curator Leader click on the IP hyperlink after 'Curator Leader: '.

The top of the page will show various details about the Curator Leader including uptime, build version, etc.

The next section is the ‘Curator Nodes’ table that shows various details about the nodes in the cluster, the roles, and health status. These will be the nodes Curator leverages for the distributed processing and delegation of tasks.

The figure shows the ‘Curator Nodes’ table:

2010 Page - Curator Nodes

The next section is the ‘Curator Jobs’ table that shows the completed or currently running jobs.

There are two main types of jobs which include a partial scan which is eligible to run every 60 minutes and a full scan which is eligible to run every 6 hours. NOTE: the timing will be variable based upon utilization and other activities.

These scans will run on their periodic schedules however can also be triggered by certain cluster events.

Here are some of the reasons for a jobs execution:

- Periodic (normal state)
- Disk / Node / Block failure
- ILM Imbalance
- Disk / Tier Imbalance

The figure shows the ‘Curator Jobs’ table:

2010 Page - Curator Jobs

The table shows some of the high-level activities performed by each job:

Activity	**Full Scan**	**Partial Scan**
ILM	X	X
Disk Balancing	X	X
Compression	X	X
Deduplication	X	
Erasure Coding	X	
Garbage Cleanup	X	

Clicking on the ‘Execution id’ will bring you to the job details page which displays various job stats as well as generated tasks.

The table at the top of the page will show various details on the job including the type, reason, tasks and duration.

The next section is the ‘Background Task Stats’ table which displays various details on the type of tasks, quantity generated and priority.

The figure shows the job details table:

2010 Page - Curator Job - Details

The figure shows the ‘Background Task Stats’ table:

2010 Page - Curator Job - Tasks

The next section is the ‘MapReduce Jobs’ table that shows the actual MapReduce jobs started by each Curator job. Partial scans will have a single MapReduce Job, full scans will have four MapReduce Jobs.

The figure shows the ‘MapReduce Jobs’ table:

2010 Page - MapReduce Jobs

Clicking on the ‘Job id’ will bring you to the MapReduce job details page which displays the tasks status, various counters and details about the MapReduce job.

The figure shows a sample of some of the job counters:

2010 Page - MapReduce Job - Counters

The next section on the main page is the ‘Queued Curator Jobs’ and ‘Last Successful Curator Scans’ section. These tables show when the periodic scans are eligible to run and the last successful scan’s details.

The figure shows the ’Queued Curator Jobs’ and ‘Last Successful Curator Scans’ section:

2010 Page - Queued and Successful Scans

Advanced CLI Information

Prism should provide all that is necessary in terms of normal troubleshooting and performance monitoring. However, there may be cases where you want to get more detailed information which is exposed on some of the backend pages mentioned above, or the CLI.

vdisk_config_printer

The vdisk_config_printer command will display a list of vdisk information for all vdisk on the cluster.

I’ve highlighted some of the important fields below:

- Vdisk ID
- Vdisk name
- Parent vdisk ID (if clone or snapshot)
- Vdisk size (Bytes)
- Container id
- To remove bool (to be cleaned up by curator scan)
- Mutability state (mutable if active r/w vdisk, immutable if snapshot)

The following shows example command output:

```
nutanix@NTNX-13SM35210012-A-CVM:~$ vdisk_config_printer | more
...
vdisk_id: 1014400
```

vdisk_usage_printer -vdisk_id=

The vdisk_usage_printer is used to get detailed information for a vdisk, its extents and egroups.

I’ve highlighted some of the important fields below:

- Egroup ID
- Egroup extent count
- Untransformed egroup size
- Transformed egroup size
- Transform ratio
- Transformation type(s)
- Egroup replica locations (disk/cvm/rack)

The following shows example command output:

```
nutanix@NTNX-13SM35210012-A-CVM:~$ vdisk_usage_printer -vdisk_id=99999
 Egid # eids UT Size T Size ... T Type Replicas(disk/svm/rack)
 1256878 64 1.03 MB 1.03 MB ... D,[73 /14/60][184108644/184108632/60]
 1256881 64 1.03 MB 1.03 MB ... D,[73 /14/60][152/7/25]
 1256883 64 1.00 MB 1.00 MB ... D,[73 /14/60][184108642/184108632/60]
 1055651 4 4.00 MB 4.00 MB ... none[76 /14/60][184108643/184108632/60]
 1056604 4 4.00 MB 4.00 MB ... none[74 /14/60][184108642/184108632/60]
 1056605 4 4.00 MB 4.00 MB ... none[73 /14/60][152/7/25]
 ...
```

NOTE: Notice the egroup size for deduped vs. non-deduped egroups (1 vs. 4MB). As mentioned in the ‘Data Structure’ section, for deduped data, a 1MB egroup size is preferred to negate any potential fragmention cause by de-duplicating the data.

curator_cli display_data_reduction_report

The curator_cli display_data_reduction_report is used to get detailed information on the storage savings per container by transform (e.g. clone, snap, dedup, compression, erasure coding, etc.)

I’ve highlighted some of the important fields below:

- Container ID
- Technique (transform applied)
- Pre reduction Size
- Post reduction size
- Saved space
- Savings ratio

The following shows example command output:

```
CVM:~$ curator_cli display_data_reduction_report
Using curator leader: 99.99.99.99:2010
```

curator_cli get_vdisk_usage lookup_vdisk_ids=<COMMA SEPARATED VDISK ID(s)>

The curator_cli get_vdisk_usage lookup_vdisk_ids command is used to get various stats about the space used by each of the specified vdisks.

I've highlighted some of the important fields below:

- Vdisk ID
- Exclusive usage (Data referred to by only this vdisk)
- Logical uninherited (Data written to vdisk, may be inherited by a child in the event of clone)
- Logical dedup (Amount of vdisk data that has been deduplicated)
- Logical snapshot (Data not shared across vdisk chains)
- Logical clone (Data shared across vdisk chains)

The following shows example command output:

```
Using curator leader: 99.99.99.99:2010
VDisk usage stats:
+------------------------------------------------------------------------+
| VDisk Id | Exclusive | Logical | Logical | Logical | Logical |
| | usage | Uninherited | Dedup | Snapshot | Clone |
+------------------------------------------------------------------------+
| 254244142 | 596.06 MB | 529.75 MB | 6.70 GB | 11.55 MB | 214 MB |
| 15995052 | 599.05 MB | 90.70 MB | 7.14 GB | 0.00 KB | 4.81 MB |
| 203739387 | 31.97 GB | 31.86 GB | 24.3 MB | 0.00 KB | 4.72 GB |
| 22841153 | 147.51 GB | 147.18 GB | 0.00 KB | 0.00 KB | 0.00 KB |
...
```

curator_cli get_egroup_access_info

The curator_cli get_egroup_access_info is used to get detailed information on the number of egroups in each bucket based upon last access (read) / modify ([over]write). This information can be used to estimate the number of egroups which might be eligible candidates to leverage erasure coding.

I've highlighted some of the important fields below:

- Container ID
- Access \ Modify (secs)

The following shows example command output:

```
Using curator leader: 99.99.99.99:2010
```

Book of AHV

» Download this section as PDF (opens in a new tab/window)

AHV is the native Nutanix hypervisor and is based on the CentOS KVM foundation. It extends its base functinality to include features like HA, live migration, IP address management, etc.

AHV is validated as part of the Microsoft Server Virtualization Validation Program and is validated to run Microsoft OS and applications.

This book will cover the architecture and features of AHV.

AHV Architecture

» Download this section as PDF (opens in a new tab/window)

Node Architecture

In AHV deployments, the Controller VM (CVM) runs as a VM and disks are presented using PCI passthrough. This allows the full PCI controller (and attached devices) to be passed through directly to the CVM and bypass the hypervisor. AHV is based upon CentOS KVM. Full hardware virtualization is used for guest VMs (HVM).

AHV Node

KVM Architecture

Within KVM there are a few main components:

- KVM-kmod
  - KVM kernel module

- Libvirtd
  - An API, daemon and management tool for managing KVM and QEMU. Communication between AOS and KVM / QEMU occurs through libvirtd.

- Qemu-kvm
  - A machine emulator and virtualizer that runs in userspace for every Virtual Machine (domain). In AHV it is used for hardware-assisted virtualization and VMs run as HVMs.

The following figure shows the relationship between the various components:

KVM Component Relationship

Communication between AOS and KVM occurs via Libvirt.

Processor generation compatibility

Similar to VMware's Enhanced vMotion Capability (EVC) which allows VMs to move between different processor generations; AHV will determine the lowest processor generation in the cluster and constrain all QEMU domains to that level. This allows mixing of processor generations within an AHV cluster and ensures the ability to live migrate between hosts.

Configuration Maximums and Scalability

The following configuration maximums and scalability limits are applicable:

- Maximum cluster size: **32**
- Maximum vCPUs per VM: **Number of physical cores per host**
- Maximum memory per VM: **4.5TB or available physical node memory**
- Maximum virtual disk size: **9EB* (Exabyte)**
- Maximum VMs per host: **N/A – Limited by memory**
- Maximum VMs per cluster: **N/A – Limited by memory**

*AHV does not have a traditional storage stack like ESXi / Hyper-V; all disks are passed to the VM(s) as raw SCSI block devices. This means the maximum virtual disk size is limited by the maximum AOS vDisk size (9 Exabytes).

The above are true as of AHV 20220304.10013 and AOS 6.6. Refer to Configuration Maximums for other versions.

Compute

The following sections outline key capabilities of Nutanix AHV compute for workload management.

VM Templates

AHV has always had the image library which focused on capturing the data within a single vDisk so that it could be easily cloned, but input from the admin was needed to complete the process of declaring the CPU, memory and network details. VM Templates take this concept to the next level of simplicity and provides a familiar construct for admins that have utilized templates on other hypervisors.

AHV VM Templates are created from existing virtual machines, inheriting the attributes of the defining VM such as the CPU, memory, vdisks, and networking details. The template can then be configured to customize the guest OS upon deployment and can optionally provide a Windows license key. Templates allow for multiple versions to be maintained, allowing for easy updates such as operating system and application patches to be applied without the need to create a new template. Admins can choose which version of the template is active, allowing the updates to be staged ahead of time or the ability to switch back to a previous version if needed.

Memory Overcommit

One of the central benefits of virtualization is the ability to overcommit compute resources, making it possible to provision more CPUs to VMs than are physically present on the server host. Most workloads don’t need all of their assigned CPUs 100% of the time, and the hypervisor can dynamically allocate CPU cycles to workloads that need them at each point in time.

Much like CPU or network resources, memory can be overcommitted also. At any given time, the VMs on the host may or may not use all their allocated memory, and the hypervisor can share that unused memory with other workloads. Memory overcommit makes it possible for administrators to provision a greater number of VMs per host, by combining the unused memory and allocating it to VMs that need it.

AOS 6.1 brings memory overcommit to AHV as an option to allow administrators flexibility in environments such as test and development where additional memory and VM density are required. Overcommit is disabled by default and can be defined on a per-VM basis allowing sharing to be done on all or just a subset of the VMs on a cluster.

VM Affinity Policies

Different types of applications can have requirements that dictate whether the VMs should run on the same host or a different host. This is typically done for performance or availability benefits. Affinity controls enable you to govern where VMs run. AHV has two types of affinity controls:

- VM-host affinity
  - Strictly ties a VM to a host or group of hosts, so the VM only runs on that host or group. Affinity is particularly applicable for use cases that involve software licensing or VM appliances. In such cases, you often need to limit the number of hosts an application can run on or bind a VM appliance to a single host.

- Anti-affinity
  - AHV lets you declare that a given list of VMs shouldn’t run on the same hosts. Anti-affinity gives you a mechanism for allowing clustered VMs or VMs running a distributed application to run on different hosts, increasing the application’s availability and resiliency. To prefer VM availability over VM separation, the system overrides this type of rule when a cluster becomes constrained.

Virtual Trusted Platform Module (vTPM)

TPM technologies are designed to provide enhanced security and privacy in handling encryption operations. The purpose of the TPM is to ensure information storage is better protected from unauthorized access. The primary use case is storing secrets, making it difficult to access them without proper authorization.

The Trusted Computing Group outlines the TPM as a dedicated hardware chip that is soldered onto the motherboard in the computer, which works great in a bare metal deployment. In a virtualized environment using a hypervisor such as AHV or ESXi, the physical TPM chip approach does not scale to support multiple guest OS running on a single hardware configuration due to the following limitations.

- Key Storage - The storage available in a physical TPM chip can hold roughly three transient keys.

- TPM Isolation – The TPM chip does not provide any separation or isolation between multiple guest OS running on the same physical device.

To address the scaling issues for a virtualized environment, the hypervisor vendors have implemented a hypervisor-level software called virtual TPM (vTPM), which conforms to the Trusted Computing Group's TPM specification. vTPM emulates these TPM specifications in the same functional manner as a physical TPM chip creating a private TPM instance per VM guest within the hypervisor. vTPM allows each VM guest to have its own key storage, isolating it from the other guests running on the same physical server. To maintain this isolation, the vTPM does not use the hardware physical TPM chip on the server.

Each hypervisor vendor is responsible for protecting the vTPM instance from outside access. Nutanix AHV, for example, ensures isolation between VMs and encrypts vTPM data using a secure distributed secrets service called Mantle, preventing unauthorized access or tampering.

Live Migrations

Live migration allows the system to move VMs from one host to another while the VM is turned on without workload interruption, regardless of whether the administrator or an automatic process initiates the movement. Live migrations occur regularly in the cluster nodes, triggered by maintenance operations, ADS workload balancing, node expansion, or administrator-driven requests.

You can also use live migration to migrate VMs to another physical cluster in the same location or in a different location to rebalance workloads, run maintenance operations, or avoid planned interruptions. For cross-cluster live migrations, we recommend a network with 5ms of latency and support a maximum 40ms latency between clusters.

There are several stages to a VM live migration:

1. Create a VM placeholder on the destination host
2. Iteratively copy the source VMs memory to the destination VM placeholder
3. Pause the VM on the source host
4. Copy the final VM state from the source VM to the destination VM placeholder
5. Update the network switches to ensure continued connectivity
6. Resume the VM on the destination host
7. Delete the paused VM from the source

Step 2, the copying of the VM's memory, can occur over and up to a specific number of iterations (50 at the time of writing). As the VM is still running, AHV will keep track of the VM's memory which is actively being modified during the copy process in each iteration. Once an iteration is complete, AHV will analyze the amount of memory that has changed and still needs to be copied over to the destination, as well as the achieved rate of memory transfer over the network, to determine if another iteration is required or if the migration can proceed to the next step. The rate at which a VM is modifying memory is more important to a successful migration than the amount of memory in a VM, so AHV will proactively manage the speed of the VM to reduce the amount of memory needing to be sent in the next iteration.

Step 3, pausing of the VM on the source host, will only occur when AHV detects it can transfer the remaining memory in 300ms or less, ensuring that the VM will respond on the destination host after a very short period. This is the maximum stun window. If the remaining memory cannot be transferred within this window after the final iteration, then the migration can abort with a 'failure to converge' error. The migration may be automatically re-tried by ADS, host evacuation, or by the administrator manually triggering another migration.

Step 5, updating the network switches, is enacted by sending a RARP broadcast network packet to all devices on the subnet while the VM is being brought up. This ensures that external network switches are aware of where the VM is now running and can route packets appropriately, for example ensuring that TCP connections are not lost during the migration.

Generation ID

There are a range of applications that use the Generation ID to access a virtual machine identifier to validate if the VM was cloned or duplicated for licensing or functionality verification. Starting in AOS release 6.7, AHV creates a Generation ID for each VM that is created, which applications running inside of that VM have access to. Applications can then make decisions on how they want to behave based on whether the correct Generation ID is present. One notable example where this is important is in the case of Windows domain controllers, which can experience problems if accidentally cloned or rolled back without proper safeguards. Generation ID is a mechanism that provides this information to a VM, allowing these restrictions to be tested and enforced by the app.

Acropolis Dynamic Scheduler (ADS)

ADS, the Acropolis Dynamic Scheduler, is a key component in the AHV stack. ADS is responsible for VM migrations and placements during many operations including resolving host hotspots (hosts with high CPU and storage usage), maintaining High Availability guarantees, optimizing VM placement to free up resources, and enforcing administrator-defined policies. ADS is always running in the background, continuously monitoring and optimizing the infrastructure.

Key features of ADS include:

- Initial VM placement - Choosing the AHV host for a VM and defragmenting the cluster if needed to ensure sufficient resources are available for the new VM
- Dynamic hotspot mitigation - Monitoring each host and, when a hotspot is detected, resolving that hotspot
- Background policy enforcement - for example moving VMs around to respect VM:Host affinity policies and VM:VM anti-affinity policies
- Enforcing High Availability guarantees - When HA guarantees are enabled, ADS will move VMs in the running system to ensure all VMs on each host can be recovered following the failure of an individual host.
- Dynamic GPU management - Supports NVIDIA GPUs to provide specific vGPU profiles based on the VMs running in the system
- Creating remediation plans based on the cost of movement to address each of the above cases

As ADS focuses on hotspot mitigation and has cost-minimizing remediation plans, this results in fewer VM movements than would be required for an active load-balancing scenario.

Hotspot Mitigation Example - Simple Plan

Figure 1 in the following example shows a host utilizing 90% of available CPU capacity. The CPU hotspot threshold is 85%, so ADS identifies that VM movements are required.

In Figure 2, ADS computes a plan that will move one of the 8GB VMs from the third host to the second host, making sufficient space for one of the VMs on the first host to move over.

The amount of memory on each of the VMs on the first host is taken into account when deciding which VM can be moved more easily. The final state in Figure 3 shows the smaller of the two VMs on the first host moving over to the third host as that is the lowest cost move.

ADS Mitigation Example

ADS plans are usually more complex than this, as multiple dimensions are considered both for hotspot detection and for cost-minimization of the remediation plans.

Networking

AHV leverages Open vSwitch (OVS) for all VM networking. VM networking is configured through Prism / ACLI and each VM nic is connected into a tap interface.

The following figure shows a conceptual diagram of the OVS architecture:

Open vSwitch Network Overview

In the prior image you see a few types of components:

Open vSwitch (OVS)

OVS is an open source software switch implemented in the Linux kernel and designed to work in a multiserver virtualization environment. By default, OVS behaves like a layer-2 learning switch that maintains a MAC address table. The hypervisor host and VMs connect to virtual ports on the switch.

OVS supports many popular switch features, including VLAN tagging, Link Aggregation Control Protocol (LACP), port mirroring, and quality of service (QoS), to name a few. Each AHV server maintains an OVS instance, and all OVS instances combine to form a single logical switch. Constructs called bridges manage the switch instances residing on the AHV hosts.

Bridge

Bridges act as virtual switches to manage network traffic between physical and virtual network interfaces. The default AHV configuration includes an OVS bridge called br0 and a native Linux bridge called virbr0. The virbr0 Linux bridge carries management and storage communication between the CVM and AHV host. All other storage, host, and VM network traffic flows through the br0 OVS bridge. The AHV host, VMs, and physical interfaces use “ports” for connectivity to the bridge.

Port

Ports are logical constructs created in a bridge that represent connectivity to the virtual switch. Nutanix uses several port types, including internal, tap, VXLAN, and bond:

- An internal port—with the same name as the default bridge (br0)—provides access for the AHV host.
- Tap ports act as bridge connections for virtual NICs presented to VMs.
- VXLAN ports are used for the IP address management functionality provided by Acropolis.
- Bonded ports provide NIC teaming for the physical interfaces of the AHV host.

Bond

Bonded ports aggregate the physical interfaces on the AHV host. By default, a bond named br0-up is created in bridge br0. After the node imaging process, all interfaces are placed within a single bond, which is a requirement for the foundation imaging process. Changes to the default bond, br0-up, often rename this to bond0. Nutanix recommends using the name br0-up to quickly identify the interface as the bridge br0 uplink.

OVS bonds allow for several load-balancing modes, including active-backup, balance-slb and balance-tcp. LACP can also be activated for a bond. The “bond_mode” setting is not specified during installation and therefore defaults to active-backup, which is the recommended configuration.

Uplink Load Balancing

Briefly mentioned in the prior section, it is possible to balance traffic across bond uplinks.

The following bond modes are available:

- active-backup
  - Default configuration which transmits all traffic over a single active adapter. If the active adapter becomes unavailable, another adapter in the bond will become active. Limits throughput to a single nic's bandwidth. (Recommended)

- balance-slb
  - Balances each VM's nic across adapters in the bond (e.g. VM A nic 1 - eth0 / nic 2 - eth1). Limits VM per-nic throughput to a single nic's bandwidth, however a VM with x nics can leverage x * adapter bandwidth (assuming x is the same for the number of VM nics and physical uplink adapters in the bond). NOTE: has caveats for multicast traffic

- balance-tcp / LACP
  - Balances each VM nic's TCP session across adapters in the bond. Limits per-nic throughput to the maximum bond bandwidth (number of physical uplink adapters * speed). Requires link aggregation and used when LACP is required.

You can find additional information on bonds in the AHV networking guide.

VM NIC Types

AHV supports the following VM network interface types:

- Access (default)
- Trunk (4.6 and above)

By default VM nics will be created as Access interfaces (similar to what you'd see with a VM nic on a port group), however it is possible to expose a trunked interface up to the VM's OS. Trunked NICs send the primary VLAN untagged, and all additional VLANs as tags to the same vNIC on the VM. This is useful to bring multiple networks to a VM without adding vNICs.

A trunked interface can be added with the following command:

```
vm.nic_create VM_NAME vlan_mode=kTrunked trunked_networks=ALLOWED_VLANS network=NATIVE_VLAN
```

Example:

```
vm.nic_create fooVM vlan_mode=kTrunked trunked_networks=10,20,30 network=vlan.10
```

Service Chaining

AHV Service chaining allows us to intercept all traffic and forward to a packet processor (NFV, appliance, virtual appliance, etc.) functions transparently as part of the network path.

Common uses for service chaining:

- Firewall (e.g. Palo Alto, etc.)
- IDS/IPS/network monitors (e.g. packet capture)

Within service chaining there are two types of way:

Service chain - Packet Processors

- Inline packet processor
  - Intercepts packets inline as they flow through OVS
  - Can modify and allow/deny packet
  - Common use: firewalls

- Tap packet processor
  - Inspects packets as they flow, can only read as it's a tap into the packet flow
  - Common uses: IDS/IPS/network monitor

Any service chaining is done after the Flow - Microsegmentation rules are applied and before the packet leaves the local OVS. This occurs in the network function bridge (br.nf):

Service Chain - Flow

NOTE: it is possible to string together multiple NFV / packet processors in a single chain. Service chaining is only applicable when Acropolis controls the network stack. Service chaining is not currently supported for network controller-based VLANs or VPCs.

Network Controller

The network controller was released in AOS 6.0 to enable Flow Virtual Networking and VPC overlay subnets. With AOS release 6.7, Nutanix enhanced the network controller, adding support for guest VMs in VLAN-backed subnets. When the network controller is enabled, a new subnet label, VLAN basic is created to describe the existing VLANs managed by the Acropolis leader. Network controller-managed subnets have no label and are simply called VLANs.

Supported Configurations

Core Use Cases:

- AHV networking with additional features in network controller-enabled VLAN subnets
- Flow Virtual Networking VPC overlay subnets
- Microsegmentation for network controller-enabled VLAN-backed subnets
- Microsegmentation within VPC overlay subnets

Management interfaces(s):

- Prism Central (PC)

Supported Environment(s):

- On-Premises:
  - AHV

- Nutanix Cloud Clusters (NC2)
  - NC2 Azure

Prerequisites for network controller-enabled VLANs:

- Prism Central 2023.3
- AOS 6.7

- AHV 9
- MSP enabled

Upgrades:

- Included in LCM

Pro tip

Deploy Prism Central as extra large for maximum network controller scalability.

Implementation Constructs

The network controller (previously Atlas Network Controller) controls the virtual networking stack used by Flow Virtual Networking VPCs and network controller-enabled VLAN-backed subnets in AHV. The network controller enables configurations at scale and allows for new features such as subnets that exist in multiple Prism Element clusters. This network stack will enable future networking and network security features. With the expanded capabilities, there are a few constructs that are different from what is used in the OVS-based architecture.

The network controller is used to centrally manage and administer VLANs, overlay subnets, IP address pools, and security policies from Prism Central. The network controller runs on Prism Central.

Network Control Plane

The network controller programs the network control plane and determines how packets are processed. This control plane uses Open Virtual Network (https://www.ovn.org/en/architecture/). A new virtual switch named brAtlas is applied to every AHV host.

Network Controller Control Plane

Network Bridges

There are 2 bridge types used with the network controller, brAtlas and br0.

brAtlas

The bridge brAtlas is a virtual switch that is applied to every AHV host and managed by the network controller. Guest VMs in VPCs and network controller-backed VLANs connect to brAtlas via a tap interface.

Network Controller Architecture Overview

br0

The bridge br0 is an uplink bridge that functions as a layer 2 switch for brAtlas and connects the physical networks. There can be multiple uplink bridges if required. These additional uplink bridges would follow the same naming construct and be named br0, br1, br2, etc.

VLANs and Logical Switches

For guest VLAN tagging, every VLAN is mapped to a logical switch within brAtlas. Each logical switch will have an associated localnet port connecting to the uplink virtual switch, such as br0. A localnet port is the point of connectivity between logical switches and physical networks.

A localnet port is made up of a pair of patch ports between brAtlas and br0. A network controller-enabled VLAN-backed subnet maps to a logical switch in brAtlas.

How Nutanix AHV Works

» Download this section as PDF (opens in a new tab/window)

Storage I/O Path

AHV does not leverage a traditional storage stack like ESXi or Hyper-V. All disk(s) are passed to the VM(s) as raw SCSI block devices. This keeps the I/O path lightweight and optimized.

Each AHV host runs an iSCSI redirector which regularly checks the health of Stargates throughout the cluster using NOP commands.

In the iscsi_redirector log (located in /var/log/ on the AHV host), you can see each Stargate’s health:

```
2017-08-18 19:25:21,733 - INFO - Portal 192.168.5.254:3261 is up
...
2017-08-18 19:25:25,735 - INFO - Portal 10.3.140.158:3261 is up
2017-08-18 19:25:26,737 - INFO - Portal 10.3.140.153:3261 is up
```

NOTE: The local Stargate is shown via its 192.168.5.254 internal address.

In the following you can see the iscsi_redirector is listening on 127.0.0.1:3261:

```
[root@NTNX-BEAST-1 ~]# netstat -tnlp | egrep tcp.*3261
Proto ... Local Address Foreign Address State PID/Program name
...
tcp ... 127.0.0.1:3261 0.0.0.0:* LISTEN 8044/python
...
```

QEMU is configured with the iSCSI redirector as the iSCSI target portal. Upon a login request, the redirector will perform an iSCSI login redirect to a healthy Stargate (preferably the local one).

iSCSI Multi-pathing - Normal State

The preferred controller type is virtio-scsi (default for SCSI devices). IDE devices, while possible, are not recommended for most scenarios. In order for virtio to be used with Windows the virtio drivers, Nutanix mobility drivers, or Nutanix guest tools must be installed. Modern Linux distros ship with virtio pre-installed.

In the event where the active Stargate goes down (thus failing to respond to the NOP OUT command), the iSCSI redirector will mark the local Stargate as unhealthy. When QEMU retries the iSCSI login, the redirector will redirect the login to another healthy Stargate.

iSCSI Multi-pathing - Local CVM Down

Once the local CVM’s Stargate comes back up (and begins responding to the NOP OUT commands), the remote Stargate will quiesce then kill all connections to remote iSCSI sessions. QEMU will then attempt an iSCSI login again and will be redirected to the local Stargate.

iSCSI Multi-pathing - Local CVM Back Up

Traditional I/O Path

Like every hypervisor and OS there is a mix of user and kernel space components which interact to perform a common activity. Prior to reading the following, it is recommended to read the ‘User vs. Kernel Space’ section to learn more about how each interact with eachother.

When a VM performs an I/O it will perform the following (some steps have been excluded for clarity):

1. VM’s OS perform SCSI command(s) to virtual device(s)
2. Virtio-scsi takes those requests and places them in the guest’s memory
3. Requests are handled by the QEMU main loop
4. Libiscsi inspects each request and forwards
5. Network layer forwards requests to local CVM (or externally if local is unavailable)
6. Stargate handles request(s)

The following shows this sample flow:

AHV VirtIO Data Path - Classic

Looking at an AHV host, you can see qemu-kvm has established sessions with a healthy Stargate using the local bridge and IPs. For external communication, the external host and Stargate IPs will be used. NOTE: There will be one session per disk device (look at PID 24845)

```
[root@NTNX-BEAST-1 log]# netstat -np | egrep tcp.*qemu
Proto ... Local Address Foreign Address State PID/Program name
tcp..........192.168.5.1:50410 192.168.5.254:3261 ESTABLISHED 25293/qemu-kvm
tcp ... 192.168.5.1:50434 192.168.5.254:3261 ESTABLISHED 23198/qemu-kvm
tcp ... 192.168.5.1:50464 192.168.5.254:3261 ESTABLISHED 24845/qemu-kvm
tcp ... 192.168.5.1:50465 192.168.5.254:3261 ESTABLISHED 24845/qemu-kvm
...
```

Now in this path there are a few inefficiencies as the main loop is single threaded and libiscsi inspects every SCSI command.

Frodo I/O Path (aka AHV Turbo Mode)

As storage technologies continue to evolve and become more efficient, so must we. Given the fact that we fully control AHV and the Nutanix stack this was an area of opportunity.

In short Frodo is a heavily optimized I/O path for AHV that allows for higher throughput, lower latency and less CPU overhead.

Pro tip

Frodo is enabled by default on VMs powered on after AOS 5.5.X.

When a VM performs an I/O it will perform the following (some steps have been excluded for clarity):

1. VM's OS perform SCSI command(s) to virtual device(s)
2. Virtio-scsi takes those requests and places them in the guest's memory
3. **Requests are handled by Frodo**
4. **Custom libiscsi appends iscsi header and forwards**
5. Network layer forwards requests to local CVM (or externally if local is unavailable)
6. Stargate handles request(s)

The following shows this sample flow:

AHV VirtIO Data Path - Frodo

The following path does looks similar to the traditional I/O except for a few key differences:

- Qemu main loop is replaced by Frodo (vhost-user-scsi)
- Frodo exposes multiple virtual queues (VQs) to the guest (one per vCPU)
- Leverages multiple threads for multi-vCPU VMs
- Libiscsi is replaced by our own much more lightweight version

To the guest it will notice that it now has multiple queues for the disk device(s), other than that it'll just see the performance improvements. In some cases we've seen a CPU overhead reduction of 25% to perform the I/O and performance increases of up to 3x compared to Qemu! Comparing to another hypervisor we've seen CPU overhead to perform I/Os drop by up to 3x.

Looking at an AHV host, you will see a frodo process for each VM (qemu-kvm process) running:

```
[root@drt-itppc03-1 ~]# ps aux | egrep frodo
... /usr/libexec/qemu-kvm ... -chardev socket,id=frodo0,fd=3 \
 -device vhost-user-scsi-pci,chardev=frodo0,num_queues=16...

... /usr/libexec/frodo ... 127.0.0.1:3261 -t iqn.2010-06.com.nutanix:vmdisk...
...
```

Pro tip

To take advantage of Frodo's multiple threads / connections, you must have >= 2 vCPU for a VM when it is powered on.

It can be characterized by the following:

- 1 vCPU UVM:
  - 1 Frodo thread / session per disk device

- >= 2 vCPU UVM:
  - 2 Frodo threads / sessions per disk device

In the following, you can see Frodo has established sessions with a healthy Stargate using the local bridge and IPs. For external communication, the external host and Stargate IPs will be used.

```
[root@NTNX-BEAST-1 log]# netstat -np | egrep tcp.*frodo
Proto ... Local Address Foreign Address State PID/Program name
tcp ... 192.168.5.1:39568 192.168.5.254:3261 ESTABLISHED 42957/frodo
tcp ... 192.168.5.1:39538 192.168.5.254:3261 ESTABLISHED 42957/frodo
tcp ... 192.168.5.1:39580 192.168.5.254:3261 ESTABLISHED 42957/frodo
tcp ... 192.168.5.1:39592 192.168.5.254:3261 ESTABLISHED 42957/frodo
...
```

IP Address Management

The Acropolis IP address management (IPAM) solution provides the ability to establish a DHCP scope and assign addresses to VMs. This leverages VXLAN and OpenFlow rules to intercept the DHCP request and respond with a DHCP response.

Here we show an example DHCP request using the Nutanix IPAM solution where the Acropolis Leader is running locally:

IPAM - Local Acropolis Leader

If the Acropolis Leader is running remotely, the same VXLAN tunnel will be leveraged to handle the request over the network.

IPAM - Remote Acropolis Leader

Traditional DHCP / IPAM solutions can also be leveraged in an ‘unmanaged’ network scenario.

VM High Availability (HA)

AHV VM HA is a feature built to ensure VM availability in the event of a host or block outage. In the event of a host failure the VMs previously running on that host will be restarted on other healthy nodes throughout the cluster. The Acropolis Leader is responsible for restarting the VM(s) on the healthy host(s).

The Acropolis Leader tracks host health by monitoring its connections to the libvirt on all cluster hosts:

HA - Host Monitoring

Once the libvirt connection goes down, the countdown to the HA restart is initiated. Should libvirt connection fail to be re-established within the timeout, Acropolis will restart VMs that were running on the disconnected host. When this occurs, VMs should be restarted within 120 seconds.

In the event the Acropolis Leader becomes partitioned, isolated or fails a new Acropolis Leader will be elected on the healthy portion of the cluster. If a cluster becomes partitioned (e.g X nodes can't talk to the other Y nodes) the side with quorum will remain up and VM(s) will be restarted on those hosts.

There are two main modes for VM HA:

- Default
  - This mode requires no configuration and is included by default when installing an AHV-based Nutanix cluster. When an AHV host becomes unavailable, the VMs that were running on the failed AHV host restart on the remaining hosts, depending on the available resources. Not all of the failed VMs restart if the remaining hosts do not have sufficient resources.

- Guarantee
  - This nondefault configuration reserves space throughout the AHV hosts in the cluster to guarantee that all failed VMs can restart on other hosts in the AHV cluster during a host failure. To enable Guarantee mode, select the Enable HA check box, as shown in the figure below. A message then appears displaying the amount of memory reserved and how many AHV host failures can be tolerated.

Resource Reservations

When using the Guarantee mode for VM HA, the system will reserve host resources for VMs. The amount of resources which are reserved is summarized by the following:

- If all containers are RF2 (FT1)
  - One "host" worth of resources

- If any containers are RF3 (FT2)
  - Two "hosts" worth of resources

When hosts have uneven memory capacities the system will use the largest host's memory capacity when determining how much to reserve per host.

Post 5.0 Resource Reservations

Prior to 5.0, we supported both host and segment based reserevations. With 5.0 and later we now only support a segment based reservation which is automatically implemented when the Guarantee HA mode is selected.

Reserve segments distributes the resource reservation across all hosts in a cluster. In this scenario, each host will share a portion of the reservation for HA. This ensures the overall cluster has enough failover capacity to restart VM(s) in the event of a host failure.

The figure shows an example scenario with reserved segments:

HA - Reserved Segment

In the event of a host failure VM(s) will be restarted throughout the cluster on the remaining healthy hosts:

HA - Reserved Segment - Fail Over

Reserved segment(s) calculation

The system will automatically calculate the total number of reserved segments and per host reservation.

Finding reservations reduces to a well known set of problems called Knapsack. The optimal solution is NP-hard (exponential), but heuristic solutions can come close to optimal for the common case. We implement one such algorithm called MTHM. Nutanix will continue improving its placement algorithms.

AHV Administration

» Download this section as PDF (opens in a new tab/window)

Command Reference

Enable 10GbE links only on OVS

Description: Enable 10g only on bond0 for local host

```
manage_ovs --interfaces 10g update_uplinks
```

Description: Show ovs uplinks for full cluster

```
allssh "manage_ovs --interfaces 10g update_uplinks"
```

Show OVS uplinks

Description: Show ovs uplinks for local host

```
manage_ovs show_uplinks
```

Description: Show ovs uplinks for full cluster

```
allssh "manage_ovs show_uplinks"
```

Show OVS interfaces

Description: Show ovs interfaces for local host

```
manage_ovs show_interfaces
```

Show interfaces for full cluster

```
allssh "manage_ovs show_interfaces"
```

Show OVS switch information

Description: Show switch information

```
ovs-vsctl show
```

List OVS bridges

Description: List bridges

```
ovs-vsctl list br
```

Show OVS bridge information

Description: Show OVS port information

```
ovs-vsctl list port br0
ovs-vsctl list port bond
```

Show OVS interface information

Description: Show interface information

```
ovs-vsctl list interface br0
```

Show ports / interfaces on bridge

Description: Show ports on a bridge

```
ovs-vsctl list-ports br0
```

Description: Show ifaces on a bridge

```
ovs-vsctl list-ifaces br0
```

Create OVS bridge

Description: Create bridge

```
ovs-vsctl add-br bridge
```

Add ports to bridge

Description: Add port to bridge

```
ovs-vsctl add-port bridge port
```

Description: Add bond port to bridge

```
ovs-vsctl add-bond bridge port iface
```

Show OVS bond details

Description: Show bond details

```
ovs-appctl bond/show bond
```

Example:

```
ovs-appctl bond/show bond0
```

Set bond mode and configure LACP on bond

Description: Enable LACP on ports

```
ovs-vsctl set port bond lacp=active/passive
```

Description: Enable on all hosts for bond0

```
for i in `hostips`;do echo $i; ssh $i source /etc/profile > /dev/null 2>&1; ovs-vsctl set port bond0 lacp=active;done
```

Show LACP details on bond

Description: Show LACP details

```
ovs-appctl lacp/show bond
```

Set bond mode

Description: Set bond mode on ports

Show OpenFlow information

Description: Show OVS openflow details

```
ovs-ofctl show br0
```

Description: Show OpenFlow rules

```
ovs-ofctl dump-flows br0
```

Get QEMU PIDs and top information

Description: Get QEMU PIDs

```
ps aux | grep qemu | awk '{print $2}'
```

Description: Get top metrics for specific PID

```
top -p PID
```

Get active Stargate for QEMU processes

Description: Get active Stargates for storage I/O for each QEMU processes

```
netstat –np | egrep tcp.*qemu
```

Metrics and Thresholds

More coming soon!

Troubleshooting & Advanced Administration

Check iSCSI Redirector Logs

Description: Check iSCSI Redirector Logs for all hosts

```
for i in `hostips`; do echo $i; ssh root@$i cat /var/log/iscsi_redirector;done
```

Example for single host

```
Ssh root@HOSTIP
Cat /var/log/iscsi_redirector
```

Monitor CPU steal (stolen CPU)

Description: Monitor CPU steal time (stolen CPU)

Launch top and look for %st (bold below)

```
Cpu(s):  0.0%us, 0.0%sy,  0.0%ni, 96.4%id,  0.0%wa,  0.0%hi,  0.1%si,  **0.0%st**
```

Monitor VM network resource stats

Description: Monitor VM resource stats

Launch virt-top

```
Virt-top
```

Book of vSphere

» Download this section as PDF (opens in a new tab/window)

Nutanix supports VMware ESXi, meaning you can run your virtual environment on vSphere while taking advantage of the Nutanix Distributed Storage Fabric. Nutanix also supports creating and managing ESXi VMs directly from Prism, providing a single management pane for managing your VMware virtual infrastructure.

This book will dive into the technical details about Nutanix running on VMware ESXi.

VMware vSphere Architecture

» Download this section as PDF (opens in a new tab/window)

Node Architecture

In ESXi deployments, the Controller VM (CVM) runs as a VM and disks are presented using VMDirectPath I/O. This allows the full PCI controller (and attached devices) to be passed through directly to the CVM and bypass the hypervisor.

ESXi Node Architecture

Configuration Maximums and Scalability

The following configuration maximums and scalability limits are applicable:

- Maximum cluster size: **48**
- Maximum vCPUs per VM: **128**
- Maximum memory per VM: **6TB**
- Maximum virtual disk size: **62TB**
- Maximum VMs per host: **1,024**
- Maximum VMs per cluster: **8,000 (2,048 per datastore if HA is enabled)**

NOTE: As of vSphere 6.5U1 and AOS 6.6. Refer to AOS Configuration Maximums and ESX Configuration Maximums for other versions.

Networking

Each ESXi host has a local vSwitch which is used for intra-host communication between the Nutanix CVM and host. For external communication and VMs a standard vSwitch (default) or dvSwitch is leveraged.

The local vSwitch (vSwitchNutanix) is for local communication between the Nutanix CVM and ESXi host. The host has a vmkernel interface on this vSwitch (vmk1 - 192.168.5.1) and the CVM has an interface bound to a port group on this internal switch (svm-iscsi-pg - 192.168.5.2). This is the primary storage communication path.

The external vSwitch can be a standard vSwitch or a dvSwitch. This will host the external interfaces for the ESXi host and CVM as well as the port groups leveraged by VMs on the host. The external vmkernel interface is leveraged for host management, vMotion, etc. The external CVM interface is used for communication to other Nutanix CVMs. As many port groups can be created as required assuming the VLANs are enabled on the trunk.

The following figure shows a conceptual diagram of the vSwitch architecture:

ESXi vSwitch Network Overview

Uplink and Teaming policy

It is recommended to have dual ToR switches and uplinks across both switches for switch HA. By default the system will have uplink interfaces in active/passive mode. For upstream switch architectures that are capable of having active/active uplink interfaces (e.g. vPC, MLAG, etc.) that can be leveraged for additional network throughput.

How vSphere on Nutanix Works

» Download this section as PDF (opens in a new tab/window)

Array Offloads – VAAI

The Nutanix platform supports the VMware APIs for Array Integration (VAAI), which allows the hypervisor to offload certain tasks to the array. This is much more efficient as the hypervisor doesn’t need to be the ‘man in the middle’. Nutanix currently supports the VAAI primitives for

NAS, including the ‘full file clone’, ‘fast file clone’, and ‘reserve space’ primitives. Here’s a good article explaining the various primitives: http://cormachogan.com/2012/11/08/vaai-comparison-block-versus-nas/.

For both the full and fast file clones, a DSF ‘fast clone’ is done, meaning a writable snapshot (using re-direct on write) for each clone that is created. Each of these clones has its own block map, meaning that chain depth isn’t anything to worry about. The following will determine whether or not VAAI will be used for specific scenarios:

- Clone VM with Snapshot –> VAAI will NOT be used
- Clone VM without Snapshot which is Powered Off –> VAAI WILL be used
- Clone VM to a different Datastore/Container –> VAAI will NOT be used
- Clone VM which is Powered On –> VAAI will NOT be used

These scenarios apply to VMware View:

- View Full Clone (Template with Snapshot) –> VAAI will NOT be used
- View Full Clone (Template w/o Snapshot) –> VAAI WILL be used
- View Linked Clone (VCAI) –> VAAI WILL be used

You can validate VAAI operations are taking place by using the ‘NFS Adapter’ Activity Traces page.

CVM Autopathing aka Ha.py

In this section, I’ll cover how CVM ‘failures’ are handled (I’ll cover how we handle component failures in future update). A CVM ‘failure’ could include a user powering down the CVM, a CVM rolling upgrade, or any event which might bring down the CVM. DSF has a feature called autopathing where when a local CVM becomes unavailable, the I/Os are then transparently handled by other CVMs in the cluster. The hypervisor and CVM communicate using a private 192.168.5.0 network on a dedicated vSwitch (more on this above). This means that for all storage I/Os, these are happening to the internal IP addresses on the CVM (192.168.5.2). The external IP address of the CVM is used for remote replication and for CVM communication.

The following figure shows an example of what this looks like:

ESXi Host Networking

In the event of a local CVM failure, the local 192.168.5.2 addresses previously hosted by the local CVM are unavailable. DSF will automatically detect this outage and will redirect these I/Os to another CVM in the cluster over 10GbE. The re-routing is done transparently to the hypervisor and VMs running on the host. This means that even if a CVM is powered down, the VMs will still continue to be able to perform I/Os to DSF. Once the local CVM is back up and available, traffic will then seamlessly be transferred back and served by the local CVM.

The following figure shows a graphical representation of how this looks for a failed CVM:

ESXi Host Networking - Local CVM Down

VMware ESXi Administration

» Download this section as PDF (opens in a new tab/window)

VM Management

Core VM management operations can be done directly from Prism without using any hypervisor management interface. Once your Nutanix nodes are added to your vCenter instance and your vCenter Server is registered with your Nutanix cluster (Settings > vCenter Registration), you can perform the following operations directly through Prism:

- Creating, cloning, updating, and deleting VMs
- Creating and deleting NICs
- Attaching and deleting disks
- VM Power operations (on/off, reset, suspend, resume, guest shutdown, guest restart)
- Launch VM console
- Manage VM guest tools (VMware or Nutanix Guest Tools)

Command Reference

ESXi cluster upgrade

Description: Perform an automated upgrade of ESXi hosts using the CLI and custom offline bundle

- Upload upgrade offline bundle to a Nutanix CVM
- Log in to Nutanix CVM
- Perform upgrade

```
cluster --md5sum=bundle_checksum --bundle=/path/to/offline_bundle host_upgrade
```

Example

```
cluster --md5sum=bff0b5558ad226ad395f6a4dc2b28597 --bundle=/tmp/VMware-ESXi-5.5.0-1331820-depot.zip host_upgrade
```

Restart ESXi host services

Description: Restart each ESXi hosts services in a incremental manner

```
for i in `hostips`;do ssh root@$i "services.sh restart";done
```

Display ESXi host nics in ‘Up’ state

Description: Display the ESXi host’s nics which are in a ‘Up’ state

Display ESXi host 10GbE nics and status

Description: Display the ESXi host’s 10GbE nics and status

```
for i in `hostips`;do echo $i && ssh root@$i esxcfg-nics -l | grep ixgbe;done
```

Display ESXi host active adapters

Description: Display the ESXi host’s active, standby and unused adapters

```
for i in `hostips`;do echo $i &&  ssh root@$i "esxcli network vswitch standard policy failover get --vswitch-name vSwitch0";done
```

Display ESXi host routing tables

Description: Display the ESXi host’s routing tables

```
for i in `hostips`;do ssh root@$i 'esxcfg-route -l';done
```

Check if VAAI is enabled on datastore

Description: Check whether or not VAAI is enabled/supported for a datastore

```
vmkfstools -Ph /vmfs/volumes/Datastore Name
```

Set VIB acceptance level to community supported

Description: Set the vib acceptance level to CommunitySupported allowing for 3rd party vibs to be installed

```
esxcli software acceptance set --level CommunitySupported
```

Install VIB

Description: Install a vib without checking the signature

```
esxcli software vib install --viburl=/VIB directory/VIB name --no-sig-check
```

OR

```
esxcli software vib install --depoturl=/VIB directory/VIB name --no-sig-check
```

Check ESXi ramdisk space

Description: Check free space of ESXi ramdisk

```
for i in `hostips`;do echo $i; ssh root@$i 'vdf -h';done
```

Clear pynfs logs

Description: Clears the pynfs logs on each ESXi host

```
for i in `hostips`;do echo $i; ssh root@$i '> /pynfs/pynfs.log';done
```

Book of Hyper-V

» Download this section as PDF (opens in a new tab/window)

Nutanix supports Microsoft Hyper-V, meaning you can run your virtual environment on Hyper-V while taking advantage of the Nutanix Distributed Storage Fabric.

This book will dive into the technical details about Nutanix running on Hyper-V.

Microsoft Hyper-V Architecture

» Download this section as PDF (opens in a new tab/window)

When a Nutanix Hyper-V cluster is created we automatically join the Hyper-V hosts to the specified Windows Active Directory domain. These hosts are then put into a failover cluster for VM HA. When this is complete there will be AD objects for each individual Hyper-V host and the failover cluster.

Node Architecture

In Hyper-V deployments, the Controller VM (CVM) runs as a VM and disks are presented using disk passthrough.

Hyper-V Node Architecture

Configuration Maximums and Scalability

The following configuration maximums and scalability limits are applicable:

- Maximum cluster size: **16**
- Maximum vCPUs per VM: **64**
- Maximum memory per VM: **1TB**
- Maximum virtual disk size: **64TB**
- Maximum VMs per host: **1,024**
- Maximum VMs per cluster: **8,000**

NOTE: As of Hyper-V 2012 R2 and AOS 6.6. Refer to Configuration Maximums for other versions.

Networking

Each Hyper-V host has a internal only virtual switch which is used for intra-host communication between the Nutanix CVM and host. For external communication and VMs a external virtual switch (default) or logical switch is leveraged.

The internal switch (InternalSwitch) is for local communication between the Nutanix CVM and Hyper-V host. The host has a virtual ethernet interface (vEth) on this internal switch (192.168.5.1) and the CVM has a vEth on this internal switch (192.168.5.2). This is the primary storage communication path.

The external vSwitch can be a standard virtual switch or a logical switch. This will host the external interfaces for the Hyper-V host and CVM as well as the logical and VM networks leveraged by VMs on the host. The external vEth interface is leveraged for host management, live migration, etc. The external CVM interface is used for communication to other Nutanix CVMs. As many logical and VM networks can be created as required assuming the VLANs are enabled on the trunk.

The following figure shows a conceptual diagram of the virtual switch architecture:

Hyper-V Virtual Switch Network Overview

Uplink and Teaming policy

It is recommended to have dual ToR switches and uplinks across both switches for switch HA. By default the system will have the LBFO team in switch independent mode which doesn't require any special configuration.

How Hyper-V on Nutanix Works

» Download this section as PDF (opens in a new tab/window)

Array Offloads – ODX

The Nutanix platform supports the Microsoft Offloaded Data Transfers (ODX), which allow the hypervisor to offload certain tasks to the array. This is much more efficient as the hypervisor doesn't need to be the 'man in the middle'. Nutanix currently supports the ODX primitives for SMB, which include full copy and zeroing operations. However, contrary to VAAI which has a 'fast file' clone operation (using writable snapshots), the ODX primitives do not have an equivalent and perform a full copy. Given this, it is more efficient to rely on the native DSF clones which can currently be invoked via nCLI, REST, or PowerShell CMDlets. Currently ODX IS invoked for the following operations:

- In VM or VM to VM file copy on DSF SMB share
- SMB share file copy

Deploy the template from the SCVMM Library (DSF SMB share) – NOTE: Shares must be added to the SCVMM cluster using short names (e.g., not FQDN). An easy way to force this is to add an entry into the hosts file for the cluster (e.g. 10.10.10.10   nutanix-130).

ODX is NOT invoked for the following operations:

- Clone VM through SCVMM
- Deploy template from SCVMM Library (non-DSF SMB Share)
- XenDesktop Clone Deployment

You can validate ODX operations are taking place by using the “NFS Adapter” Activity Traces page (even though this is being performed via SMB). The operations activity show will be “NfsWorkerVaaiCopyDataOp” when copying a vDisk and “NfsWorkerVaaiWriteZerosOp” when zeroing out a disk.

Microsoft Hyper-V Administration

» Download this section as PDF (opens in a new tab/window)

Important Pages

More coming soon!

Command Reference

Execute command on multiple remote hosts

Description: Execute a PowerShell on one or many remote hosts

```
$targetServers = "Host1","Host2","Etc"
Invoke-Command -ComputerName  $targetServers {

}
```

Check available VMQ Offloads

Description: Display the available number of VMQ offloads for a particular host

```
gwmi -Namespace "root\virtualization\v2" -Class Msvm_VirtualEthernetSwitch | select elementname, MaxVMQOffloads
```

Disable VMQ for VMs matching a specific prefix

Description: Disable VMQ for specific VMs

```
$vmPrefix = "myVMs"
Get-VM | Where {$_.Name -match $vmPrefix} | Get-VMNetworkAdapter | Set-VMNetworkAdapter -VmqWeight 0
```

Enable VMQ for VMs matching a certain prefix

Description: Enable VMQ for specific VMs

```
$vmPrefix = "myVMs"
Get-VM | Where {$_.Name -match $vmPrefix} | Get-VMNetworkAdapter | Set-VMNetworkAdapter -VmqWeight 1
```

Power-On VMs matching a certain prefix

Description: Power-On VMs matching a certain prefix

Shutdown VMs matching a certain prefix

Description: Shutdown VMs matching a certain prefix

```
$vmPrefix = "myVMs"
Get-VM | Where {$_.Name -match $vmPrefix -and $_.StatusString -eq "Running"}} | Shutdown-VM -RunAsynchronously
```

Stop VMs matching a certain prefix

Description: Stop VMs matching a certain prefix

```
$vmPrefix = "myVMs"
Get-VM | Where {$_.Name -match $vmPrefix} | Stop-VM
```

Get Hyper-V host RSS settings

Description: Get Hyper-V host RSS (recieve side scaling) settings

```
Get-NetAdapterRss
```

Check Winsh and WinRM connectivity

Description: Check Winsh and WinRM connectivity / status by performing a sample query which should return the computer system object not an error

```
allssh "winsh "get-wmiobject win32_computersystem"
```

Metrics and Thresholds

More coming soon!

Troubleshooting & Advanced Administration

More coming soon!

Part 2: Services

Book of Nutanix Clusters (NC2)

» Download this section as PDF (opens in a new tab/window)

Leveraging the public cloud means being able to scale your applications and data on-demand within minutes. Nutanix Cloud Clusters (NC2) allows you to use a single management plane to manage both your Nutanix private cloud and your public cloud resources in the same Prism interface, allowing for a seamless experience.

This book will go into the technical details of the architecture, networking, storage, and other aspects of running NC2.

NC2 is currently supported on AWS and Azure.

Nutanix Cloud Clusters on AWS

» Download this section as PDF (opens in a new tab/window)

Nutanix Cloud Clusters (NC2) on AWS provides on-demand clusters running in target cloud environments using bare metal resources. This allows for true on-demand capacity with the simplicity of the Nutanix platform you know. Once provisioned the cluster appears like any traditional AHV cluster, just running in a cloud providers datacenters.

Supported Configurations

The solution is applicable to the configurations below (list may be incomplete, refer to documentation for a fully supported list):

Core Use Case(s):

- On-Demand / burst capacity
- Backup / DR
- Cloud Native
- Geo Expansion / DC consolidation
- App migration
- Etc.

Management interfaces(s):

- Nutanix Clusters Portal - Provisioning
- Prism Central (PC) - Nutanix Management
- AWS Console - AWS Management

Supported Environment(s):

- Cloud:
  - AWS
  - Azure

- EC2 Metal Instance Types:
  - i3.metal
  - m5d.metal
  - z1d.metal
  - i3en.metal
  - g4dn.metal
  - i4i.metal
  - m6d.metal

Upgrades:

- Part of AOS

Compatible Features:

- AOS Features
- AWS Services

Key terms / Constructs

The following key items are used throughout this section and defined in the following:

- Nutanix Clusters Portal
  - The Nutanix Clusters Portal is responsible for handling cluster provisioning requests and interacting with AWS and the provisioned hosts. It creates cluster specific details and handles the dynamic CloudFormation stack creation.

- Region
  - A geographic landmass or area where multiple Availability Zones (sites) are located. A region can have two or more AZs. These can include regions like US-East-1 or US-West-1.

- Availability Zone (AZ)
  - An AZ consists of one or more discrete datacenters inter-connected by low latency links. Each site has it’s own redundant power, cooling, network, etc. Comparing these to a traditional colo or datacenter, these would be considered more resilient as a AZ can consist of multiple independent datacenters. These can include sites like US-East-1a or US-West-1a.

- VPC
  - A logically isolated segment of the AWS cloud for tenants. Provides a mechanism to to secure and isolate environment from others. Can be exposed to the internet or other private network segments (other VPCs, or VPNs).

- S3
  - Amazon’s object service which provides persistent object storage accessed via the S3 API. This is used for archival / restore.

- EBS
  - Amazon’s volume / block service which provides persistent volumes that can be attached to AMIs.

- Cloud Formation Template (CFT)
  - A Cloud Formation Template simplifies provisioning, but allowing you to define a “stack” of resources and dependencies. This stack can then be provisioned as a whole instead of each individual resource.

Cluster Architecture

From a high-level the Nutanix Clusters Portal is the main interface for provisioning Nutanix Clusters on AWS and interacting with AWS.

The provisioning process can be summarized with the following high-level steps:

1. Create cluster in NC2 Portal
2. Deployment specific inputs (e.g. Region, AZ, Instance type, VPC/Subnets, etc.)
3. The NC2 Portal creates associated resources
4. Host agent in Nutanix AMI checks-in with Nutanix Clusters on AWS
5. Once all hosts as up, cluster is created

The following shows a high-level overview of the NC2A interaction:

NC2A - Overview

The following shows a high-level overview of a the inputs taken by the NC2 Portal and some created resources:

Nutanix Clusters on AWS - Cluster Orchestrator Inputs

The following shows a high-level overview of a node in AWS:

NC2A - Node Architecture

Given the hosts are bare metal, we have full control over storage and network resources similar to a typical on-premises deployment. For the CVM and AHV host boot, EBS volumes are used. NOTE: certain resources like EBS interaction run through the AWS Nitro card which appears as a NVMe controller in the AHV host.

Placement policy

Nutanix uses a partition placement strategy when deploying nodes inside an AWS Availability Zone. One Nutanix cluster can't span different Availability Zones in the same Region, but you can have multiple Nutanix clusters replicating between each other in different zones or Regions. Using up to seven partitions, Nutanix places the AWS bare-metal nodes in different AWS racks and stripes new hosts across the partitions.

NC2 on AWS supports combining heterogenous node types in a cluster. You can deploy a cluster of one node type and then expand that cluster's capacity by adding heterogenous nodes to it. This feature protects your cluster if its original node type runs out in the Region and provides flexibility when expanding your cluster on demand. If you're looking to right-size your storage solution, support for heterogenous nodes can give you more instance options to choose from.

When combining instance types in a cluster, you must always maintain at least three nodes of the original type you deployed the base cluster with. You can expand or shrink the base cluster with any number of heterogenous nodes if at least three nodes of the original type remain and the cluster size stays within the limit of 28 nodes.

The following table and figure both refer to the cluster's original instance type as Type A and its compatible heterogenous type as Type B.

Table: Supported Instance Type Combinations

Type A	Type B
i3.metal	i3en.metal
i3en.metal	i3.metal
z1d.metal	m5d.metal
m5d.metal	z1d.metal

NC2A - Partition Placement

When you’ve formed the Nutanix cluster, the partition groups map to the Nutanix rack-awareness feature. AOS Storage writes data replicas to other racks in the cluster to ensure that the data remains available for both replication factor 2 and replication factor 3 scenarios in the case of a rack failure or planned downtime.

Storage

Storage for Nutanix Cloud Clusters on AWS can be broken down into two core areas:

1. Core / Active
2. Hibernation

Core storage is the exact same as you’d expect on any Nutanix cluster, passing the “local” storage devices to the CVM to be leveraged by Stargate.

Instance Storage

Given that the "local" storage is backed by the AWS instance store, which isn't fully resilient in the event of a power outage / node failure additional considerations must be handled.

For example, in a local Nutanix cluster in the event of a power outage or node failure, the storage is persisted on the local devices and will come back when the node / power comes back online. In the case of the AWS instance store, this is not the case.

In most cases it is highly unlikely that a full AZ will lose power / go down, however for sensitive workloads it is recommended to:

- Leverage a backup solution to persist to S3 or any durable storage
- Replicate data to another Nutanix cluster in a different AZ/Region/Cloud (on-prem or remote)

One unique ability with NC2A is the ability to “hibernate” a cluster allowing you to persist the data while spinning down the EC2 compute instances. This could be useful for cases where you don’t need the compute resources and don’t want to continue paying for them, but want to persist the data and have the ability to restore at a later point.

When a cluster is hibernated, the data will be backed up from the cluster to S3. Once the data is backed up the EC2 instances will be terminated. Upon a resume / restore, new EC2 instances will be provisioned and data will be loaded into the cluster from S3.

Networking

Networking can be broken down into a few core areas:

- Host / Cluster Networking
- Guest / UVM Networking
- WAN / L3 Networking

Native vs. Overlay

Instead of running our own overlay network, we decided to run natively on AWS subnets, this allows VMs running on the platform to natively communicate with AWS services with zero performance degradation.

NC2A are provisioned into an AWS VPC, the following shows a high-level overview of an AWS VPC:

NC2A - AWS VPC

New vs. Default VPC

AWS will create a default VPC/Subnet/Etc. with a 172.31.0.0/16 ip scheme for each region.

It is recommended to create a new VPC with associated subnets, NAT/Internet Gateways, etc. that fits into your corporate IP scheme. This is important if you ever plan to extend networks between VPCs (VPC peering), or to your existing WAN. This should be treated as you would treat any site on the WAN.

Host Networking

The hosts running on baremetal in AWS are traditional AHV hosts, and thus leverage the same OVS based network stack.

The following shows a high-level overview of a AWS AHV host’s OVS stack:

NC2A - OVS Architecture

The OVS stack is relatively the same as any AHV host except for the addition of the L3 uplink bridge.

For UVM (Guest VM) networking, VPC subnets are used. A UVM network can be created during the cluster creation process or via the following steps:

From the AWS VPC dashboard, click on ‘subnets’ then click on ‘Create Subnet’ and input the network details:

NC2A - OVS Architecture

NOTE: the CIDR block should be a subset of the VPC CIDR range.

The subnet will inherit the route table from the VPC:

NC2A - Route Table

In this case you can see any traffic in the peered VPC will go over the VPC peering link and any external traffic will go over the internet gateway.

Once complete, you will see the network is available in Prism.

WAN / L3 Networking

In most cases deployments will not be just in AWS and will need to communicate with the external world (Other VPCs, Internet or WAN).

For connecting VPCs (in the same or different regions), you can use VPC peering which allows you to tunnel between VPCs. NOTE: you will need to ensure you follow WAN IP scheme best practices and there are no CIDR range overlaps between VPCs / subnets.

The following shows a VPC peering connection between a VPC in the eu-west-1 and eu-west-2 regions:

NC2A - VPC Peering

The route table for each VPC will then route traffic going to the other VPC over the peering connection (this will need to exist on both sides if communication needs to be bi-directional):

NC2A - Route Table

For network expansion to on-premises / WAN, either a VPN gateway (tunnel) or AWS Direct Connect can be leveraged.

Security

Given these resources are running in a cloud outside our full control security, data encryption and compliance is a very critical consideration.

The recommendations can be characterized with the following:

- Enable data encryption
- Only use private subnets (no public IP assignment)
- Lock down security groups and allowed ports / IP CIDR blocks
- For more granular security, leverage Flow

AWS Security Groups

With AWS security groups, you can limit access to the AWS CVMs, AHV host, and UVMs only from your on-premises management network and CVMs. You can control replication from on-premises to AWS down to the port level, and you can easily migrate workloads because replication software is embedded in the CVMs on both ends.

NC2 can help you save on the cost of additional compute that overlay networks require. You can also avoid the costs for management gateways, network controllers, edge devices, and storage incurred from adding appliances. A simpler system offers significant operational savings on maintenance and troubleshooting.

AOS 6.7 adds support for custom AWS Security Groups. Prior to this update, two main security groups provided native protection. The new enhancement provides additional flexibility so AWS security groups can apply to the Virtual Private Cloud (VPC) domain and at the cluster and subnet levels.

Custom AWS Security Groups are applied when the ENI is attached to the bare-metal host. You can use and re-use pre-created Security Groups across different clusters without additional scripting to maintain and support the prior custom Security Groups.

You can use the tags in the following list with any AWS Security Group, including custom Security Groups. The Cloud Network Service (CNS) uses these tags to evaluate which Security Groups to attach to the network interfaces. The CNS is a distributed service that runs in the CVM and provides cloud-specific back-end support for subnet management, IP address event handling, and security group management. The following list is arranged in dependency order.

- **Scope**: VPC
  - **Key**: tag:nutanix:clusters:external
  - **Value**: <none> (leave this tag blank)

You can use this tag to protect multiple clusters in the same VPC.

- **Scope**: VPC or cluster
  - **Key**: tag:nutanix:clusters:external:cluster-uuid

    - **Value**: <cluster-uuid>

This tag protects all the UVMs and interfaces that the CVM and AHV use.

- **Scope:** VPC, cluster, network, or subnet
    - **Key:** tag:nutanix:clusters:external:networks
    - **Value:** <cidr1, cidr2, cidr3>

This tag only protects the subnets you provide.

If you want to apply a tag based on the subnet or CIDR, you need to set both external and cluster-uuid for the network or subnet tag to be applied. The following subsections provide configuration examples.

Default Security Groups

The red lines in the preceding figure represent the standard AWS Security Groups that deploy with the cluster.

- Internal management Security Group: Allows all internal traffic between all CVMs and AHV hosts (EC2 bare-metal hosts). Don’t edit this group without approval from Nutanix Support.
- User management Security Group: Allows users to access Prism Element and other services running on the CVM.
- UVM Security Group: Allows UVMs to talk to each other. By default, all UVMs on all subnets can talk to each other. This Security Group doesn’t offer subnet granularity.

VPC Level

The green line in the preceding figure represents the VPC-level tag protecting Cluster 1 and Cluster 2.

Cluster Level

The green line in the preceding figure represents the cluster-level tag. Changes to these Security Groups affect the management subnet and all the UVMs running in Cluster 1.

Network Level

This network-level custom Security Group covers just the database subnet, as shown by the green line in the preceding figure. To cover the Files subnet with this Security Group, simply change the tag as follows:

- tag:nutanix:clusters:external:networks, Value: 10.72.50.0/24, 10.73.55.0/24

Usage and Configuration

The following sections cover how to configure and leverage NC2A.

The high-level process can be characterized into the following high-level steps:

1. Create AWS Account(s)
2. Configure AWS network resources (if necessary)
3. Provision cluster(s) via Nutanix Clusters Portal
4. Leverage cluster resources once provisioning is complete
5. Protect your cluster

Native Backup with Nutanix Cluster Protection

Even when you migrate your application to the cloud, you still must provide all of the same day-two operations as you would if the application was on-premises. Nutanix Cluster Protection provides a native option for backing up Nutanix Cloud Clusters (NC2) running on AWS to S3 buckets, including user data and Prism Central with its configuration data. Cluster Protection backs up all user-created VMs and volume groups on the cluster.

As you migrate from on-premises to the cloud, you can be sure that there is another copy of your applications and data in the event of an AWS Availability Zone (AZ) failure. Nutanix already provides native protection for localized failures at the node and rack level, and Cluster Protection extends that protection to the cluster's AZ. Because this service is integrated, high-performance applications are minimally affected as the backup process uses native AOS snapshots to send the backup directly to S3.

Two Nutanix services help deliver Cluster Protection:

- Prism Central Disaster Recovery is responsible for backing up the Prism Central data. Instead of backing up to an AOS storage container, you can now supply a new S3 bucket to point the backup to.
- Nutanix Multicloud Snapshot Technology (NMST) replicates native Nutanix AOS snapshots to object storage. In the Cluster Protection design, the you supply a second new S3 bucket in AWS to send all the protected clusters' snapshots to the same S3 bucket. The Cloud Snapshot Engine runs on the Prism Central instance in AWS.

The following high-level process describes how to protect your clusters and Prism Central in AWS.

- Deploy a one- or three-node Prism Central instance in AWS.
- Create two S3 Buckets: one for Prism Central and one for your cloud clusters.
- Enable Prism Central protection.
- Deploy NMST.
- Protect your AWS cloud clusters.

The system takes both the Prism Central and AOS snapshots every hour and retains up to two snapshots in S3. A Nutanix Disaster Recovery category protects all of the user-created VMs and volume groups on the clusters. A service watches for create or delete events and assigns them a Cluster Protection category.

The following high-level process describes how to recover your Prism Central instances and clusters on AWS.

- The NC2 console automatically deploys a new NC2 cluster during the recovery process.
- Add your Prism Central subnet and any user VM networks to your recreated cloud cluster.
- Recover your Prism Central configuration from the S3 bucket.
- Register your Prism Central instance with the recovered cluster.
- Recover NMST.
- Create a recovery plan.
- Run the recovery plan from Prism Central.

Once the NMST is recovered, you can restore using the recovery plan in Prism Central. The recovery plan has all the VMs you need to restore. By using Nutanix Disaster Recovery with this new service, administrators can easily recover when disaster strikes.

Nutanix Cloud Clusters on Azure

» Download this section as PDF (opens in a new tab/window)

Nutanix Cloud Clusters (NC2) on Azure provides on-demand clusters running in target cloud environments using bare metal resources. This allows for true on-demand capacity with the simplicity of the Nutanix platform you know. Once provisioned the cluster appears like any traditional AHV cluster, just running in a cloud provider's datacenter(s).

Supported Configurations

The solution is applicable to the configurations below (list may be incomplete, refer to documentation for a fully supported list):

Core Use Case(s):

- On-Demand / burst capacity
- Backup / DR
- Cloud Native
- Geo Expansion / DC consolidation
- App migration
- Etc.

Management interfaces(s):

- Nutanix Clusters Portal - Provisioning
- Prism Central (PC) - Nutanix Management
- Azure Portal - Azure Management

Supported Environment(s):

- Cloud:
  - AWS
  - Azure
- Bare Metal Instance Types:
  - AN36
  - AN36P

Upgrades:

- Part of AOS

Compatible Features:

- AOS Features
- Azure Services

Key terms / Constructs

The following key items are used throughout this section and defined in the following:

- Nutanix Clusters Portal
  - The Nutanix Clusters Portal is responsible for handling cluster provisioning requests and interacting with Azure and the provisioned hosts. It creates cluster specific details and handles the cluster creation and helps to remediate hardware problems.
- Region
  - A geographic landmass or area where multiple Availability Zones (sites) are located. A region can have two or more AZs. These can include regions like East US (Virginia) or West US 2 (Washington).

- Availability Zone (AZ)
  - An AZ consists of one or more discrete datacenters interconnected by low latency links. Each site has its own redundant power, cooling, network, etc. Comparing these to a traditional colo or datacenter, these would be considered more resilient as a AZ can consist of multiple independent datacenters.

- VNet
  - A logically isolated segment of the Azure cloud for tenants. Provides a mechanism to secure and isolate environment from others. Can be exposed to the internet or other private network segments (other VNets, or VPNs).

Cluster Architecture

From a high-level the Nutanix Clusters (NC2) Portal is the main interface for provisioning Nutanix Clusters on Azure and interacting with Azure.

The provisioning process can be summarized with the following high-level steps:

1. Create cluster in NC2 Portal
2. Deployment specific inputs (e.g. Region, AZ, Instance type, VNets/Subnets, etc.)
3. The NC2 Portal creates associated resources
4. Host agent running on AHV checks-in with Nutanix Clusters on Azure
5. Once all hosts as up, cluster is created

The following shows a high-level overview of the NC2 on Azure interaction:

NC2 on Azure - Overview

The following shows a high-level overview of a the inputs taken by the NC2 Portal and some created resources:

Nutanix Clusters on Azure - Cluster Orchestrator Inputs

Node Architecture

Given the hosts are bare metal, we have full control over storage and network resources similar to a typical on-premises deployment. We are consuming Ready Nodes as our building blocks. Unlike AWS, Azure-based nodes are not consuming any additional services for the CVM or AHV.

Placement policy

Nutanix Clusters on Azure uses a partition placement policy with 7 partitions by default. Hosts are striped across these partitions which correspond with racks in Nutanix. This ensures you can have 1-2 full “rack” failures and still maintain availability.

The following shows a high-level overview of the partition placement strategy and host striping:

NC2 on Azure - Partition Placement

Storage

Core storage is the exact same as you’d expect on any Nutanix cluster, passing the “local” storage devices to the CVM to be leveraged by Stargate.

Instance Storage

Given that the "local" storage is backed by the local flash, it is fully resilient in the event of a power outage.

Networking

NC2 utilizes Flow Virtual Networking in Azure to create an overlay network to ease administration for Nutanix administrators and reduce networking constraints across Cloud vendors. Flow Virtual Networking is used to abstract the Azure native network by creating overlay virtual networks. On the one hand this abstracts the underlying network in Azure, while at the same time, it allows the network substrate (and its associated features and functionalities) to be consistent with the customer’s on-premises Nutanix deployments. You will be able to create new virtual networks (called Virtual Private Clouds or VPCs) within Nutanix, subnets in any address range, including those from the RFC1918 (private) address space and define DHCP, NAT, routing, and security policy right from the familiar Prism Central interface.

Flow Virtual Networking can mask or reduce Cloud constraints by providing an abstraction layer. As an example, Azure only allows for one delegated subnet per VNet. Subnet delegation enables you to designate a specific subnet for an Azure PaaS service of your choice that needs to be injected into your virtual network. NC2 needs a management subnet delegated to the Microsoft.BareMetal/AzureHostedService. Once your subnet is delegated to the BareMetal service the Clusters Portal will be able to use that subnet to deploy your Nutanix Cluster. The AzureHostedService is what the Clusters portal uses to deploy and configure networking on the bare-metal nodes.

Every subnet used for user native VM networking also needs to be delegated to the same service. Since a VNet can only have one delegated subnet, networking configuration would get out of hand with needing to peer VNets among each other to allow communication. With Flow

Virtual Networking we can drastically reduce the amount of VNets needed to allow communication of the workloads running on Clusters and Azure. Flow Virtual Networking will allow you to create over 500 subnets while only consuming 1 Azure VNet.

It is recommended to create a new VPC with associated subnets, NAT/Internet Gateways, etc. that fits into your corporate IP scheme. This is important if you ever plan to extend networks between VPCs (VPC peering), or to your existing WAN. I treat this as I would any site on the WAN.

Prism Central (PC) will be deployed onto the Nutanix Cluster after deployment. Prism Central contains the control plane for Flow Virtual Networking. The subnet for PC will be delegated to the Microsoft.BareMetal/AzureHostedService so native Azure networking can be used to distribute IPs for PC. Once PC is deployed, the Flow Gateway will be deployed into the same subnet PC is using. The Flow Gateway allows the User VMs using the Flow VPC(s) to communicate to native Azure services and allows the VMs to have parity with native Azure VMs, such as:

- User defined routes - You can create custom, or user-defined (static), routes in Azure to override Azure's default system routes, or to add additional routes to a subnet's route table. In Azure, you create a route table, then associate the route table to zero or more virtual network subnets.
- Load Balancer Deployment - The ability to front-end services offered by UVMs with Azure-native load balancer.
- Network Security Groups - The ability to write stateful firewall policies.

The Flow Gateway VM is responsible for all VM traffic going north and south bound from the cluster. During deployment you can pick different sizes for the Flow Gateway VM based on how much bandwidth you need. It's important to realize that CVM replication between other CVMs and on-prem do not flow through the Flow Gateway VM so you don't have to size for that traffic.

Flow Virtual Networking Gateway VM High Availability

When you initially deploy your first cluster, you're able to choose how many FGW VMs you want to create (2 - 4). Prior to AOS 6.7, If you deploy only one gateway VM, the NC2 portal redeploys a new FGW VM with an identical configuration when it detects that the original VM is down. Because this process invokes various Azure APIs, it can take about 5 minutes before the new FGW VM is ready to forward traffic, which can affect the north-south traffic flow.

To reduce this downtime, NC2 on Azure has moved to an active-active configuration. This setup provides a flexible scale-out configuration when you need more traffic throughput.

The following workflow describes what happens when you turn off a FGW VM gracefully for planned events like updates.

1. The NC2 portal disables the FGW VM.
2. Prism Central removes the VM from the traffic path
3. The NC2 portal deletes the original VM and creates a new FGW VM with an identical configuration.
4. The NC2 portal registers the new instance with Prism Central.
5. Prism Central adds the new instance back to the traffic path.

For ungraceful or unplanned failures, the NC2 portal and Prism Central both have their own detection mechanisms based on keepalives. They take similar actions to those for the graceful or planned cases.

Network Address Translation (NAT): UVMs that want to communicate with AHV/CVM/PC and Azure resources will flow though the external network card on the Flow Gateway VM. The NAT provided uses native Azure address to ensure routing to all resources. User defined routes in Azure can be used to talk directly to Azure resources if using a NAT is not preferred. This allows for fresh installs to communicate with Azure right away but also gives customers options for more advanced configurations.

Each FGW instance has two NICs: one on the internal subnet that exchanges traffic with AHV and another on the external subnet that exchanges traffic with the Azure network. Each FGW instance registers with Prism Central and is added to the traffic path. A point-to-point external subnet is created for each FGW and the transit VPC is attached to it, with the FGW instance hosting the corresponding logical-router gateway port. In the following diagram, EN-NONAT1 and EN-NONAT2 are the point-to-point external subnets.

Flow Virtual Networking Gateway Using the NoNAT Path

For northbound traffic, the transit VPC has an equal-cost multi-path (ECMP) default route, with all the point-to-point external subnets as possible next hops. In this case, the transit VPC distributes traffic across multiple external subnets hosted on different FGWs.

For southbound traffic using more than one FGW, a Border Gateway Protocol (BGP) gateway is deployed as an Azure native VM instance in the Prism Central Virtual Network (VNet). Azure Route Servers deploy in the same Prism Central VNet. With an Azure Route Server, you can exchange routing information directly through BGP between any network virtual appliance that supports BGP and the Azure VNet without the need to manually configure or maintain route tables.

The BGP gateway peers with the Azure Route Servers. The BGP gateway advertises the externally routable IP addresses to the Azure Route Server with each active FGW external IP address as the next hop. Externally routable IP addresses compose the address range that you've created and want advertised to the rest of the network in your Flow Virtual Networking user VPCs. Once the externally routable IP address is set in Prism Central at the user VPC, the Azure network distributes southbound packets across all the FGW instances.

Prism Central determines which FGW instance should host a given NAT IP address and then configures each NAT IP address as a secondary IP address on each FGW. Packets sourced from those IP addresses can be forwarded through the corresponding FGW only. No NAT traffic distributes across all FGWs.

NAT traffic originating from the Azure network and destined to a floating IP address goes to the FGW VM that owns the IP address because Azure knows which NIC currently has it.

Prism Central uses policy-based routing to support forwarding based on the source IP address matching the floating IP address. The custom forwarding policy-based routing rules built into Flow Virtual Networking are used to auto install routes in the transit VPC.

Host Networking

The hosts running on baremetal in Azure are traditional AHV hosts, and thus leverage the same OVS based network stack.

The following shows a high-level overview of a Azure AHV host's OVS stack:

NC2 on Azure - Host Networking

Nutanix's Open vSwitch implementation is very similar to the on-premises implementation. The above diagrams shows an internal architecture of the AHV that is deployed onto the bare-metal. Br0 bridge will split traffic between br0.cluster (AHV/CVM IPs) and br0.uvms(User VMs IPs).

For AHV/CVM traffic via br0.cluster, it will be a simple pass-through to br0.azure bridge, with no modification to data packets. The top of rack switching is providing the security for br0.cluster traffic. For UVM IPs traffic will flow via br0.uvms, OVS rules would be installed for vlan-id translation and pass-through traffic to br0.azure.

br0.azure will have OVS bond br0.azure-up which will form a bonded interface with bare-metal attached physical nics. Thus, br0.azure hides the bonded interface from br0.uvms and br0.cluster.

Creating a Subnet

Subnets you create will have its own built in IPAM and you will have the option to stretch your network from on-prem into Azure. If outside applications need to talk directly your UVM inside the subnet you also have the option to assign floating IPs from a pool of IPs from Azure that will come from the external network of the Flow Gateway.

NC2 on Azure - IPAM with Azure

For a successful deployment, Nutanix Clusters needs outbound access to the NC2 portal, either using an NAT gateway or an on-prem VPN with outbound access. Your Nutanix cluster can sit in a private subnet that can only be accessed from your VPN, limiting exposure to your environment.

WAN / L3 Networking

In most cases deployments will not be just in Azure and will need to communicate with the external world (Other VNets, Internet or WAN).

For connecting VNets (in the same or different regions), you can use VPC peering which allows you to tunnel between VPCs. NOTE: you will need to ensure you follow WAN IP scheme best practices and there are no CIDR range overlaps between VNets / subnets.

For network expansion to on-premises / WAN, either a VNet gateway (tunnel) or Express Route can be leveraged.

Usage and Configuration

The following sections cover how to configure and leverage NC2 on Azure.

The high-level process can be characterized into the following high-level steps:

1. Setup up an active Azure subscription.
2. Create a My Nutanix account & subscribe to NC2.
3. Register Azure resource providers.
4. Create an app registration in Azure AD with “Contributor” access to the new subscription
5. Configure DNS.
6. Create a resource group or re-use an existing resource group.
7. Create required VNets and required subnets.
8. Configure two NAT gateways.
9. Establish the VNet peering required for the Nutanix cluster.
10. Add your Azure account to the NC2 console.
11. Create a Nutanix Cluster in Azure by using the NC2 console.

More to come!

Book of Storage Services

» Download this section as PDF (opens in a new tab/window)

At its inception, Nutanix was focused on providing optimized storage services for VM data. We then expanded this in order to allow any operating system to access the storage system with Nutanix Volumes, for use cases such as shared disks and guest-initiated iSCSI for bare metal consumers. With Nutanix Files, we expanded this further to provide the capability to use the platform as a highly available file server. Finally, Nutanix Objects provides highly scalable object storage via an S3 compliant API.

This book will cover each of these storage services in depth.

Volumes (Block Services)

» Download this section as PDF (opens in a new tab/window)

The Nutanix Volumes feature (previously know as Acropolis Volumes) exposes back-end DSF storage to external consumers (guest OS, physical hosts, containers, etc.) via iSCSI.

This allows any operating system to access DSF and leverage its storage capabilities. In this deployment scenario, the OS is talking directly to Nutanix bypassing any hypervisor.

Core use-cases for Volumes:

- Shared Disks
    - Oracle RAC, Microsoft Failover Clustering, etc.

- Disks as first-class entities
    - Where execution contexts are ephemeral and data is critical
    - Containers, OpenStack, etc.

- Guest-initiated iSCSI
    - Bare-metal consumers
    - Exchange on vSphere (for Microsoft Support)

Qualified Operating Systems

The solution is iSCSI spec compliant, the qualified operating systems are just those of which have been validated by QA.

- Microsoft Windows Server 2008 R2, 2012 R2
- Redhat Enterprise Linux 6.0+

Volumes Constructs

The following entities compose Volumes:

- **Data Services IP:** Cluster wide IP address used for iSCSI login requests (Introduced in 4.7)
- **Volume Group:** iSCSI target and group of disk devices allowing for centralized management, snapshotting, and policy application
- **Disk(s):** Storage devices in the Volume Group (seen as LUNs for the iSCSI target)
- **Attachment:** Allowing a specified initiator IQN access to the volume group
- **Secret(s):** Secret used for CHAP/Mutual CHAP authentication

NOTE: On the backend, a VG’s disk is just a vDisk on DSF.

Pre-Requisites

Before we get to configuration, we need to configure the Data Services IP which will act as our central discovery / login portal.

We’ll set this on the ‘Cluster Details’ page (Gear Icon -> Cluster Details):

Volumes - Data Services IP

This can also be set via NCLI / API:

```
ncli cluster edit-params external-data-
services-ip-address=DATASERVICESIPADDRESS
```

Target Creation

To use Volumes, the first thing we'll do is create a 'Volume Group' which is the iSCSI target.

From the 'Storage' page click on '+ Volume Group' on the right hand corner:

Volumes - Add Volume Group

This will launch a menu where we'll specify the VG details:

Volumes - Add VG Details

Next we'll click on '+ Add new disk' to add any disk(s) to the target (visible as LUNs):

A menu will appear allowing us to select the target container and size of the disk:

Volumes - Add Disk

Click 'Add' and repeat this for however many disks you'd like to add.

Once we've specified the details and added disk(s) we'll attach the Volume Group to a VM or Initiator IQN. This will allow the VM to access the iSCSI target (requests from an unknown initiator are rejected):

Volumes - Initiator IQN / VM

Click 'Save' and the Volume Group configuration is complete!

This can all be done via ACLI / API as well:

Create VG

```
vg.create VGName
```

Add disk(s) to VG

```
Vg.disk_create VGName container=CTRName create_size=Disk size, e.g. 500G
```

Attach initiator IQN to VG

```
Vg.attach_external VGName InitiatorIQN
```

Path High-Availability (HA)

As mentioned previously, the Data Services IP is leveraged for discovery. This allows for a single address that can be leveraged without the need of knowing individual CVM IP addresses.

The Data Services IP will be assigned to the current iSCSI leader. In the event that fails, a new iSCSI leader will become elected and assigned the Data Services IP. This ensures the discovery portal will always remain available.

The iSCSI initiator is configured with the Data Services IP as the iSCSI target portal. Upon a login request, the platform will perform an iSCSI login redirect to a healthy Stargate.

Volumes - Login Redirect

In the event where the active (affined) Stargate goes down, the initiator retries the iSCSI login to the Data Services IP, which will then redirect to another healthy Stargate.

Volumes - Failure Handling

If the affined Stargate comes back up and is stable, the currently active Stargate will quiesce I/O and kill the active iSCSI session(s). When the initiator re-attempts the iSCSI login, the Data Services IP will redirect it to the affined Stargate.

Volumes - Failback

Health Monitoring and Defaults

Stargate health is monitored using Zookeeper for Volumes, using the exact same mechanism as DSF.

For failback, the default interval is 120 seconds. This means once the affined Stargate is healthy for 2 or more minutes, we will quiesce and close the session. Forcing another login back to the affined Stargate.

Given this mechanism, client side multipathing (MPIO) is no longer necessary for path HA. When connecting to a target, there’s now no need to check ‘Enable multi-path’ (which enables MPIO):

Volumes - No MPIO

Multi-Pathing

The iSCSI protocol spec mandates a single iSCSI session (TCP connection) per target, between initiator and target. This means there is a 1:1 relationship between a Stargate and a target.

As of 4.7, 32 (default) virtual targets will be automatically created per attached initiator and assigned to each disk device added to the volume group (VG). This provides an iSCSI target per disk device. Previously this would have been handled by creating multiple VGs with a single disk each.

When looking at the VG details in ACLI/API you can see the 32 virtual targets created for each attachment:

Here we’ve created a sample VG with 3 disks devices added to it. When performing a discovery on my client we can see an individual target for each disk device (with a suffix in the format of ‘-tgt[int]’):

Volumes - Virtual Target

This allows each disk device to have its own iSCSI session and the ability for these sessions to be hosted across multiple Stargates, increasing scalability and performance:

Volumes - Multi-Path

Load balancing occurs during iSCSI session establishment (iSCSI login), for each target.

Active Path(s)

You can view the active Stargate(s) hosting the virtual target(s) with the following command (will display CVM IP for hosting Stargate):

```
# Windows
Get-NetTCPConnection -State Established -RemotePort 3205
```

As of 4.7 a simple hash function is used to distribute targets across cluster nodes. In 5.0 this is integrated with the Dynamic Scheduler which will re-balance sessions if necesary. We will continue to look at the algorithm and optimize as necessary. It is also possible to set a preferred node which will be used as long as it is in a healthy state.

SCSI UNMAP (TRIM)

Volumes supports the SCSI UNMAP (TRIM) command in the SCSI T10 specification. This command is used to reclaim space from deleted blocks.

Files (File Services)

» Download this section as PDF (opens in a new tab/window)

The Nutanix Files feature allows users to leverage the Nutanix platform as a highly available file server. This allows for a single namespace where users can store home directories and files.

Supported Configurations

The solution is applicable to the configurations below (list may be incomplete, refer to documentation for a fully supported list):

Core Use Case(s):

- Home folders / user profiles
- File storage
- Media repository
- Persistent container storage

Management interfaces(s):

- Prism Element (PE)
- Prism Central (PC)

Hypervisor(s):

- AHV
- ESXi

Upgrades:

- Prism

Compatible Features:

- Nutanix Snapshots and DR
- File share level replication and DR (Smart DR)
- File tiering to S3 compliant storage (Smart Tier)
- File level snapshots including Windows Previous Version (WPV)
- Self Service Restore
- CFT Backups

File Protocols:

- SMB 2
- SMB 3

- NFS v4
- NFS v3

Files Constructs

This feature is composed of a few high-level constructs:

- File Server
  - High-level namespace. Each file server will have its own set of Files VMs (FSVM) which are deployed

- Share
  - Share exposed to users. A file server can have multiple shares (e.g. departmental shares, etc.)

- Folder
  - Folders for file storage. Folders are sharded across FSVMs

The figure shows the high-level mapping of the constructs:

Files Mapping

The Nutanix Files feature follows the same methodology for distribution as the Nutanix platform to ensure availability and scale. A minimum of 3 FSVMs will be deployed as part of the File Server deployment.

The figure shows a detailed view of the components:

Files Detail

The FSVMs are combined into a logical file server instance sometimes referred to as a Files cluster. You can create multiple Files clusters within a single Nutanix cluster. The FSVMs are transparently deployed as part of the configuration process.

The figure shows a detailed view of FSVMs on the AOS platform:

FSVM Deployment Arch

Authentication and Authorization

The Nutanix Files feature is fully integrated into Microsoft Active Directory (AD) and DNS. This allows all of the secure and established authentication and authorization capabilities of AD to be leveraged. All share permissions, user and group management is done using the traditional Windows MMC for file management.

As part of the installation process the following AD / DNS objects will be created:

- AD Computer Account for File Server
- AD Service Principal Name (SPN) for File Server and each FSVM
- DNS entry for File Server pointing to all FSVM(s)
- DNS entry for each FSVM

AD Privileges for File Server Creation

A user account with the domain admin or equivalent privileges must be used to deploy the File Service feature as AD and DNS objects are created.

High-Availability (HA)

Each FSVM leverages the Volumes API for its data storage which is accessed via in-guest iSCSI. This allows any FSVM to connect to any iSCSI target in the event of a FSVM outage.

The figure shows a high-level overview of the FSVM storage:

FSVM Storage

In the event a CVM becomes unavailable (e.g. active path down), iSCSI redirection is used to failover targets to another CVM which will then takeover IO.

FSVM Storage Failover

When the original CVM comes back and is healthy it will be marked as the active path to provide for IO.

In a normal operating environment each FSVM will be communicating with its own VGs for data storage with passive connections to the others. Each FSVM will have an IP which clients use to communicate with the FSVM as part of the DFS referral process. Clients do not need to know each individual FSVM's IP as the DFS referral process will connect them to the correct IP hosting their shares and folder(s).

FSVM Normal Operation

In the event of a FSVM "failure" (e.g. maintenance, power off, etc.) the VG and IP of the failed FSVM will be taken over by another FSVM to ensure client availability.

The figure shows the transfer of the failed FSVM's IP and VG:

FSVM Failure Scenario

When the failed FSVM comes back and is stable, it will re-take its IP and VG and continue to serve client IO.

Objects (Object Services)

» Download this section as PDF (opens in a new tab/window)

The Nutanix Objects feature provides highly scalable and durable object services via an S3 compliant API (More Information on S3: LINK). Given Nutanix Objects is deployed on top of the Nutanix platform, it can take advantage of AOS features like compression, erasure coding, replication and more. Objects was introduced in AOS 5.11.

Supported Configurations

The solution is applicable to the configurations below (list may be incomplete, refer to documentation for a fully supported list):

Core Use Case(s):

- Backup target with compliance-level WORM
- Archival target for PACS / VNA, Email, long term backups
- Big Data including Data Lakes, AI/ML workloads, analytics, machine loggins, SIEM, Pub/Sub
- DevOps, cloud-native, content repos, K8s

Management interfaces(s):

- Prism Central (PC)

Hypervisor(s):

- N/A - Runs on Nutanix MSP (Dependent on MSP supported Hypervisors)

Upgrades:

- LCM

Key Features:

- WORM
- Versioning
- Legal Hold
- Multiprotocol
- Streaming Replication
- Multi-part Upload

- Fast Copy
- S3 Select

Supported Protocols:

- S3 (version 4)
- NFSv3

Nutanix Microservices Platform (MSP)

Nutanix Objects leverages the Nutanix Microservices Platform (MSP) and is one of the first core services to do so.

Nutanix MSP provides a common framework and services to deploy the Objects component's associated containers and platform services like Identity and Access Management (IAM) and Load Balancing (LB).

Key terms

The following key terms are used throughout this section and defined in the following:

- Bucket
  - An organization unit exposed to users and contains the objects (think share to a file on a file server). A deployment can, and typically will, have multiple buckets (e.g. departmental, compartmental, etc.)

- Object
  - The actual unit (blob) of storage and item interfaced with via the API (GET/PUT).

- S3
  - The term used to describe the original object service Amazon Web Services (AWS) introduced and is now used synonymously for an object service. S3 also is used to define the object API which is highly leveraged throughout projects.

- Worker VMs
  - Virtual machines created during object store deployment that host various containerized Objects services. The are also referred to as Objects nodes.

- Objects Browser
  - A user interface (UI) that allows the users to directly launch the object store instance in a web browser and perform bucket and object level operations.

The figure shows the high-level mapping of the conceptual structure:

Objects - Hierarchy

Objects Constructs

This feature is composed of a few high-level constructs:

- Load Balancer
    - The load balancer is part of the Nutanix MSP and serves as a proxy for service and data requests. This ensures high-availability for the service and load balancing among the Objects containers.
- Service Manager
    - The service manager serves as the endpoint for all UI requests and manages object store instances. It is also responsible for collecting stats from instances.
- Metadata Server
    - The metadata server contains all the metadata around a Nutanix Objects deployment (e.g. buckets, objects, etc.). To bolster performance of the metadata server, Nutanix developed ChakrDB, leveraging a RocksDB based Key-Value store which consumes Nutanix Volumes for storage.
- Object Controller
    - Built in-house, this is the data path engine that is responsible for managing object data and coordinating metadata updates with the Metadata Server. It interfaces with Stargate via the Storage Proxy API.
- Region Manager
    - The Region Manager is responsible to managing all of the object storage information (e.g. Region) on DSF.
- Region
    - A region provides the high-level mapping between an object and the corresponding locations on Nutanix vDisk(s). Similar to a vDisk ID, offset and length.
- Atlas Service
    - The Atlas Service is based on the MapReduce framework and is responsible for object lifecycle policy enforcement, garbage collection, tiering and more.
- S3 Adapter
    - Built in-house and all new in Objects 3.2, the S3 adapter provides the S3 API, handles authorization as well as API requests from S3 clients. These are then translated to the Object Controller and processed.

The figure shows a detailed view of the Objects service architecture:

Objects - Architecture

The Objects specific components are highlighted in Nutanix Green. With objects there’s no concept of an “overwrite” hence the CxxD vs. CRUD (Create/Replace/Update/Delete). The commonly employed method for an object “overwrite” is to create a new revision or create a new object and point to the new object.

Object Storage and I/O

An object is stored in logical constructs called regions. A region is a fixed segment of space on a vDisk.

The figure shows an example of the relationship between a vDisk and region:

Objects - vDisk Region

Smaller objects may fit in a chunk of a single region (region id, offset, length), whereas larger objects may get striped across regions. When a large object is striped across multiple regions these regions can be hosted on multiple vDisks allowing multiple Stargates to be leveraged concurrently.

The figure shows an example of the relationship between a object, chunk and region:

Objects - Object Chunk

The object services feature follows the same methodology for distribution as the Nutanix platform to ensure availability and scale. A minimum of 3 object VMs will be deployed as part of the Objects deployment.

Book of Network Services

» Download this section as PDF (opens in a new tab/window)

This book will cover the network and network security services provided by Nutanix.

- Flow Network Security
- Security Central
- Flow Virtual Networking

Here's an overview of each product with a bit of history just in case you've heard of these products by another name.

We released Nutanix Flow as a category and policy-based microsegmentation solution. The stateful, distributed, microsegmentation firewall formerly known as Flow is now called Flow Network Security.

To provide security planning, threat detection, and compliance auditing for both on-premises and cloud environments, Nutanix released Beam, a SaaS based offering. The security component of Beam is now known as Security Central.

For truly multi-tenant networking with overlapping IP addresses, self-service network provisioning, and more, Nutanix released Flow Networking, now called Flow Virtual Networking.

The following sections will cover these in more detail.

Flow Network Security

» Download this section as PDF (opens in a new tab/window)

Flow Network Security is a distributed, stateful firewall that enables granular network monitoring and enforcement between VMs running on the AHV platform as well as external entities they communicate with.

Supported Configurations

The solution applies to the configurations below:

Core Use Case(s):

- Microsegmentation

Management interfaces(s):

- Prism Central (PC)

Supported Environment(s):

- On-Premises:
    - AHV

- Cloud:
    - Nutanix Cloud Clusters on AWS

Upgrades:

- Included in LCM as Flow

Compatible Features:

- Service Chaining
- Security Central

- Calm

Flow Network Security configuration is done via Prism Central by defining policies and assigning categories to VMs. Prism Central can define the security policies and categories of many connected AHV clusters in one place. Each AHV host implements the rules using OVS and OpenFlow as required for distributed enforcement.

Implementation Constructs

Within Flow Network Security, there are a few key constructs:

Categories

Categories are simple text key value pairs used to define groups of VMs that policies are applied to. Typical categories are environment, application type, and application tier. Any key and value tag that is helpful to identify a VM can be used as a category, but some categories such as AppType and AppTier are required for application security policies.

- Category: “Key: Value” or Tag
- Examples Keys: AppType, AppTier, Group, Location

For example, a VM providing production database services may have the following assigned categories:

- AppTier: Database
- AppType: Billing
- Environment: Production

These categories can then be leveraged by security policies to determine what rules or actions to apply. Categories aren’t only for Flow Network Security, you can also use these same categories for protection policies.

Security Policies

Security policies are made of defined rules that determine what is allowed between a source and a destination. A rule inside an application policy includes all the inbound and outbound traffic for a specific application tier. A single rule can include multiple sources and multiple destinations. In the following example, there is one single defined rule for AppTier: Web. If we added allowed traffic to and from AppTier: Database, there would then be two rules.

Flow Network Security - Rules

There are a few types of security policies, and they’re evaluated in the following order:

- Quarantine Policy
  - Deny All traffic for specified VM(s): Strict
  - Deny All traffic except specific traffic for investigation: Forensic
  - Example 1: VMs A,B,C infected with a virus, isolate them to stop the virus from further infecting the network
  - Example 2: VMs A,B,C infected. Quarantine them but allow security team to connect to the VMs for analysis

- Isolation Policy
    - Deny traffic between two categories, allow traffic within category
    - Example: separate tenant A from tenant B, clone environment and allow to run in parallel without affecting normal network communication.

- Application Policy
    - This is your common 5-tuple rule allowing you to define what transport (TCP/UDP), Port, and source/destination is allowed.
    - Allow Transport: Port(s) To,From
    - Example: Allow TCP 443 from VMs with category Location:HQ to VMs with category AppTier:Web

- VDI Policy
    - Identity-based firewall to apply a category to a VDI VM based on the AD Groups of the logged-in user.
    - Implement policy based on the assigned AD groups

The following shows an example using Flow Network Security to control traffic in a sample application:

Flow Network Security - Example Application

Policy State

The policy state determines what action is taken when a rule is matched. With Flow Network Security there are two main states:

- Enforce
    - Enforce the policy by allowing only defined flows and dropping all others.

- Monitor
    - Allow all flows, but highlight any packets that would have violated the policy in the policy visualization page.

Rule Enforcement

Flow Network Security policies are applied to a packet once it leaves the UVM, and before it gets to any other VM. This occurs in the microsegmentation bridge (br.microseg) on the AHV host.

Flow Network Security - Rule Order Overview

Policies are built based on categories, but rule enforcement happens based on the detected VM IP address. The job of Flow Network Security is to evaluate the categories and policies assigned to all the VMs, and then program the right rules into the br.microseg bridge on the host or hosts where protected VMs run. VMs that use Nutanix AHV IPAM have a known IP address as soon as their NIC is provisioned and the rules are programmed when the VM is powered on. The Nutanix Acropolis process intercepts DHCP and ARP messages to detect the IP address of any VM with static IPs or external DHCP. For these VMs, rules are enforced as soon as the VM IP is known.

Rule evaluation for Quarantine, Application, and VDI policies is based on the detected IPv4 address.

Rule evaluation for Isolation policies is based on both the IPv4 address and the VM MAC address.

Evaluation order is on a first-match basis in the following order.

- Quarantine
- Isolation
- Application
- VDI

Flow Network Security - Policy Order

The first policy matched has the action taken, and all further processing stops. For example, if traffic encounters an Isolation policy that is in monitor mode, the action taken is to forward the traffic and log it as allowed and monitored. No further rules are evaluated, even if an Application or VDI policy further down the list would have blocked this traffic.

Further, VMs can belong to only one AppType category and on AppType category can be in the target group of only a single AppType policy. This means that any VM can only belong in the target group of one AppType policy. All traffic into and out of the VM must be defined in this single AppType policy. There is currently no concept of a VM being at the center of multiple Application policies, and therefore no conflict or evaluation order happens between Application policies.

Flow Network Security Next-Generation

Flow Network Security Next-Gen (FNS NG) provides a new security policy model that improves security policy flexibility and scalability while enabling new capabilities not found in previous versions of Flow Network Security. Enhancements made to the networking stack with the network controller, allow Nutanix to introduce the next-generation policy model for Flow Network Security.

Supported Configurations

The solution applies to the configurations below:

Core Use Case(s):

- Microsegmentation for network controller-enabled VLAN subnets
- Microsegmentation in VPC overlay subnets

Management interfaces(s):

- Prism Central (PC)

Supported Environment(s):

- On-Premises:
  - AHV

Prerequisites:

- Nutanix network controller enabled
- Protected entities must use network controller-enabled VLAN subnets or VPC overlay subnets

Upgrades:

- Included in LCM as Flow Network Security 4.0.1

Compatible Features:

- Security Central
- Calm

Flow Network Security Next-Gen configuration is done via Prism Central by defining policies and assigning categories to VMs. Prism Central can define the security policies and categories of many connected AHV clusters in one place. Prism Central and the network controller implement the configured security rules on the virtual switch of each AHV host.

Implementation Constructs

Within FNS NG, there are a few key constructs:

Categories

Categories are text key-value pairs used to define groups of VMs that policies are applied to. Traditional system-defined categories are environment, application type, and application tier. With FNS NG, administrators can create security policies using any system-defined or user-created categories to protect applications. This enhancement provides flexibility, allowing a user to define categories that align with their application constructs and naming conventions.

- Category: “Key: Value” or Tag
- Examples System defined Keys: AppType, AppTier, Group, Location
- Examples User Defined Keys: AccessType, Service, Team

For example, a VM providing production database services may have the following assigned categories:

- AppTier: Database
- AppType: Billing

- Environment: Production
- AccessType: SecureAdmin
- Service: Monitor

These categories can then be leveraged by security policies to determine what rules or actions to apply. Categories aren’t only for Flow Network Security, you can also use these same categories for protection policies.

FNS NG Security Policies

Security policies are constructed of defined rules that determine what is allowed between a source and a destination. A rule inside an application policy includes all the inbound and outbound traffic for specific secured entities. A single rule can include multiple sources and multiple destinations. With FNS 4.0.1 and later, a VM can be a secured entity in multiple security policies, as opposed to requiring a VM to appear in only a single, monolithic policy in previous FNS versions. This allows users to create broader policies applied to a wide range of VMs to address requirements like shared services, monitoring, and even a default catch-all security policy, while still having a tightly focused security policy to protect applications. Using the multiple-policy approach streamlines security policy management, making it easier to update a policy when requirements or resources change.

For example, here’s how monolithic policies are used in the original FNS policy model. When an organization has 15 different applications to secure, each application has its own security policy for a total of 15 policies. Each of those application security policies contains rules specific to the requirements of the protected application. There are additional rules within each policy to secure VM management access and system monitoring. In this example, a change to the monitoring platform or the method used to monitor the component VMs of an application would require a change to each security policy. There are 15 security policies that would need to be updated. Each policy change increases the risk of a configuration error that could introduce an unwanted security exposure. Applying multiple security policies to a VM helps reduce or eliminate this risk.

Policy Interaction

Let’s investigate the multiple combined policy capabilities in FNS NG. Using the same example scenario previously mentioned, the applications are each protected with security policies specific to the requirements of the application, these are now 15 specific policies. In addition, the component VMs that make up the application could have a security policy applied that addresses management access and another security policy used for monitoring. With this method, a change to the monitoring platform or the method used to monitor the component VMs of an application would only require an update to the security policy that is used to restrict monitoring access. This is a change to only one security policy, monitoring, leaving the 15 security policies for the applications unaltered.

In the following example, we have a VM that is a secured entity in three separate security policies, Shared Services, MyHR App, and Enterprise Monitoring. It’s important to note that this same VM is protected with three separate categories. A single category can’t appear as a security entity in multiple policies.

Policy 1: Shared Services

Flow Network Security Next Generation - Policy 1: Shared Services

This example Shared Services policy is used to permit secure administrative access to VMs that are in the category AccessType:SecAdmin This policy can be applied to multiple VMs and used as a default secure administrative access policy.

Policy 2: MyHR App

Flow Network Security Next Generation - Policy 2: MyHR App

The example MyHR App policy is used to restrict access to a specific application. VMs in the category App: HR_Web are protected by this policy.

Policy 3: Enterprise Monitoring

Flow Network Security Next Generation - Policy 3: Enterprise Monitoring

The Enterprise Monitoring policy is used to restrict application monitoring services to a specific source group. This is a good example of a policy that can be applied to many VMs independent of the application they are a part of. This could be a good default policy for new applications.

Security Policy Management

FNS NG introduces new policy mode constructs to enhance policy management. These additions streamline the task of administration at scale by making it easy to create new policies from templates or existing policies.

Security Policy Management options:

- Monitor
  - Allow all flows, but highlight any packets that would have violated the policy in the policy visualization page.

- Enforce
  - Enforce the policy by allowing only defined flows and dropping all others.

- Save
  - The save option allows you to create a policy and make changes without applying the policy. This is helpful as information and requirements for the policy are being defined. You can create the policy and iterate changes as you solidify requirements. In this mode, the policy is saved in a draft state.

- Clone
  - The clone option allows administrators to make a copy of an existing security policy. With this functionality, users can create a security policy with configured default security rules, and then save the policy to be used as a template for new application policies. Any time there is a new application that needs to be secured, users can clone the saved template policy and then update the secured entities of the policy to reflect the requirements of the application. This is a good method for maintaining the default security rules required for all apps.

Security Central

» Download this section as PDF (opens in a new tab/window)

Nutanix Security Central is a Software as a Service (SaaS) based offering that provides microsegmentation security planning, threat detection alerts, and continuous compliance monitoring. Using multiple Machine Learning (ML) models and algorithms such as Louvain, Arima, tree based, and clustering, Security Central gives insight into the security of your on-premises deployments based on over 500 audit checks and security best practices.

The Security Central portal at https://flow.nutanix.com/securitycentral provides an inventory and configuration assessment of your cloud and on-premises infrastructure to scan for common and high-risk configuration errors. Security Central users can also get instant insights on security status by using custom or system-defined SQL like queries. Powered by this inventory and coupled with compliance tracking tools, security posture monitoring is also provided. Finally, network flow data ingested from on-premises AHV clusters provides near real-time threat detection based on machine learning analysis of traffic patterns.

Supported Environments

- Nutanix on-premises
  - Private cloud using AHV with Flow Network Security enabled

- Public clouds: AWS and Microsoft Azure
  - Monitor resources and services within your Public Cloud infrastructure

Management Interface(s)

- Nutanix Security Central SaaS UI
- Flow Security Central VM (FSC VM)
  - Initial configuration and upgrades when necessary

Implementation Constructs

Security Central introduces a few new constructs to provide a security monitoring and management framework. Here are the two elements introduced:

- Flow Security Central VM (FSC VM) [Required only for Nutanix on-prem]
  - Used during initial configuration
  - IPFix log collection
  - Inventory collection
  - Push security policy configurations

- Security Central SaaS
  - Security posture monitoring
  - User & network anomaly detection
  - Compliance reporting
  - Microsegmentation security planning
  - Multi-cloud inventory and query

Upgrades

- Automatic as Security Central is a SaaS platform
  - As new features are released, they are available at the next login.

- On occasion, an upgrade to the on-prem FSC VM may be required to enhance functionality
  - Upgrades can be performed on the Flow Security Central VM Settings page

- The latest FSC VM images (both GUI and server-only) are located on the Nutanix Portal

Security Central Architecture Overview

Security Central Architecture Overview

Security Central High Level Architecture Detail

Security Central introduces a new service VM known as the FSC VM, which is required to enable security monitoring for Nutanix clusters. The FSC VM acts as a proxy and aggregates information from the Nutanix clusters. It then sends this information to Security Central through a secure network connection. The FSC VM collects inventory information about the environments and VMs, referred to as metadata. Security Central does not collect any data owned by the VM, such as installed software packages and versions, but integration with partner agents such as Qualys is available to enrich the metadata.

Once installed, the administrator connects to the FSC VM UI to enable network log collection. The FSC VM then instructs Prism Central to enable the IPFIX exporter on all AHV clusters managed by this Prism Central instance. The FSC VM also collects cluster inventory information from Prism Central for all registered clusters and VMs. The inventory information collected includes items such as VM names, connected network information, configuration information and any categories which may be assigned to the VMs. Inventory polling recurs every 3 hours and the SaaS portal can log changes and perform analysis.

Inventory and IPFix logs are securely transmitted to the Security Central SaaS environment using HTTPS/TLS connections. IPFix logs are transmitted in 15 minute intervals allowing the FSC VM to transmit data in batched increments, decreasing storage capacity required for the FSC VM and reducing network constraints to the cloud. Administrators can push required inventory updates to the cloud manually using the FSC VM user interface if needed.

An FSC VM is needed for each Prism Central (PC) deployment regardless of the number of clusters managed by that PC. The FSC VM will facilitate internal communications to Prism Central, and outbound communication to the SaaS portal. Security Central uses the TCP ports listed below for communication between components. Please ensure that your firewall has the following ports open:

- TCP Port 9440 for the FSC VM to connect to the Prism Central VM
- TCP Port 443 from the FSC VM to connect to *.nutanix.com and *.amazonaws.com

Refer to the Security Central Ports and Protocols page for the complete list of ports.

The FSC platform undergoes strict security controls. Visit Nutanix Trust to view FSC compliance certifications and learn more.

To ensure you meet the configuration requirements for the Flow Security Central VM, please consult the Security Central Guide.

Core Use Cases

- Monitoring and Visibility
  - Multi-cloud dashboards, asset inventory, reporting, and alerts
- Audit and Remediation
  - Insights on Nutanix environments and public clouds using real-time, automated security audits
- Compliance
  - Continuous environment monitoring, automated compliance checks, and assessment for STIG, PCI, and HIPAA
- Security Planning
  - VM to VM network traffic visualization, workload categorization, automated security policy recommendation
- Investigating Violations
  - Perform detailed investigation with an SQL like query language
- Threat Detection Alerting
  - Network- and user-based anomaly detection. End user behavior analysis (EUBA) helps to detect internal and external security threats

Security Central Key Features

Main Dashboard

The first screen presented after successful login to Security Central is the main dashboard. The dashboard provides an at-a-glance view of the many areas monitored and alerted on by Security Central. More detailed displays can be launched from the context of each individual widget.

Security Central Main Dashboard

Audit and Remediation

Utilizing security audits provides deep insights into the security of your on-premises deployments. Security Central runs more than 500 out-of-the-box automated security audits of your environment and reports any audit failures along with steps to remediation. Security Central audit checks align to the following categories and Nutanix security best practices:

- Access Security
  - Configuration and alerts for items that could allow unprivileged access or lack of access auditing

- Data Security
  - Nutanix clusters without Data-at-Rest Encryption (DARE) enabled. DARE is an essential component to securing your infrastructure that prevents unauthorized access to data by systems and users that do not possess the encryption keys.

- Host Security
  - Nutanix VMs without backup protection enabled, making it difficult to recover should a VM be infected by ransomware

- Logging & Monitoring
  - Prism Central critical alerts unacknowledged could indicate that alerts are being ignored or missed. These alerts can indicate a potential exposure risk that could put your infrastructure in jeopardy

- Network Security
  - VMs allow all traffic from all sources can create an exceptionally large attack surface for malicious actors to compromise the VM and infiltrate your environment.

In addition to the constantly updated built-in audits, Security Central provides the capability to customize audits and reports using Common Query Language (CQL). This allows you to run CQL queries on your cloud inventory for various attributes, and create audit checks for your specific use-cases.

Compliance

Compliance is critical for many Nutanix customers. Maintaining compliance allows you to validate that your company's operating environments meet the governing standards they must follow. Monitoring environments for compliance can be challenging and time consuming. Continuous compliance monitoring capabilities are needed to stay on top of ever changing requirements, regulations and environments. Customers benefit from automating these checks, allowing for a concise view and faster time to resolution when addressing compliance violations.

Security Central provides audit checks for common regulatory frameworks such as PCI-DSS, HIPAA, NIST and Nutanix security best practices using the Nutanix STIG policy. The Compliance dashboard provides a compliance score for each of the frameworks being monitored. The score elements can be further examined to view elements of the given framework that are monitored and audited, the number of passed checks and the number of failed checks. The compliance view provides detail on these checks such as which check failed, which section of the framework the check is associated with, and what issues were discovered. The compliance reports and their details can also be exported for offline use and sharing. Security Central also provides extensible capabilities that allow you to create custom audits and custom compliance policies that are based on your business requirements.

Findings and Alerts

> "A problem well-stated is a problem half-solved." - Charles Kettering

With the ever changing threat landscape, security monitoring and alerting is critical in today's environments. Continuously monitoring for and alerting on threats and vulnerabilities found in your network, security controls, servers, endpoints and user applications allowing a proactive approach to alerting in order to strengthen your overall security posture and providing a quicker time to resolution when an issue is detected.

The Findings and Alerts view within Security Central provides a view into your current security posture. This view displays detected configuration issues and anomalous behaviors observed in your Nutanix clusters that are being monitored by Security Central. These findings are based on the selected audit policy. Users have customizable options on how these issues are displayed, providing great flexibility to tailor the findings by audit categories, resources, roles, and business requirements.

Threat Detection Alerts are included within the Findings and Alerts view. Security Central analyzes the Nutanix IPFix network logs to detect and report observed potential threats and anomalous behavior occurring within the monitored Nutanix clusters. These alerts are modeled using machine learning and observed data points. The following are some of the potential threats and behaviors being detected along with the minimum number of days or data-points that must be observed in order for the anomaly to trigger an entry or alert :

- VMs communicating with Blacklisted IPs (known suspect IPs)
  - 1 data-point

- VMs with potential Dictionary Attack
  - 1 data-point

- VMs with potential Denial of Service (DDOS)
  - 21 days

- VLANs with unusual network behavior
    - 21 days
- VMs with unusual network behavior
    - 21 days
- VMs with potential Port Scan Attack
    - 21 days
- VMs with potential Data Leak Attacks
    - 21 days

Security Planning

When planning to secure an application, there are many components to discovery and information collection required to create an effective security policy. Gathering this information can be quite the challenge. You are often tasked with collecting logs from firewalls, networking gear, applications and operating systems to gain an understanding of your application’s communication profile. Once that data has been gathered and analyzed, you will consult with the application owners to compare the observed findings to the expected behavior.

Security Central’s ML-based Security Planning provides a detailed visualization to aid you in discovery and planning of your application security policies. You can visualize your network traffic flows within your Nutanix on-prem clusters and Security Central makes recommendations on how to categorize and secure your applications.

Within the Security Planning section, you have the flexibility to utilize up to 2 levels of groupings for your applications and environments. This capability allows you to drill down your analysis to specific clusters, VLANs, VMs, and categories, making it extremely helpful to focus on securing your applications. With this grouping, you also have the capability to download all observed or filtered network traffic for further offline sharing, discovery, and planning.

Using the inherent machine learning capabilities, Security Central also has the ability to make category recommendations and assignments to VMs based on observed network flows. This can be especially useful in new microsegmentation or application deployments. Categorizing your application VMs is an important step to securing your application. To take it a step further, Security Central can also generate inbound and outbound rule recommendations and create application security policies in monitor mode on your Nutanix AHV clusters.

With the security policy in monitor mode, you can observe the behavior of the application being secured, without applying the policy actions. You can use Prism Central to make edits of the policy if needed prior to enforcing the policy.

Investigate

Investigate helps you analyze security and operational insights for your hybrid cloud infrastructure. It allows you to analyze network logs and security configurations using CQL (Common Query Language). CQL queries offer a user-friendly experience, similar to SQL. To get desired almost real-time results, you can effortlessly run CQL queries on both on-premises and public cloud infrastructure.

Investigate supports following types of queries:

- Inventory and Config: Search for the resource inventory and configuration in your on-premises and public cloud environments
- Network: Search network events in your on-premises environment

Providing cross stack visibility is one of the prime use case for investigate in hybrid cloud environments. For example, in most cases we have multiple stacks of security rules. For the on-premises side we may have Flow Network Security policies and in cloud we may have security groups on EC2 instances. In this case investigate queries can provide a list of all the microsegmentation policies and AWS security policies applied to particular VMs.

The investigate feature also provides a library of most used queries to start with and sets the baseline to create more complex queries based on your needs

Example queries:

- List VM with status (protected or unprotected) and the policy name if the VM is protected

```
SELECT NX.VM.name, NX.VM.NetworkPolicy.*, NX.VM.Category.* FROM NX.VM
```

- List all tags of EC2 instances in AWS

```
 SELECT AWS.EC2Instance.Tag.tagKey, AWS.EC2Instance.Tag.tagValue, AWS.EC2Instance.instanceId, AWS.EC2Instance.name FROM AWS.EC2Instance
```

Integrations

Security architectures often consist of solutions from multiple vendors to build a defense-in-depth posture. Solutions that monitor endpoints for threats and vulnerabilities, ticketing systems, log management, and threat and event analytics can all be critical components in a security architecture. While these products are essential, they are often standalone with limited integration with other components of the security infrastructure. Integration is key to bringing security solutions together within the construct of a security architecture. This drives efficiency and faster time to threat awareness, analysis, and threat remediation.

Security Central provides the capability to integrate non-Nutanix applications directly with Security Central. These integrations cover multiple solutions from OS level monitoring and protection to enterprise ticketing systems and analysis engines. Some of the supported integrations are listed below:

- Splunk
- Webhook
- ServiceNow
- Qualys

More detailed information on the specifics of each integration can be found in the Security Central guide.

Flow Virtual Networking

» Download this section as PDF (opens in a new tab/window)

Flow Virtual Networking allows you to create completely isolated virtual networks that are separated from the physical network. You can do multi-tenant network provisioning with overlapping IP addresses, self-service network provisioning, and IP address preservation. Best of all, these isolated virtual networks provide security by default.

Supported Configurations

Core Use Cases:

- Multi-tenant networking
- Network isolation for security
- Overlapping IP addresses

- Self-service network creation
- VM IP mobility
- Hybrid Cloud connectivity

Management interfaces(s):

- Prism Central (PC)

Supported Environment(s):

- On-Premises:
  - AHV

Upgrades:

- Major Feature Upgrades Depend on
  - Prism Central
  - AHV

- Minor Feature Upgrades Included in LCM
  - Advanced Network Controller
  - Network Gateway (VPN, Subnet Extension, BGP)

Implementation Constructs

Flow Virtual Networking introduces a number of new constructs to provide a complete networking solution.

- VPCs
- Overlay Subnets
- Routes
- Policies
- External Networks
  - NAT
  - Routed (NoNAT)
  - Multiple External Networks

- Network Gateway
  - Layer 3 VPN
  - Layer 2 Extended Subnets with VXLAN
  - BGP

VPCs (Virtual Private Clouds)

The VPC or Virtual Private Cloud is the basic unit of Flow Virtual Networking. Each VPC is an isolated network namespace with a virtual router instance to connect all of the subnets inside the VPC. This is what allows the IP addresses inside of one VPC to overlap with any other VPC, or even with the physical network. A VPC can expand to include any cluster managed by the same Prism Central, but generally a VPC should exist only within a single AHV cluster, or within clusters in the same availability zone. More on that when we get to External Networks.

Flow Virtual Networking - VPC

Overlay Subnets

Each VPC can have one or more subnets and they’re all connected to the same VPC virtual router. Behind the scenes, a VPC leverages Geneve encapsulation to tunnel traffic between AHV hosts as needed. This means the subnets inside the VPC don’t need to be created or even present on the top-of-rack switches for VMs on different hosts to communicate. When two VMs in a VPC on two different hosts send traffic to each other, packets are encapsulated in Geneve on the first host and sent to the other host where they’re decapsulated and sent to the destination VM.

Flow Virtual Networking - Geneve Encapsulation

When you select a NIC for a VM, you can place that NIC into an overlay subnet, or a traditional VLAN backed subnet. When you choose an overlay subnet, that is also choosing the VPC.

Pro tip

Each VM can be placed inside only a single VPC. You can't connect a VM to both a VPC and a VLAN at the same time, or to two different VPCs at the same time.

Routes

Every VPC contains a single virtual router and there are a few types of routes:

- External Networks

- Direct Connections
- Remote Connections

External networks should be the default destination of the 0.0.0.0/0 network prefix for the whole VPC. You can choose an alternate network prefix route for each external network in use. For completely isolated VPCs, you may choose not to set a default route.

Directly connected routes are created for each subnet inside the VPC. Flow Virtual Networking assigns the first IP address of each subnet as the default gateway for that subnet. The default gateway and network prefix are determined by the subnet configuration and cannot be altered directly. Traffic between two VMs on the same host and same VPC, but in two different subnets, will be routed locally in that host.

Remote connections such as VPN connections and External Networks can be set as the next hop destination for a network prefix.

Pro tip

Each network prefix inside a VPC routing table must be unique. Don't program two different next hop destinations with the same destination prefix.

Policies

The virtual router acts as a control point for traffic inside a VPC. You can apply simple stateless policies here, and any traffic that flows through the router will be evaluated by the policies. Traffic from one VM to another inside the same subnet won’t go through a policy.

Flow Virtual Networking - Policies

Inside a VPC, policies are evaluated in priority order from highest (1,000) to lowest (10).

You can match traffic based on the following values:

- Source or Destination IPv4 Address
  - Any
  - External to the VPC (Any traffic entering the VPC)
  - Custom IP Prefix
- Protocol
  - Any
  - Protocol Number
  - TCP
  - UDP
  - ICMP
- Source or Destination Port Number

Once traffic is matched a policy can take the following actions:

- Permit
- Deny
- Reroute
  - Redirect traffic to another /32 IPv4 address in another subnet

The reroute policy is incredibly helpful to take action like routing all inbound traffic through a load balancer or firewall VM running inside another subnet in the VPC. This has the added value of requiring only a single Network Function VM (NFVM) for all traffic inside the VPC, rather than a traditional service chain that requires an NFVM per AHV host.

Pro tip

Stateless policies require separate rules defined in both the forward and reverse direction if a Permit rule is overriding a Drop rule. Otherwise, return traffic would be denied by the Drop rule. Use similar priorities to group these matching forward and reverse entries.

External Networks

An External Network is the primary way traffic enters and exits a VPC. External Networks are created in Prism Central and exist on only a single Prism Element cluster. This network defines the VLAN, the default gateway, the IP address pool, and the NAT type for all the VPCs using it. One External Network can be used by many VPCs.

There are two types of External Networks:

- NAT
- Routed (NoNAT)

Starting in PC.2022.1 and AOS 6.1, you can select a maximum of two External Networks for a VPC. A VPC can have at most one NAT External Network and at most one Routed (NoNAT) External Network. You cannot configure two NAT, or two Routed External Networks in the same VPC.

NAT

A NAT (Network Address Translation) External Network hides the IP addresses of VMs in the VPC behind either a Floating IP or the VPC SNAT (Source NAT) address. Each VPC has an SNAT IP address selected randomly from the External Network IP pool, and traffic exiting the VPC is rewritten with this source address.

Flow Virtual Networking - NAT External Network

Floating IP addresses are also selected from the External Network IP pool and are assigned to VMs in a VPC to allow ingress traffic. When a Floating IP is assigned to a VM, the egress traffic is rewritten with the Floating IP instead of the VPC SNAT IP. This is useful for advertising public services outside the VPC without revealing the private IP address of the VM.

Routed

Routed, or NoNAT External Networks allow the IP address space of the physical network to be shared inside the VPC through routing. Instead of a VPC SNAT IP address, the VPC router IP is selected randomly from the External Network pool. Share this VPC router IP with the physical network team so they can set this virtual router IP as the next hop for all of the subnets provisioned inside the VPC.

For example, a VPC with an External Network of 10.5.0.0/24 may be assigned a virtual router IP of 10.5.0.200. If the subnets inside the VPC are created in the 10.10.0.0/16 network, the physical network team will create a route to 10.10.0.0/16 via 10.5.0.200. The 10.10.0.0/16 network becomes the externally routable prefix for the VPC.

Flow Virtual Networking - Routed External Network

Multiple External Networks

Having two external networks is useful in the following scenarios.

- Connectivity to internal and external resources through separate routes.
  - Internal or Corporate through Routed, or NoNAT
  - External or Internet services through NAT and Floating IPs

- Providing Floating IPs for network gateway VMs inside the VPC. Network gateways inside a VPC require Floating IPs.
  - VPN Network Gateway VM
  - VXLAN VTEP Network Gateway VM
  - BGP Network Gateway VM inside the VPC

The following VPC is connected to a NAT network for Internet access, as shown in the VPC routing table **Virtual Router AZ01-VPC-1: Routing Table** . All Internet directed traffic (0.0.0.0/0) exits the VPC through AZ01-NAT external network. This VPC is also connected to a Routed or NoNAT external network and all internal traffic destined to 10.5.0.0/16 exits through the AZ01-NoNAT external network.

Flow Virtual Networking - Multiple External Networks

Network Gateways

A network gateway acts as a connector between subnets. These subnets can be of many different types and in different locations.

- Subnet Types
  - ESXi VLAN
  - AHV VLAN
  - VPC Overlay (Network gateways in a VPC require Floating IPs)
  - Physical Network VLAN

- Subnet Locations
  - On-premises VLANs
  - On-premises VPCs
  - Cloud VPCs

The network gateway has several methods of connecting subnets.

- Layer 3 VPN
  - Network Gateway to Network Gateway
  - Network Gateway to Physical Firewall or VPN

- Layer 2 VXLAN VTEP
  - Network Gateway to Network Gateway
  - Network Gateway to Physical Router or Switch VTEP

- Layer 2 VXLAN VTEP over VPN
  - Network Gateway to Network Gateway

- BGP
  - Dynamic VPC subnet advertisement to a remote BGP neighbor

Layer 3 VPN

In the Layer 3 VPN connection type, two subnets with two separate network prefixes are connected. For example, local subnet 10.10.1.0/24 can be connected to remote subnet 10.10.2.0/24. The following diagram assumes a NAT external network and the VPN VM is assigned a Floating IP from this NAT network.

Flow Virtual Networking - Layer 3 VPN

When using two network gateways inside a VPC, each gateway VM is assigned an external Floating IP address from the NAT external network DHCP pool, and they must be able to communicate over these addresses. When the network gateway is inside a VLAN, the assigned IP address must be reachable from the remote site.

You can also connect the network gateway VM to a remote physical firewall or VPN appliance or VM. The local network gateway must still be able to communicate with the remote physical or virtual appliance.

Traffic from each subnet to the remote subnet is directed over the created VPN connection using IP routing inside the VPC. Routes can be either static or shared between network gateways using BGP. Traffic inside this VPN tunnel is encrypted with IPSEC.

Pro tip

Use VPN with IPSEC encryption when traffic between subnets will go over a public link such as the Internet.

Layer 3 VPN Detailed View for NAT

The following diagram shows a more detailed view of the assigned interfaces and addresses for a VPN network gateway VM connecting between two VPCs with NAT external networks.

Flow Virtual Networking - Layer 3 VPN NAT Detail

Note that a new internal subnet in the 100.64.1.0/24 overlay network is automatically created when the VPN gateway VM is deployed. This VM reserves a Floating IP from the NAT external network pool. Pay special attention to the routing table within the VPC. Traffic to the remote subnet is routed through the VPN connection.

Layer 3 VPN Detailed View for Routed or NoNAT

The following diagram shows the detailed interface view when a routed external network is attached to the VPC. Multiple external networks are mandatory in this case to provide a NAT external network for the VPN VM Floating IP.

Flow Virtual Networking - Layer 3 VPN NoNAT Detail

In this case the AZ01-1 and AZ01-2 VPC routing tables contain default routes to the NAT network, and more specific routes to the private internal NoNAT or Routed network. The VPN tunnel traffic traverses the NAT external network to reach the remote site.

Layer 2 VXLAN VTEP Subnet Extension

In the Layer 2 VXLAN VTEP case, called subnet extension, two subnets that share the same network prefixes are connected. For example, local overlay subnet 10.10.1.0/24 is connected to a remote overlay subnet that also uses 10.10.1.0/24.

Flow Virtual Networking - Layer 2 VXLAN VTEP

When using two network gateway VMs to connect two VPCs, each VPN VM is assigned an external floating IP address, and the two VPN VMs must be able to communicate over these addresses. Therefore a VPC that uses layer 2 subnet extension must have a NAT external subnet with available floating IPs. If you are performing subnet extension for a VPC that uses routed, or NoNAT external subnets exclusively, you must add a NAT external subnet to the VPC.

Layer 2 Subnet Extension Detailed View for NAT

The following diagram shows the detailed interface view of a network gateway VM used for subnet extension between two VPCs with NAT external subnets. The network gateway VM for subnet extension has two interfaces, one interface inside the internal 100.64.1.0/24 overlay subnet, and another interface inside the overlay subnet being extended. The floating IP address of the network gateway VM is assigned to the 100.64.1.0/24 internal interface. You can think of subnet extension like plugging a virtual cable in between two networks. The network gateway VM acts as a layer 2 bridge with an interface in the target subnet.

Flow Virtual Networking - Subnet Extension NAT VPC Detail

Layer 2 Subnet Extension Detail View for Routed or NoNAT

The following detailed diagram shows both the Routed and NAT subnets needed for subnet extension in a Routed VPC. Careful planning is needed with routed networks and subnet extension to ensure traffic is directed to the right endpoints. In this example, traffic from overlay subnet 10.10.1.0/24 in AZ1 destined for 10.20.2.0/24 will route through the physical NoNAT network and not the subnet extension. Traffic from AZ2 10.10.1.0/24 destined to 10.20.2.0/24 will instead route through the Nutanix virtual router. Consider these routing paths when using subnet extension.

Flow Virtual Networking - Subnet Extension Routed VPC Detail

Layer 2 Subnet Extension Physical Device

The local network gateway can also connect to a remote physical VXLAN VTEP terminating device such as a physical switch. The physical device can be any standard VXLAN device from popular vendors such as, but not limited to, Cisco, Arista, and Juniper. Just enter the remote physical device IP address in the VXLAN VTEP connection.

Traffic from the local subnet to the remote subnet is exchanged via layer 2 switching and encapsulated in unencrypted VXLAN. Each network gateway maintains a source MAC address table and can forward unicast or flooded packets to the remote subnet.

Pro tip

Use VXLAN VTEP only when traffic is traversing private or secured links, because this traffic is not encrypted.

Layer 2 VXLAN VTEP over VPN

For extra security, the VXLAN connection can be tunneled through an existing VPN connection to add encryption. In this case, the network gateway VM provides the VXLAN and the VPN connections, so a network gateway VM is required in both the local and remote subnet.

Flow Virtual Networking - Layer 2 VXLAN VTEP over VPN

BGP Gateway

The BGP Gateway automates the exchange of routing information between a VPC and an external BGP peer. Use the BGP Gateway when a VPC has a Routed or NoNAT external network along with an overlay subnet whose IP address space has a requirement to communicate with external networks without using NAT, and this communication needs dynamic address advertisement. Unlike other network gateway options in Flow Virtual Networking, the BGP gateway is not in the data path. This gateway is part of the control plane.

BGP Gateway Deployed on a VLAN

BGP Gateway Deployment

The BGP gateway can be deployed on an underlay VLAN-backed network or within the VPC in an overlay subnet.

The BGP gateway establishes peering sessions with upstream infrastructure routers. A peering session is used to advertise the reachability of the configured routable VPC subnets to the BGP neighbor. The gateway can advertise up to 20 routable prefixes. The next hop advertised from the VPC BGP gateway is the VPC router interface that is connected to the NoNAT, or Routed, external subnet. The gateway supports one session per BGP peer and up to 10 unique BGP peers.

The remote BGP neighbor in turn advertises this reachability to the rest of the network. The remote BGP neighbor also advertises external routes to the VPC BGP gateway. The gateway can learn and install up to 250 routes into the VPC route table per BGP peering session. The gateway will advertise the configured externally routable IP addresses. These IP addresses are defined in CIDR format and support x.x.x.x/16 through x.x.x.x/28 networks. This provides the flexibility to advertise your entire subnet or smaller subsets of network addresses. The gateway will learn routes from configured external peers and install these routes into the routing table of the VPC.

BGP Gateway in a VLAN

When the BGP gateway is deployed in a VLAN-backed subnet, it should have IP reachability to access the BGP peers. The VPC is configured with a Routed or NoNAT external subnet and the BGP gateway advertises the externally routable IP addresses of the VPC with the next-hop of the VPC virtual router.

Routing Tables with BGP Established

Pro tip

Deploying the BGP gateway in a VLAN-backed underlay network makes configuration easier as there are no requirements for a NAT external network and floating IP assignment.

BGP Gateway Deployed in Overlay Subnet

BGP Gateway Deployed in VPC Overlay Subnet

There is also the option to deploy the BGP gateway on an overlay subnet within the VPC. Placing the BGP gateway in an overlay subnet requires two types of external subnets, NoNat and NAT. The BGP gateway requires an assigned floating IP address from the NAT external subnet pool. The gateway uses this floating IP address to establish BGP sessions with remote neighbors. These peering sessions will be routed through the NAT external network path.

Routing Table with BGP Gateway Deployed in a VPC Overlay Subnet

The next hop advertised from the VPC BGP gateway is the VPC router interface that is connected to the NoNAT (routed) external subnet. All traffic to destinations learned from a BGP neighbor will exit the VPC via the NoNAT external subnet.

Book of Backup / DR Services

» Download this section as PDF (opens in a new tab/window)

Having a solid backup strategy is an integral part of any infrastructure design. In this book we will dive into the details of Nutanix Leap, which provides policy driven backup, DR, and runbook automation.

Note: Nutanix Mine section coming soon!

Leap (Policy Driven DR/Run Books)

» Download this section as PDF (opens in a new tab/window)

The Nutanix Leap feature provides policy driven backup, DR and run book automation services configured via Prism Central (PC). This capability builds upon and extends the native DR and replications features that have been availble in AOS and configured in PE for years. For more information on the actual back-end mechanism being leveraged for replication, etc. refer to the 'Backup and Disaster Recovery (DR)' section in the 'Book of AOS'. Leap was introduced in AOS 5.10.

Test Drive

For those who are interested in getting hands on, take it for a spin with Nutanix Test Drive!

https://www.nutanix.com/test-drive-disaster-recovery

Supported Configurations

The solution is applicable to the configurations below (list may be incomplete, refer to documentation for a fully supported list):

Core Use Case(s):

- Policy based backups and replication
- DR run book automation
- DRaaS (via Xi)

Management interfaces(s):

- Prism Central (PC)

Supported Environment(s):

- On-Prem:
  - AHV (As of AOS 5.10)
  - ESXi (As of AOS 5.11)

- Cloud:
  - Xi (As of AOS 5.10)

Upgrades:

- Part of AOS

Compatible Features:

- AOS BC/DR features

Key terms

The following key terms are used throughout this section and defined in the following:

- Recovery Point Objective (RPO)
  - Refers to the acceptable data loss in the event of a failure. For example, if you want an RPO of 1 hour, you'd take a snapshot every 1 hour. In the event of a restore, you'd be restoring data as of up to 1 hour ago. For synchronous replication typically an RPO of 0 is achieved.

- Recovery Time Objective (RTO)
  - Recovery time objective. Refers to the period of time from failure event to restored service. For example, if a failure occurs and you need things to be back up and running in 30 minutes, you'd have an RTO of 30 minutes.

- Recovery Point
  - A restoration point aka snapshot.

Implementation Constructs

Within Nutanix Leap, there are a few key constructs:

Protection Policy

- Key Role: Backup/Replication policy for assigned categories
- Description: A protection policy defines the RPO (snap frequency), recovery location (remote cluster / Xi), snapshot retention (local vs. remote cluster), and associated categories. With Protection Policies everything is applied at the category level (with a default that can apply to any/all). This is different from Protection Domains where you have to select VM(s).

The following image shows the structure of the Nutanix Leap Protection Policy:

Leap - Protection Policy

Recovery Plan

- Key Role: DR run book
- Description: A Recovery Plan is a run book that defines the power on sequencing (can specify categories or VMs) and network mapping (primary vs. recovery and test failover / failback). This is most synonymous with what people would leverage SRM for. NOTE: a Protection Policy must be configured before a Recovery Plan can be configured. This is necessary as the data must exist at the recovery site in order for it to be recovered.

The following image shows the structure of the Nutanix Leap Recovery Plan:

Leap - Recovery Plan

Linear Retention Policy

- Key Role: Recovery Point retention policy
- Description: A linear retention policy specifies the number of recovery points to retain. For example, if the RPO is 1 hour and your retention is set to 10, you'd keep 10 hours (10 x 1 hour) of recovery points (snaps).

Roll-up Retention Policy

- Key Role: Recovery Point retention policy
- Description: A roll-up retention policy will "roll-up" snaps dependent on the RPO and retention duration. For example, if the RPO is 1 hour and your retention is set to 5 days it'll keep 1 day of hourly and 4 days of daily recovery points. The logic can be characterized as follows: If retention is n days, keep 1 day of RPO and n-1 days of daily recovery points. If retention is n weeks, keep 1 day of RPO and 1 week of

daily and n-1 weeks of weekly recovery points. If retention is n months, keep 1 day of RPO and 1 week of daily and 1 month of weekly and n-1 months of monthly recovery points. If retention is n years, keep 1 day of RPO and 1 week of daily and 1 month of weekly and n-1 months of monthly recovery points.

Linear vs. roll-up retention

Use linear policies for small RPO windows with shorter retention periods or in cases where you always need to be able to recover to a specific RPO window.

Use roll-up policies for anything with a longer retention period. They're more flexible and automatically handle snapshot aging / pruning while still providing granular RPOs for the first day.

The following shows a high-level overview of the Leap constructs:

Leap - Overview

The following shows how Leap can replicate between on-premises and Xi:

Leap - Topology

Usage and Configuration

The following sections cover how to configure and leverage Leap.

The high-level process can be characterized into the following high-level steps:

1. Connect to Availability Zones (AZs)
2. Configure Protection Policies
3. Configure Recovery Plan(s)
4. Perform/Test Failover & Failback

Connect Availability Zone(s)

The first step is connecting to an AZ which can be a Xi AZ or another PC. NOTE: As of 5.11 you will need at least 2 PCs deployed (1 for each site).

In PC, search for 'Availability Zones' or navigate to 'Administration' -> 'Availability Zones':

Leap - Connect to Availability Zone

Click on 'Connect to Availability Zone' and select the AZ Type ('Xi' or 'Physical Location' aka PC instance):

Leap - Connect to Availability Zone

Input credentials for PC or Xi and click 'Connect':

Leap - Connect to Availability Zone

The connected AZ will now be displayed and be available.

Configure Protection Policies

In PC, search for ‘Protection Policies’ or navigate to ‘Policies’ -> ‘Protection Policies’:

Leap - Protection Policies

Click on ‘Create Protection Policy’:

Leap - Create Protection Policy

Input details for the name, recovery location, RPO and retention policy (describe previously):

Leap - Protection Policy Inputs

NOTE: for Xi you don’t need select a ‘Target Cluster’:

Leap - Protection Policy Inputs - Xi

Next we’ll select the categories for the policy to apply to:

Leap - Protection Policy Categories

Click ‘Save’ and you will now see the newly created Protection Policy:

Leap - Protection Policies

Configure Recovery Plans

In PC, search for ‘Recovery Plans’ or navigate to ‘Policies’ -> ‘Recovery Plans’:

Leap - Recovery Plans

On the first launch you will be greeted with a screen to create the first Recovery Plan:

Leap - Create Recovery Plan

Select the ‘Recovery Location’ using the drop down:

Leap - Select Recovery Location

NOTE: This can be either a Xi AZ or Physical AZ (PC with corresponding managed clusters).

Input the Recovery Plan name and description and click ‘Next’:

Leap - Recovery Plan - Naming

Next click on ‘Add Entities’ and specify the power on sequence:

Leap - Recovery Plan - Power On Sequence

Search for VMs or Categories to add to each stage:

Leap - Recovery Plan - Power On Sequence

Once the power on sequence looks good with the stages, click ‘Next’:

Leap - Recovery Plan - Power On Sequence

We will now map the network between our source and target environments:

Leap - Recovery Plan - Network Mapping

Failover / Failback Networks

In most cases you will want to use a non-routable or isolated network for your test networks. This will ensure you don't have any issues with duplicate SIDs, arp entries, etc.

Book of Cloud Management

» Download this section as PDF (opens in a new tab/window)

Once you have modernized your infrastructure with Nutanix, the next step is to modernize your operations and automation and establish a cloud operating model. A cloud operating model enables customers of all sizes to build and manage their cloud platform faster and more efficiently. The goal is to spend less time responding to service requests and more time innovating and providing business value.

Here are the four main aspects of cloud management:

- Build
- Operate
- Use
- Govern

Nutanix Cloud Manager provides the tools to establish the cloud operating model in your own datacenter, and integrates with native public cloud services as well. For example, Nutanix Self-Service (formerly Calm) enables you to deploy infrastructure into AWS, Azure, or GCP using native instances (e.g. EC2, Azure VMs, GCP Compute Engine instances). Or you can leverage Nutanix Intelligent Operations to understand VM behavior, create easy-to-use automation for routine tasks, and plan for future workloads. Finally, you can govern cost and security across clouds with Cost Governance and Security Central.

So to summarize, the four services in Nutanix Cloud Manager are:

- Intelligent Operations
- Cost Governance
- Self-Service
- Security Central

This book will dive into these services in more detail. Note that the Cost Governance and Self-Service sections are coming soon.

NOTE: Security Central section is covered in the Book of Network Services.

Intelligent Operations

» Download this section as PDF (opens in a new tab/window)

In the following sections we'll cover some key features that are part of Nutanix Intelligent Operations.

NOTE: Not all features are currently covered. In this section, we currently cover:

- Anomaly Detection
- Capacity Planning
- Low-code/no-code Automation with X-Play

Stay tuned for updates for the rest of the features in Intelligent Operations:

- Reporting
- Non-Nutanix ESXi Monitoring (i.e. monitoring 3-tier ESXi)
- SQL Monitoring
- Application Discovery
- Self-Tuning with Machine Learning (X-Pilot)

Anomaly Detection

In the world of IT operations there is a lot of noise. Traditionally systems would generate a great deal of alerts, events and notifications, often leading to the operator either a) not seeing critical alerts since they are lost in the noise or b) disregarding the alerts/events.

With the Anomaly Detection feature of Intelligent Operations, the system will monitor seasonal trends for time-series data (e.g. CPU usage, memory usage, latency, etc.) and establish a "band" of expected values. Only values that hit outside the "band" will trigger an event / alert. You can see the anomaly events / alerts from any entity or events page.

The following chart shows a lot of I/O and disk usage anomalies as we were performing some large batch loads on these systems:

Anomaly Chart

The following image shows the time-series values for a sample metric and the established "band":

Anomaly Band

This reduces unnecessary alerts as we don’t want alerts for a “normal” state. For example, a database system will normally run at >95% memory utilization due to caching, etc. In the event this drops to say 10% that would be an anomaly as something may be wrong (e.g. database service down).

Another example would be how some batched workloads run on the weekend. For example, I/O bandwidth may be low during the work week, however on the weekends when some batch processes run (e.g. backups, reports, etc.) there may be a large spike in I/O. The system would detect the seasonality of this and bump up the band during the weekend.

Here you can see an anomaly event has occured as the values are outside the expected band:

Anomaly Event

Another topic of interest for anomalies is seasonality. For example, during the holiday period retailers will see higher demand than other times of the year, or during the end of month close.

Anomaly detection accounts for this seasonality and leverages the following periods to compare between micro (daily) and macro (quarterly) trends:

- Daily
- Weekly
- Monthly

You can also set your own custom alerts (also called user-defined alerts), that enables you to alert on behavioral anomalies or static thresholds:

Anomaly Custom Event

Anomaly Detection Algorithm

Nutanix leverages a method for determining the bands called 'Generalized Extreme Studentized Deviate Test'. A simple way to think about this is similar to a confidence interval where the values are between the lower and upper limits established by the algorithm.

The algorithm requires 3 x the granularity (e.g. daily, weekly, monthly, etc.) to calculate the seasonality and expected bands. For example, the following amounts of data would be required to adapt to each seasonality:

- Daily: 3 days
- Weekly: 3 weeks (21 days)
- Monthly: 3 months (90 days)

Twitter has a good resource on how they leverage this which goes into more detail on the logic: LINK

Capacity Planning

To get detailed capacity planning details you can click on a specific cluster under the ‘cluster runway’ section in Prism Central to get more details:

Capacity Planning

This view provides detailed information on cluster runway and identifies the most constrained resource (limiting resource). You can also get detailed information on what the top consumers are as well as some potential options to clean up additional capacity or ideal node types for cluster expansion.

Capacity Planning - Recommendations

X-Play

When we think about our daily activities the more we can automate the better. We are constantly doing this in our daily lives with our routines and technology enables us to do the same in other areas. Nutanix X-Play (pronounced CrossPlay) allows us to automate a common set of activities. However, before diving into the product, let’s first cover what we’re trying to do.

Event driven automation works in the following manner:

event(s) → logic → action(s)

In this scenario there’s some sort of event (or cascading events) that occur which triggers a series or set of actions. A great example of this is IFTTT which takes an event, applies some logic (hence the ‘if this then that’ acronym), then performs some action.

For example, take turning off the lights at home when we leave. If we can program the event (e.g. leaving home / device not present) to trigger the system to turn off all the lights automatically, that makes our lives much simpler.

If we compare this to our IT operations activities we see a similar pattern. An event occurs (e.g. a VM needs more disk space) and then we perform a series of actions (e.g. create a ticket, add storage, close ticket etc.). These repetitive activies are a perfect example of where automation can add value and enable us to focus on more beneficial activities.

With X-Play we can take a series of events / alerts and allow the system to intercept those and perform a series of actions.

To get started navigate to the ‘Plays’ section under ‘Operations’ in the Infrastructure menu:

X-Play - Navigation

This will launch the main X-Play page:

X-Play - Playbooks Overview

Click on ‘Get Started’ to view the current plays and/or create a new one:

X-Play - Playbooks

From here you can create a new playbook by first defining the trigger:

X-Play - Trigger

The following shows an example trigger which is based upon a user-defined alert policy:

X-Play - Trigger - User Defined Alert

Once the trigger has been defined, you now specify a series of actions. The following shows some sample actions:

X-Play - Actions

You then input the details for the action, this shows a sample REST API call:

X-Play - Sample REST Action

REST API Actions and External Systems

X-Play provides a multitude of default actions like sending email, sending a slack message, as well as others like performing a REST API call.

This is critical when we think about interfacing with external systems like a CMDB or other ticketing / automation tools. By using a REST API action we can interface with those to create / resolve tickets, kick off other workflows, etc. This is an extremely powerful option as it enables all systems to be in sync.

For entity / event specific details you can use the ‘parameters’ variables which will give you details about the event, entity and others:

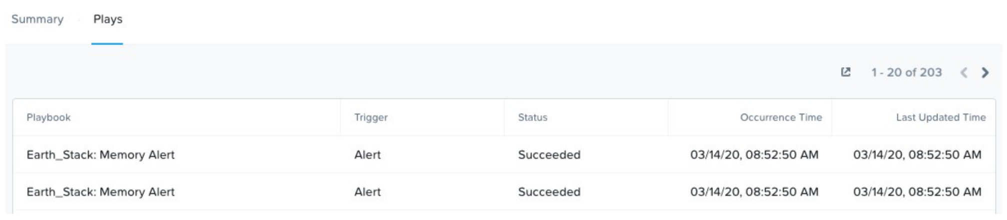
X-Play - Action Parameters

Once complete you can save you play and it will start to execute as defined.

The following shows a sample play whith multiple actions performed:

X-Play - Sample Playbook

The plays tab will show execution time of the play and status:



Playbook	Trigger	Status	Occurrence Time	Last Updated Time
Earth_Stack: Memory Alert	Alert	Succeeded	03/14/20, 08:52:50 AM	03/14/20, 08:52:50 AM
Earth_Stack: Memory Alert	Alert	Succeeded	03/14/20, 08:52:50 AM	03/14/20, 08:52:50 AM

X-Play - Plays Executed

For some example playbooks that you can import into your own instance, check out the Playbooks Library on nutanix.dev!

Also, check out Getting Started with Nutanix X-Play for more examples and references.

Book of Cloud Native Services

» Download this section as PDF (opens in a new tab/window)

The CNCF defines cloud native as “a set of technologies that empower organizations to build and run scalable applications in modern, dynamic environments such as public, private and hybrid clouds”. The primary technologies driving this shift to application modernization include containers, microservices, and Kubernetes.

We believe Nutanix hyperconverged infrastructure (HCI) is the ideal infrastructure foundation for cloud native workloads running on Kubernetes at scale. Nutanix provides platform mobility giving you the choice to run workloads on both your Nutanix private cloud as well as the public cloud. The Nutanix architecture was designed keeping hardware failures in mind, which offers better resilience for both Kubernetes platform components and application data. With the addition of each HCI node, you benefit from the scalability and resilience provided to the Kubernetes compute nodes. Equally important, there is an additional storage controller that deploys with each HCI node which results in better storage performance for your stateful containerized applications.

The Nutanix Cloud Platform provides a built-in turnkey Kubernetes experience with Nutanix Kubernetes Engine (NKE). NKE is an enterprise-grade offering that simplifies the provisioning and lifecycle management of multiple clusters. Nutanix is about customer choice, customers can run their preferred distribution such as OpenShift, Rancher, Anthos, and others, due to the superior full stack resource management. Read more on that in the chapters below.

Nutanix Unified Storage provides persistent and scalable software-defined storage to the Kubernetes clusters. These include block and file storage via the Nutanix CSI driver as well as S3-compatible object storage. Furthermore, with Nutanix Database Service, you can provision and operate databases at scale.

The following chapters will cover these in more detail:

Nutanix Kubernetes Engine (formerly Nutanix Karbon)

» Download this section as PDF (opens in a new tab/window)

Nutanix Kubernetes Engine (NKE) coupled with Prism Central-based Kubernetes Management is the Nutanix certified enterprise Kubernetes management solution that enables turnkey provisioning, operations, and lifecycle management of Kubernetes.

Supported Configurations

The solution is applicable to the configurations below:

Core Use Case(s):

- Containers
- Microservices
- Application modernization

Management interfaces(s):

- Kubernetes Management in Prism Central

Note

Starting with NKE 2.8 and Prism Central 2023.1.0.1, Kubernetes Engine has been renamed to Kubernetes Management.

Supported Environment(s):

- Hypervisors:
    - AHV
- Locations:
    - On-premises:
        - Owned
        - (Managed) Service Providers

Supported node OS image(s):

- CentOS Linux-based provided by Nutanix

Upgrades:

- Included in LCM as Karbon

Compatible Features:

- Lifecycle operations
- Kubernetes RBAC
- Cluster expansion
- Multi-cluster management
- Node pool
- GPU pass-through

Enable Kubernetes Management using Prism Central marketplace. Any Nutanix AOS cluster registered with a Kubernetes Management-enabled PC can be used as a target for provisioning Kubernetes clusters.

NKE Architecture

NKE runs as a containerized service in Prism Central. When Kubernetes Management is enabled on a PC, two containers are provisioned under the covers: the *karbon-core* container and the *karbon-ui* container.

- *Karbon-core* is responsible for the lifecycle of Kubernetes clusters. Tasks such as provisioning, node OS upgrades, cluster expansion, and others, are performed by this container.
- *Karbon-ui* is responsible for providing an intuitive console via Prism Central. From this integrated UI the IT admins have full control over the entire Kubernetes landscape managed by the NKE instance.

Air-gapped environments

NKE can be enabled in air-gapped environments too (see NKE Airgap for more information)

Kubernetes Cluster Configurations OS Images

Nutanix provides a CentOS image for installing and scaling Kubernetes nodes. New OS image versions are periodically released including patches to fix vulnerabilities. For a list of supported OS image versions, check the NKE Release Notes.

Operating System Images

Bringing your own OS image is not supported.

Nutanix Guest Tools

Do not install Nutanix Guest Tools (NGT) or any other services on Kubernetes nodes.

Compute

The recommended configurations include two options: *development* cluster and *production* cluster.

- The *development* cluster option does not have a highly available control plane. In the event of a control plane node going offline, the Kubernetes cluster will be impacted.

  The minimum cluster size for development is three nodes:

  - The control plane is divided into two nodes, the etcd node, and the Kubernetes control plane node.
  - The worker node pool has a single node that can be scaled out up to 100 nodes (see NKE Configuration Maximums for more information).

- The *production* cluster option does have a highly available control plane. There is no single-point-of-failure with this configuration option.

  The minimum cluster size for production is eight nodes:

  - The control plane is divided into five nodes, three etcd nodes - can scale up to five nodes -, and two Kubernetes control plane nodes. The Kubernetes control plane nodes can operate as active-passive (two nodes), or active-active (up to five nodes). For the latter, an external load balancer is required.
  - The worker node pool has a minimum of three nodes that can be scaled out up to 100 nodes.

Affinity Policies

After the cluster configuration, the anti-affinity rules are automatically created to ensure that control plane and worker node VMs run on different AHV hosts to reduce single points of failure.

Pro tip

Resource recommendations

Networking

There are a total of three networks required by a Kubernetes cluster which can be grouped into virtual machines network and Kubernetes networks.

- Virtual machines network or node network. This has to be allocated either by DHCP (development clusters only) or via a Managed network that is IPAM enabled (with associated domain settings and IP address pools). Production configuration requires additional static IP addresses for active-passive, and active-active modes.

- Kubernetes networks. A cluster requires a minimum of two classless inter-domain routing (CIDR) ranges, one for the Kubernetes Services network, and another for the Kubernetes Pods network.

  NKE supports two container network interface (CNI) providers for the Kubernetes networks: Flannel and Calico.

  - *Flannel*. NKE uses the VXLAN mode. Changing the mode to host-gw is possible, but unsupported.
  - *Calico*. NKE uses the Direct mode. Changing the mode to IP in IP or VXLAN is possible, but unsupported.

Pro tip

You can leave the service CIDR and pod CIDR ranges as default, but the ranges must not overlap with each other or with an existing network in your data center if a pod in the cluster will require access to that external network.

Optionally, you can also specify an additional secondary worker node network to optimize storage traffic. The purpose of the secondary network is to segment CSI I/O traffic for optimization so that data travels through this network (see NKE Network Segmentation for more information).

Storage

When deploying a Kubernetes cluster, the Nutanix container storage interface (CSI) driver is also deployed along with it.

A default StorageClass is created as well during the deployment, which uses Nutanix Volumes. This is required by the included add-ons such as Prometheus for monitoring, and EFK (Elasticsearch, Fluent Bit, and Kibana) logging stack, to store metrics and logs. After deployment, more storage classes can be added using the same CSI driver (see Nutanix CSI Release Notes for more information).

Apart from Nutanix Volumes, you can also create a StorageClass for file storage using Nutanix Files. Depending on what storage backend is configured in a StorageClass, different access modes are supported when creating a PersistentVolumeClaim.

Storage backend	**ReadWriteOnce RWO**	**ReadOnlyMany ROX**	**ReadWriteMany RWX**	**ReadWriteOncePod RWOP**
Volumes	√	√	-	-
Files	√	√	√	-

Access modes supported by CSI driver and storage backend.

Security

Access and Authentication

There are two components to keep in mind when it comes to access and authentication: NKE in PC, and an NKE-enabled Kubernetes cluster.

- NKE uses Prism Central authentication and RBAC. Nutanix requires configuring NKE users to a directory service in Prism Central like Microsoft Active Directory. Users can access the NKE console and perform certain tasks based on the assigned role.

  A member of the *User Admin* role in PC has full access to NKE and its functionalities.

  A member of the *Cluster Admin* role or *Viewer* role can only download kubeconfig.

- An NKE-enabled Kubernetes cluster out-of-the-box uses Prism Central for authentication and maps the PC role *User Admin* with the Kubernetes role *cluster-admin*.

  From an authentication perspective, the Kubernetes cluster sends authentication requests to NKE which uses PC directory services. This means that you can authenticate your users against Active Directory out-of-the-box.

  From the RBAC standpoint, the PC role *User Admin* maps with the Kubernetes super-admin role named *cluster-admin*. This means that a user member of the *User Admin* role in PC is a super-user in all Kubernetes clusters managed by the NKE instance. On the other hand, the PC roles *Cluster Admin* and *Viewer* do not have a mapping with a Kubernetes role. This means that a user member of any of these two roles can download the kubeconfig from NKE, but not perform any action at the Kubernetes level. A super-admin user will have to create the correct role mapping inside Kubernetes.

Blog

Providing RBAC for your Karbon Kubernetes Clusters

Note that the kubeconfig generated by NKE is valid for 24-hours, after which the user will have to request a new kubeconfig file. This can be done using the NKE GUI, CLI, API, or this kubectl plug-in (recommended).

Nodes

The SSH access to the Kubernetes nodes is locked down using an ephemeral certificate - available in the NKE console, which expires after 24-hours. Installing software or changing settings in the node OS is unsupported, changes are not persistent during upgrades or when scaling out a node pool. The only reason for accessing the nodes via SSH is for troubleshooting at the discretion of Nutanix support.

Private Registry

By default, NKE does not add additional container image registries to Kubernetes clusters. To use your own images for container deployment, add a private registry to NKE and configure private registry access for the intended Kubernetes clusters (see Configuring a Private Registry for more information).

CIS Benchmark for Kubernetes

Nutanix has evaluated NKE-enabled Kubernetes cluster against the CIS Kubernetes Benchmark-1.6. You can verify compliance through Kube Bench, an automated open-source tool available on GitHub. See the report CIS Benchmark for Kubernetes (requires a Nutanix account).

Add-ons

NKE add-ons are open source software extensions that provide additional features to your deployment.

Nutanix Kubernetes Engine includes the following add-ons:

- An infra logging add-on powered by Elasticsearch, Fluent Bit, and Kibana (EFK)
- A monitoring add-on powered by Prometheus

These add-ons are for cluster internal use only. Their configuration is not designed for supporting the data generated by the applications running on the Kubernetes cluster. For collecting logs and metrics for the containerized applications, deploy dedicated instances of EFK and Prometheus, or re-use existing ones available in your environment (see Enabling Log Forwarding for more information).

Logging

The logging stack aggregates all the operating system and infrastructure logs from the Kubernetes nodes. The Kibana dashboard is accessible via the NKE console.

Note

Starting with NKE 2.6, Elasticsearch and Kibana are not enabled by default during cluster deployment (see Enabling Infra Logging for more information).

Monitoring

The Kubernetes clusters have the Prometheus operator installed and one instance of it deployed for collecting infrastructure metrics. Additional Prometheus instances can be deployed using the operator, for example, for application monitoring (see the blog Monitoring Application Metrics With Nutanix Karbon for more details).

Note

Starting with NKE 2.8, you can disable the monitoring stack only during cluster provisioning. Disabling the monitoring stack also disables the alerts in NKE UI. This setting cannot be modified later.

SMTP-based alert forwarding to an e-mail address can be enabled (see Enabling Alert Forwarding for more information).

Kubernetes namespaces

Any initial Kubernetes cluster starts with four namespaces: *default*, *kube-node-lease*, *kube-public*, and *kube-system* (see Initial Namespaces for more information).

NKE includes an additional namespace called **ntnx-system**. This namespace always contains at least the Nutanix CSI plug-in deployment and the fluentbit daemonset.

Get ntnx-system namespace applications

```
kubectl -n ntnx-system get deployment,daemonset,statefulset
```

This command outputs the current services enabled in the NKE cluster. The default output will display at least:

```
NAME                                          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/csi-snapshot-controller       1/1     1            1           15d
deployment.apps/csi-snapshot-webhook          1/1     1            1           15d
deployment.apps/kubernetes-events-printer     1/1     1            1           15d
deployment.apps/nutanix-csi-controller        1/1     1            1           15d

NAME                               DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR   AGE
daemonset.apps/fluent-bit          2         2         2       2            2           <none>          15d
daemonset.apps/nutanix-csi-node    1         1         1       1            1           <none>          15d
```

In addition to these resources, if enabled, you can also find the deployments, daemonsets, and statefulsets for the monitoring and logging stack.

```
NAME                                          READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/blackbox-exporter             1/1     1            1           15d
deployment.apps/csi-snapshot-controller       1/1     1            1           15d
deployment.apps/csi-snapshot-webhook          1/1     1            1           15d
deployment.apps/kibana-logging                1/1     1            1           4m54s
deployment.apps/kube-state-metric             1/1     1            1           15d
deployment.apps/kubernetes-events-printer     1/1     1            1           15d
deployment.apps/nutanix-csi-controller        1/1     1            1           15d
deployment.apps/prometheus-adapter            1/1     1            1           15d
deployment.apps/prometheus-operator           1/1     1            1           15d

NAME                               DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE   NODE SELECTOR            AGE
daemonset.apps/fluent-bit          2         2         2       2            2           <none>                   15d
daemonset.apps/node-exporter       2         2         2       2            2           kubernetes.io/os=linux   15d
daemonset.apps/nutanix-csi-node    1         1         1       1            1           <none>                   15d

NAME                                       READY   AGE
statefulset.apps/alertmanager-main         1/1     15d
statefulset.apps/elasticsearch-logging     1/1     4m54s
statefulset.apps/prometheus-k8s            1/1     15d
```

Warning

Do not delete the ntnx-system namespace.

Lifecycle management

There are two different types of NKE upgrades:

- NKE version upgrades using the Life Cycle Management feature.
- Kubernetes cluster upgrades for node OS image, and Kubernetes version.

NKE upgrade via LCM

To check the current version of NKE or to upgrade to later versions, perform the inventory check in Prism Central using LCM. LCM upgrades the following NKE components:

- NKE core (karbon-core container)
- NKE UI (karbon-ui container)

NKE Upgrades

Be aware when upgrading to a latest version of NKE, that all the Kubernetes clusters must be running or upgraded first to a supported version by the target NKE. Check the Nutanix portal for updated supported versions.

Kubernetes cluster upgrades

There are two aspects when it comes to upgrading a Kubernetes cluster:

- Node operating system upgrades
- Kubernetes + add-ons version upgrade

Kubernetes Cluster Upgrades

Be aware that node OS or Kubernetes version upgrades can be disruptive depending on your Kubernetes cluster type, development vs. production.

Node OS upgrade

When a node OS image upgrade is available, NKE displays an option to download the new image in the **OS Images** tab. NKE also displays an **Upgrade Available** icon next to the cluster in the Clusters view.

Kubernetes + add-ons version upgrade

Clusters that have a Kubernetes version eligible for an upgrade display the **Upgrade Available** icon in the table. As a part of the upgrade process, it will upgrade the Kubernetes version as well as any upgrade available for the installed add-ons.

NKE CLI and API

NKE CLI

The NKE CLI, **karbonctl**, gives users the ability to execute lifecycle management tasks for NKE and Kubernetes clusters. Certain advanced tasks can be done using karbonctl only.

To use **karbonctl** you have to SSH into a Prism Central instance. The path for the binary is */home/nutanix/karbon/karbonctl*

Some common tasks you can run with **karbonctl** are:

- Configuring airgap deployment
- Configuring GPU support
- Rotating certificates
- Enabling/disabling alert forwarding
- Configuring a private registry
- Renewing the kubeconfig file

Get karbonctl options

This command outputs all the available options. The following output is for NKE 2.8.0:

```
Karbonctl is a command line utility to manage your k8s clusters

Usage:
  karbonctl [command]

Available Commands:
  airgap            Used for Karbon Airgap configuration
  cluster           Used for k8s cluster specific operations
  completion        generate the autocompletion script for the specified shell
  help              Help about any command
  k8s               Used for getting the list of available k8s packages from the Nutanix Portal
  karbon-agent      Used for Karbon agent specific operations
  karbon-management Used for Advanced Kubernetes Managment specific operations
  login             Generate a karbonctl configuration to allow passwordless authentication to Karbon
  os-image          Used for OS image management
  registry          Used for private registry operations
  version           Output of the karbonctl version information

Flags:
      --config string        Karbonctl configuration file path (default "/home/nutanix/.karbon/config/karbonctl.yaml")
  -h, --help                 help for karbonctl
      --output string        Supported output formats: ['default', 'json'] (default "default")
      --pc-ip string         Prism Central IP (default "127.0.0.1")
      --pc-password string   Password of the user in Prism Central
      --pc-port int          Prism port on the Prism Central VM (default 9440)
      --pc-username string   Username of the user in Prism Central
  -t, --toggle               Help message for toggle

Use "karbonctl [command] --help" for more information about a command.
```

NKE API

The NKE API lets users programmatically run management task for NKE and Kubernetes clusters. The API documentation is available at https://www.nutanix.dev/api_references/nke.

Partner Kubernetes Distributions

» Download this section as PDF (opens in a new tab/window)

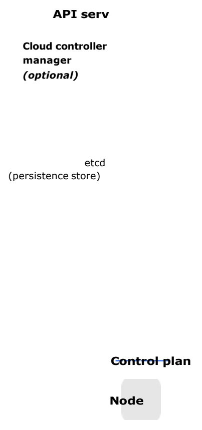
The Nutanix Cloud Platform is an ideal solution for running any certified Kubernetes distribution. Nutanix brings an enterprise-class platform with all the resources needed to successfully run your modern applications at scale.

Kubernetes distributions require compute, network, and storage. With Nutanix, these resources are easily accessible to IT administrators and developers to run their preferred Kubernetes distributions. Several leading Kubernetes distribution providers have certified their solutions for use with Nutanix including Red Hat OpenShift, Google Anthos, and several others. View our supported partner software solutions online on the Nutanix support portal.

Kubernetes architecture

All Kubernetes distributions have a base architecture with the following components at a minimum:

- etcd - the key-value store for storing all Kubernetes cluster configuration data, state data, and metadata.
- Kubernetes control plane - this includes the Kubernetes API server and other components for scheduling pods and detecting and responding to cluster events.
- Kubernetes worker nodes - Machines that host the application workloads.



In addition, these components run on all Kubernetes control plane and worker nodes.

- Kubelet
- Kube-proxy
- Container runtime

Detailed information on these components can be found in the Kubernetes documentation.

Red Hat OpenShift Container Platform

There are multiple supported installation methods for OpenShift Container Platform on Nutanix. OCP is available from the Nutanix Marketplace to be deployed and consumed directly as an application.
From OCP 4.11 and onwards, the full stack automated installer-provisioned infrastructure method is available.
From OCP 4.12 and onwards, the Assisted Installer method is also supported.

In addition, the Nutanix CSI Operator that provides persistent storage is readily available from the OpenShift OperatorHub.

The Nutanix Validated Design for OpenShift on Nutanix is available, which is a joint validation of software, hardware and services. These are complete designs that undergone in-depth functional and scale tests to meet enterprise requirements. These solutions' modularity and pre-built nature ensure that they can be easily used in production.

Amazon EKS-A

EKS Anywhere is an enterprise-class solution for deploying Kubernetes clusters on premises with simplified cluster creation and integrated 3rd party software. The Nutanix Cloud Platform and the AHV hypervisor provide supported infrastructure for EKS-A.

Cluster API Provider Nutanix (CAPX)

Cluster API is a Kubernetes sub-project that brings declarative, Kubernetes-style APIs to cluster creation and management. Nutanix provides an implementation of Cluster API for Nutanix Cloud Infrastructure known as CAPX. The Nutanix CSI driver is fully supported on CAPI/CAPX clusters.

Google Anthos

Google Anthos can be deployed on the Nutanix Cloud Infrastructure running Nutanix AHV hypervisor using the Anthos clusters on bare metal installation method.

Rancher

Rancher's Kubernetes management tool can manage any Kubernetes cluster running on Nutanix. Rancher also supports provisioning Rancher Kubernetes Engine (RKE) clusters on the Nutanix platform.
After activating Rancher's built-in Nutanix node driver and creating a node template from it, the RKE cluster can be provisioned with node pools.

Azure Arc

Nutanix Kubernetes Engine (NKE) is validated for running Azure Arc-enabled Kubernetes clusters. Via Azure Arc, Azure customers can seamlessly extend Azure's data services and management capabilities to on-prem Kubernetes clusters.

Kubermatic

With the Nutanix and Kubermatic partnership, Nutanix is a supported infrastructure provider for the Kubermatic Kubernetes Platform.

Book of AI/ML

» Download this section as PDF (opens in a new tab/window)

AI is used almost everywhere in our daily lives. For example, your smartphone can recognize images and voices, translate languages, identify music, and more. What about AI in your datacenter? How can it help you, and more importantly, how can you get there?

The enterprise data explosion - all of the databases, conversations, logs, metrics, etc. that get generated in an organization - is fueling the demand for AI to extract actionable insights for making better business decisions. All industries can benefit from actionable insight generation from AI. Until 2021, most enterprise AI applications were discriminative in nature, focusing on the computation of a decision boundary for classification (for example, text or speech recognition, object detection) and regression tasks (for example, sales forecasting, stock price prediction). Since 2022, there has been a sweeping change in the enterprise AI landscape with increasing adoption of generative AI-based applications triggered by the popularity of chatGPT and diffusion models. Unlike discriminative AI, generative AI generates a distribution from sample data (i.e. it creates new data).

The Nutanix Cloud Platform provides the ideal stack for running your AI/ML workloads, especially for generative AI applications, which take massive computational resources. The following chapters will cover this in more detail.

AI on Nutanix

» Download this section as PDF (opens in a new tab/window)

There is a growing need for infrastructure that can span across edge, core, and/or the public cloud. For example, data can be generated and processed locally at several far-edge nodes, then sent back to the core datacenter for aggregate decision-making and fine-tuning the model.

Regardless of what is deployed, you need a consistent operating model across the tiers. Nutanix Cloud Platform can run at all locations - whether it's a single-node edge device, a multi-node cluster in your core datacenter, or in the public cloud with NC2. This provides a unified cloud operating model that enables the simple, consistent operation of your AI/ML platform.

NCP supports a variety of hardware and NVIDIA GPU cards, and you can run your workloads on Nutanix Kubernetes Engine, Red Hat Openshift, or on VMs.

A key aspect of running AI/ML workloads is managing the machine learning lifecycle. The term MLOps is a compound of Machine Learning and Operations, and is analogous to DevOps, in that it aims to automate, orchestrate, and deploy models in a consistent and efficient manner. Kubeflow is a popular solution for MLOps, and the Nutanix Cloud Platform is validated for Kubeflow, enabling consistent MLOps across the tiers.

For more information on running AI/ML on Nutanix, check out the following resources:

- Nutanix.com AI/ML Solutions Page
- Nutanix Validated Design for the AI-enabled Edge
- Nutanix.dev blog posts
- DEMO: Running a Chatbot on Nutanix Cloud Platform

GPT-in-a-Box

» Download this section as PDF (opens in a new tab/window)

GPT-in-a-Box is a turnkey AI solution for organizations wanting to implement GPT capabilities while maintaining control of their data and applications. It includes everything needed to build AI-ready infrastructure, including:

- Nutanix Cloud Platform infrastructure on GPU-enabled server nodes
- Nutanix Files and Object storage for running and fine-tuning GPT models
- Open source software to deploy and run AI workloads, including PyTorch and Kubeflow
- Support for a curated set of LLMs (including Llama2, Falcon, and MPT)

For more information, please visit https://www.nutanix.com/solutions/ai

Addendum

Release Notes & Change Log

Please use this document to stay up to date on Nutanix Bible changes, updates and additions. This document will be updated each time a change is made to the Nutanix Bible content.

Change Log

Date	Summary	Details
December 21, 2023	Updates to Book of AHV and Book of Network Services	• Added information about the network controller and support for VMs in network controller-enabled VLANs. • Added information about Flow Network Security Next-Gen.
December 15, 2023	Updates to Book of AOS and Book of AHV	• Added information on NGT install packages in Book of AOS • Added sections on Acropolis Dynamic Scheduler (ADS), vTPM, Live Migration, and Generation ID in Book of AHV
November 29, 2023	Updates to Book of Prism	• Added Microservices Infrastructure and PC Backup & Restore chapters.
October 30, 2023	Updates to Book of Cloud Native	• Updated Partner Kubernetes Distributions section with additional

Date	Summary	Details
		information on the cloud native partnerships.
September 6, 2023	Updates to Book of NC2 for AOS 6.7	• Added section on Custom AWS Security Groups • Added section on Native Backup with Nutanix Cluster Protection in AWS • Added section on Flow Virtual Networking Gateway HA in Azure
August 17, 2023	Added Book of AI/ML	• Added Nutanix on AI chapter • Added GPT-in-a-Box chapter
June 30, 2023	Updates to the Book of Cloud Native	• Updated Nutanix Kubernetes Engine section with the following: ◦ Update diagrams ◦ Update to NKE 2.8 and PC 2023.1.0.1 ◦ Update links to more details ◦ Include note about NGT ◦ Include affinity policy ◦ Include secondary network ◦ Include private registry ◦ Update add-ons information ◦ Include ntnx-system namespace
June 20, 2023	Added Book of APIs	• Moved API content from Book of Prism to Book of APIs
June 9, 2023	Updates to the Book of Network Services	• Added Flow Virtual Networking BGP. • Added detail to Flow Virtual Networking VPC Subnet Extension.
June 6, 2023	Multiple Updates	• Updated Storage (formerly Distributed Storage Fabric) chapter in Book of AOS. • Updated Security Central chapter in Book of Network Services, addition of the Investigate section
June 2, 2023	Update to the Book of NC2	• Removed "private-preview" label from Azure as it is now GA.
May 24th, 2023	Update Cluster Maximums	• Updated the cluster maximums for ESXi, AHV and Hyper-V.
May 18th, 2023	Added Book of Cloud Management, updates to Nutanix Cloud Clusters section	• Moved Nutanix Intelligent Operations content from Book of Prism to Cloud Management • Updated screenshots • Azure support is now GA for NC2
April 5th, 2023	Nutanix Rebrand	• Rolled out new version of The Nutanix Bible, incorporating Nutanix branding changes launched April 2023. • Nutanix Bible content unchanged.

Date	Summary	Details
March 10th, 2023	Updates to the Book of Network Services - Flow Virtual Networking	• Added details and diagrams for multiple external networks and Network Gateway VMs to the Book of Network Services, Flow Virtual Networking.
February 28th, 2023	Updated images and Book of Basics product info	• Updated to latest product images and product info in Book of Basics, including AOS Scale-Out Storage, Nutanix Cloud Manager (NCM), Nutanix Unified Storage Services and Nutanix Database Services.
November 14, 2022	Updates to the Book of Nutanix Cloud Clusters - AWS	• Updated placement policy
September 9, 2022	Updates to the Book of Storage Services - Files	• Updated diagram and removed mention of MPIO
July 28, 2022	Fixed description of curator_cli command in Book of AOS	• Book of AOS contained duplicated description for two different curator_cli usages. Correct descriptions added.
July 26, 2022	Updates to the Book of Storage Services - Objects	• Added clarity around use cases,and re-worded use of RocksDB
July 25, 2022	Updates to the Book of Storage Services under the Objects section	• Updates to use cases, protocols, constructs and note for open source disclosure
June 1st, 2022	Updates to the Book of Nutanix Cloud Clusters	• New NC2 on Azure Architecture section for the private and upcoming public preview.
May 20, 2022	Updates to the Book of Network Services	• Updates to the Security Central section - architecture and overview
May 9, 2022	Updates to the Book of AHV	• Updates to the AHV Architecture section - VM templates, Memory Overcommit, VM Affinity policies
May 3, 2022	New Book of Cloud Native	• Book of Cloud Native deployed
April 15, 2022	Updates to the Book of AOS	• Replaced list of VSS supported OS's with link to the NGT / VSS table on portal • Changed text to point readers to the correct book/chapter in the bible • Resilient Capacity section added to Book of AOS
April 8, 2022	Multiple updates	• Various updates to I/O Path section, including new vDisk Sharding section • Removed Cloud Connect section in the book of AOS. • Update Nutanix Clusters sections to reflect name change to NC2

Date	Summary	Details
March 25, 2022	Updates to Nutanix Bible images	• All images now show a larger version on mouse click
March 18, 2022	Updates to the Book of AOS	• Stargate I/O Logic and Tiers to include Optane Tiering
March 16, 2022	Updates to the Book of Network Services	• Addition of Security Central and Flow Virtual Networking sections
February 18, 2022	Updated resource links	• Updated resources links in APIs & Interfaces section
December 3rd, 2021	Nutanix Bible Classic PDF added	• Published PDF version of Nutanix Bible Classic view
November 19th, 2021	Release notes added	• Release notes and change log document released • First collection of downloadable PDF sections released

A Brief Lesson in History

» Download this section as PDF (opens in a new tab/window)

A brief look at the history of infrastructure and what has led us to where we are today.

The Evolution of the Datacenter

The datacenter has evolved significantly over the last several decades. The following sections will examine each era in detail.

The Era of the Mainframe

The mainframe ruled for many years and laid the core foundation of where we are today. It allowed companies to leverage the following key characteristics:

- Natively converged CPU, main memory, and storage
- Engineered internal redundancy

But the mainframe also introduced the following issues:

- The high costs of procuring infrastructure
- Inherent complexity
- A lack of flexibility and highly siloed environments

The Move to Stand-Alone Servers

With mainframes, it was very difficult for organizations within a business to leverage these capabilities which partly led to the entrance of pizza boxes or stand-alone servers. Key characteristics of stand-alone servers included:

- CPU, main memory, and direct-attached storage (DAS)
- Higher flexibility than the mainframe
- Accessed over the network

These stand-alone servers introduced more issues:

- Increased number of silos
- Low or unequal resource utilization
- The server became a single point of failure (SPOF) for both compute AND storage

Centralized Storage

Businesses always need to make money and data is a key piece of that puzzle. With direct-attached storage (DAS), organizations either needed more space than was locally available, or data high availability (HA) where a server failure wouldn't cause data unavailability.

Centralized storage replaced both the mainframe and the stand-alone server with sharable, larger pools of storage that also provided data protection. Key characteristics of centralized storage included:

- Pooled storage resources led to better storage utilization
- Centralized data protection via RAID eliminated the chance that server loss caused data loss
- Storage were performed over the network

Issues with centralized storage included:

- They were potentially more expensive, however data is more valuable than the hardware
- Increased complexity (SAN Fabric, WWPNs, RAID groups, volumes, spindle counts, etc.)
- They required another management tool / team

The Introduction of Virtualization

At this point in time, compute utilization was low and resource efficiency was impacting the bottom line. Virtualization was then introduced and enabled multiple workloads and operating systems (OSs) to run as virtual machines (VMs) on a single piece of hardware. Virtualization enabled businesses to increase utilization of their pizza boxes, but also increased the number of silos and the impacts of an outage. Key characteristics of virtualization included:

- Abstracting the OS from hardware (VM)
- Very efficient compute utilization led to workload consolidation

Issues with virtualization included:

- An increase in the number of silos and management complexity
- A lack of VM high-availability, so if a compute node failed the impact was much larger
- A lack of pooled resources
- The need for another management tool / team

Virtualization Matures

The hypervisor became a very efficient and feature-filled solution. With the advent of tools, including VMware vMotion, HA, and DRS, users obtained the ability to provide VM high availability and migrate compute workloads dynamically. The only caveat was the reliance on centralized storage, causing the two paths to merge. The only down turn was the increased load on the storage array before and VM sprawl led to contention for storage I/O.

Key characteristics included:

- Clustering led to pooled compute resources
- The ability to dynamically migrate workloads between compute nodes (DRS / vMotion)
- The introduction of VM high availability (HA) in the case of a compute node failure
- A requirement for centralized storage

Issues included:

- Higher demand on storage due to VM sprawl
- Requirements to scale out more arrays creating more silos and more complexity
- Higher $ / GB due to requirement of an array

- The possibility of resource contention on array
- It made storage configuration much more complex due to the necessity to ensure:
  - VM to datastore / LUN ratios
  - Spindle count to facilitate I/O requirements

Solid State Disks (SSDs)

SSDs helped alleviate this I/O bottleneck by providing much higher I/O performance without the need for tons of disk enclosures. However, given the extreme advances in performance, the controllers and network had not yet evolved to handle the vast I/O available. Key characteristics of SSDs included:

- Much higher I/O characteristics than traditional HDD
- Essentially eliminated seek times

SSD issues included:

- The bottleneck shifted from storage I/O on disk to the controller / network
- Silos still remained
- Array configuration complexity still remained

In Comes Cloud

The term cloud can be very ambiguous by definition. Simply put it's the ability to consume and leverage a service hosted somewhere provided by someone else.

With the introduction of cloud, the perspectives IT, the business and end-users have shifted.

Business groups and IT consumers require IT provide the same capabilities of cloud, its agility and time to value. If not, they will go directly to cloud which causes another issue for IT: data security.

Core pillars of any cloud service:

- Self-service / On-demand
  - Rapid time to value (TTV) / little barrier to entry

- Service and SLA focus
  - Contractual guarantees around uptime / availability / performance

- Fractional consumption model
  - Pay for what you use (some services are free)

Cloud Classifications

Most general classifications of cloud fall into three main buckets (starting at the highest level and moving downward):

- Software as a Service (SaaS)
  - Any software / service consumed via a simple url
  - Examples: Workday, Salesforce.com, Google search, etc.

- Platform as a Service (PaaS)
  - Development and deployment platform
  - Examples: Amazon Elastic Beanstalk / Relational Database Services (RDS), Google App Engine, etc.

- Infrastructure as a Service (IaaS)
  - VMs/Containers/NFV as a service
  - Examples: Amazon EC2/ECS, Microsoft Azure, Google Compute Engine (GCE), etc.

Shift in IT focus

Cloud poses an interesting dilemma for IT. They can embrace it, or they can try to provide an alternative. They want to keep the data internal, but need to allow for the self-service, rapid nature of cloud.

This shift forces IT to act more as a legitimate service provider to their end-users (company employees).

The Importance of Latency

The figure below characterizes the various latencies for specific types of I/O:

Item	**Latency**	**Comments**
L1 cache reference	0.5 ns	
L2 cache reference	7 ns	14x L1 cache
DRAM access	100 ns	20x L2 cache, 200x L1 cache
3D XPoint based NVMe SSD read	10,000 of ns (expected)	10 us or 0.01 ms
NAND NVMe SSD R/W	20,000 ns	20 us or 0.02 ms
NAND SATA SSD R/W	50,000-60,000 ns	50-60 us or 0.05-0.06 ms
Read 4K randomly from SSD	150,000 ns	150 us or 0.15 ms
P2P TCP/IP latency (phy to phy)	150,000 ns	150 us or 0.15 ms
P2P TCP/IP latency (vm to vm)	250,000 ns	250 us or 0.25 ms
Read 1MB sequentially from memory	250,000 ns	250 us or 0.25 ms
Round trip within datacenter	500,000 ns	500 us or 0.5 ms
Read 1MB sequentially from SSD	1,000,000 ns	1 ms, 4x memory
Disk seek	10,000,000 ns or 10,000 us	10 ms, 20x datacenter round trip
Read 1MB sequentially from disk	20,000,000 ns or 20,000 us	20 ms, 80x memory, 20x SSD
Send packet CA -> Netherlands -> CA	150,000,000 ns	150 ms

(credit: Jeff Dean, https://gist.github.com/jboner/2841832)

The table above shows that the CPU can access its caches at anywhere from ~0.5-7ns (L1 vs. L2). For main memory, these accesses occur at ~100ns, whereas a local 4K SSD read is ~150,000ns or 0.15ms.

If we take a typical enterprise-class SSD (in this case the Intel S3700 - SPEC), this device is capable of the following:

- Random I/O performance:
  - Random 4K Reads: Up to 75,000 IOPS
  - Random 4K Writes: Up to 36,000 IOPS

- Sequential bandwidth:
  - Sustained Sequential Read: Up to 500MB/s
  - Sustained Sequential Write: Up to 460MB/s

- Latency:
  - Read: 50us
  - Write: 65us

Looking at the Bandwidth

For traditional storage, there are a few main types of media for I/O:

- Fiber Channel (FC)
  - 4-, 8-, 16- and 32-Gb

- Ethernet (including FCoE)
    - 1-, 10-Gb, (40-Gb IB), etc.

For the calculation below, we are using the 500MB/s Read and 460MB/s Write BW available from the Intel S3700.

The calculation is done as follows:

```
numSSD = ROUNDUP((numConnections * connBW (in GB/s))/ ssdBW (R or W))
```

NOTE: Numbers were rounded up as a partial SSD isn't possible. This also does not account for the necessary CPU required to handle all of the I/O and assumes unlimited controller CPU power.

Network BW		**SSDs required to saturate network BW**	
Controller Connectivity	**Available Network BW**	**Read I/O**	**Write I/O**
Dual 4Gb FC	8Gb == 1GB	2	3
Dual 8Gb FC	16Gb == 2GB	4	5
Dual 16Gb FC	32Gb == 4GB	8	9
Dual 32Gb FC	64Gb == 8GB	16	19
Dual 1Gb ETH	2Gb == 0.25GB	1	1
Dual 10Gb ETH	20Gb == 2.5GB	5	6

As the table shows, if you wanted to leverage the theoretical maximum performance an SSD could offer, the network can become a bottleneck with anywhere from 1 to 9 SSDs depending on the type of networking leveraged

The Impact to Memory Latency

Typical main memory latency is ~100ns (will vary), we can perform the following calculations:

- Local memory read latency = 100ns + [OS / hypervisor overhead]
- Network memory read latency = 100ns + NW RTT latency + [2 x OS / hypervisor overhead]

If we assume a typical network RTT is ~0.5ms (will vary by switch vendor) which is ~500,000ns that would come down to:

- Network memory read latency = 100ns + 500,000ns + [2 x OS / hypervisor overhead]

If we theoretically assume a very fast network with a 10,000ns RTT:

- Network memory read latency = 100ns + 10,000ns + [2 x OS / hypervisor overhead]

What that means is even with a theoretically fast network, there is a 10,000% overhead when compared to a non-network memory access. With a slow network this can be upwards of a 500,000% latency overhead.

In order to alleviate this overhead, server side caching technologies are introduced.

User vs. Kernel Space

One frequently debated topic is the argument between doing things in kernel vs. in user-space. Here I'll explain what each is and their respective pros/cons.

Any operating system (OS) has two core areas of execution:

- Kernel space
    - The most priviliged part of the OS
    - Handles scheduling, memory management, etc.
    - Contains the physical device drivers and handles hardware interaction

- User space
  - “Everything else”
  - This is where most applications and processes live
  - Protected memory and execution

These two spaces work in conjunction for the OS to operate. Now before moving on let’s define a few key items:

- System call
  - A.k.a. kernel call, a request made via interrupt (more here later) from an active process that something be done by the kernel

- Context switch
  - Shifting the execution from the process to the kernel and vice-versa

For example, take the following use-case of a simple app writing some data to disk. In this the following would take place:

1. App wants to write data to disk
2. Invokes a system call
3. Context switch to kernel
4. Kernel copies data
5. Executes write to disk via driver

The following shows a sample of these interactions:

User and Kernel Space Interaction

Is one better than the other? In reality there are pros and cons for each:

- User space
  - Very flexible
  - Isolated failure domains (process)
  - *Can be* inefficient
    - Context switches cost time(~1,000ns)

- Kernel space
  - Very rigid
  - Large failure domain
  - *Can be* efficient
    - Reduces context switches

Polling vs. Interrupts

Another core component is how the interaction between the two is handled. There are two key types of interaction:

- Polling
  - Constantly "poll" e.g. consistently ask for something
  - Examples: Mouse, monitor refresh rate, etc.
  - Requires constant CPU, but much lower latency
  - Eliminates expense of kernel interrupt handler
    - Removes context switch

- Interrupt
  - "Excuse me, I need foo"
  - Example: Raising hand to ask for something
  - Can be more "CPU efficient", but not necessarily
  - Typically much higher latency

The Move to User Space / Polling

As devices have become far faster (e.g. NVMe, Intel Optane, pMEM), the kernel and device interaction has become a bottleneck. To eliminate these bottlenecks, a lot of vendors are moving things **out of the kernel** to user space with polling and seeing much better results.

A great example of this are the Intel Storage Performance Development Kit (SPDK) and Data Plane Development Kit (DPDK). These projects are geared at maximizing the performance and reducing latency as much as possible, and have shown great success.

This shift is composed of two core changes:

1. Moving device drivers to user space (instead of kernel)
2. Using polling (instead of interrupts)

This enables far superior performance when compared to the kernel based predecessors, as it eliminates:

- Expensive system calls and the interrupt handler
- Data copies
- Context switches

The following shows the device interaction using user space drivers:

User Space and Polling Interaction

In fact, a piece of software Nutanix had developed for their AHV product (vhost-user-scsi), is actually being used by Intel for their SPDK project.

Book of Web-Scale

web·scale - /web ' skāl/ - noun - computing architecture

a new architectural approach to infrastructure and computing.

This section will present some of the core concepts behind “Web-scale” infrastructure and why we leverage them. Before we get started, it should be stated that the Web-scale doesn’t mean you need to be “web-scale” (e.g. Google, Facebook, or Microsoft). These constructs are applicable and beneficial at any scale (3-nodes or thousands of nodes).

Historical challenges included:

- Complexity, complexity, complexity
- Desire for incremental based growth
- The need to be agile

There are a few key constructs used when talking about “Web-scale” infrastructure:

- Hyper-convergence
- Software defined intelligence
- Distributed autonomous systems
- Incremental and linear scale out

Other related items:

- API-based automation and rich analytics
- Security as a core tenant
- Self-healing

The following sections will provide a technical perspective on what they actually mean.

Hyper-Convergence

There are differing opinions on what hyper-convergence actually is. It also varies based on the scope of components (e.g. virtualization, networking, etc.). However, the core concept comes down to the following: natively combining two or more components into a single unit. ‘Natively’ is the key word here. In order to be the most effective, the components must be natively integrated and not just bundled together. In the case of Nutanix, we natively converge compute + storage to form a single node used in our appliance. For others, this might be converging storage with the network, etc.

What it really means:

- Natively integrating two or more components into a single unit which can be easily scaled

Benefits include:

- Single unit to scale
- Localized I/O
- Eliminates traditional compute / storage silos by converging them

Software-Defined Intelligence

Software-defined intelligence is taking the core logic from normally proprietary or specialized hardware (e.g. ASIC / FPGA) and doing it in software on commodity hardware. For Nutanix, we take the traditional storage logic (e.g. RAID, deduplication, compression, etc.) and put that into software that runs in each of the Nutanix Controller VMs (CVM) on standard hardware.

What it really means:

- Pulling key logic from hardware and doing it in software on commodity hardware

Benefits include:

- Rapid release cycles
- Elimination of proprietary hardware reliance
- Utilization of commodity hardware for better economics
- Lifespan investment protection

To elaborate on the last point: old hardware can run the latest and greatest software. This means that a piece of hardware years into its depreciation cycle can run the latest shipping software and be feature parity with new deployments shipping from the factory.

Distributed Autonomous Systems

Distributed autonomous systems involve moving away from the traditional concept of having a single unit responsible for doing something and distributing that role among all nodes within the cluster. You can think of this as creating a purely distributed system. Traditionally, vendors have assumed that hardware will be reliable, which, in most cases can be true. However, core to distributed systems is the idea that hardware will eventually fail and handling that fault in an elegant and non-disruptive way is key.

These distributed systems are designed to accommodate and remediate failure, to form something that is self-healing and autonomous. In the event of a component failure, the system will transparently handle and remediate the failure, continuing to operate as expected. Alerting will make the user aware, but rather than being a critical time-sensitive item, any remediation (e.g. replace a failed node) can be done on the admin's schedule. Another way to put it is fail in-place (rebuild without replace) For items where a "leader" is needed, an election process is utilized. In the event this leader fails a new leader is elected. To distribute the processing of tasks MapReduce concepts are leveraged.

What it really means:

- Distributing roles and responsibilities to all nodes within the system
- Utilizing concepts like MapReduce to perform distributed processing of tasks
- Using an election process in the case where a "leader" is needed

Benefits include:

- Eliminates any single points of failure (SPOF)
- Distributes workload to eliminate any bottlenecks

Incremental and linear scale out

Incremental and linear scale out relates to the ability to start with a certain set of resources and as needed scale them out while linearly increasing the performance of the system. All of the constructs mentioned above are critical enablers in making this a reality. For example, traditionally you'd have 3-layers of components for running virtual workloads: servers, storage, and network – all of which are scaled independently. As an example, when you scale out the number of servers you're not scaling out your storage performance. With a hyper-converged platform like Nutanix, when you scale out with new node(s) you're scaling out:

- The number of hypervisor / compute nodes
- The number of storage controllers
- The compute and storage performance / capacity
- The number of nodes participating in cluster wide operations

What it really means:

- The ability to incrementally scale storage / compute with linear increases to performance / ability

Benefits include:

- The ability to start small and scale
- Uniform and consistent performance at any scale

Making Sense of It All

In summary:

1. Inefficient compute utilization led to the move to virtualization
2. Features including vMotion, HA, and DRS led to the requirement of centralized storage
3. VM sprawl led to increased load and contention on storage
4. SSDs came in to alleviate the issues but changed the bottleneck to the network / controllers
5. Cache / memory accesses over the network face large overheads, minimizing their benefits
6. Array configuration complexity still remains the same
7. Server side caches were introduced to alleviate the load on the array / impact of the network, however introduces another component to the solution
8. Locality helps alleviate the bottlenecks / overheads traditionally faced when going over the network
9. Shifts the focus from infrastructure to ease of management and simplifying the stack
10. The birth of the Web-Scale world!

About the Nutanix Bible

» Download this section as PDF (opens in a new tab/window)

The Nutanix Bible would not be possible without Dheeraj Pandey (founder of Nutanix and former CEO) and Steven Poitras (former Chief Architect).

A quote from Dheeraj on why and how the Nutanix Bible was born:

> First and foremost, let me address the name of the book, which to some would seem not fully inclusive vis-à-vis their own faiths, or to others who are agnostic or atheist. There is a Merriam Webster meaning of the word “bible” that is not literally about scriptures: “a publication that is preeminent especially in authoritativeness or wide readership”. And that is how you should interpret its roots. It started being written by one of the most humble yet knowledgeable employees at Nutanix, Steven Poitras, our first Solution Architect who continues to be authoritative on the subject without wielding his “early employee” primogeniture. Knowledge to him was not power – the act of sharing that knowledge is what makes him eminently powerful in this company. Steve epitomizes culture in this company – by helping everyone else out with his authority on the subject, by helping them automate their chores in Power Shell or Python, by building insightful reference architectures (that are beautifully balanced in both content and form), by being a real-time buddy to anyone needing help on Yammer or Twitter, by being transparent with engineers on the need to self-reflect and self-improve, and by being ambitious.
>
> When he came forward to write a blog, his big dream was to lead with transparency, and to build advocates in the field who would be empowered to make design trade-offs based on this transparency. It is rare for companies to open up on design and architecture as much as Steve has with his blog. Most open source companies – who at the surface might seem transparent because their code is open source – never talk in-depth about design, and “how it works” under the hood. When our competitors know about our product or design weaknesses, it makes us stronger – because there is very little to hide, and everything to gain when something gets critiqued under a crosshair. A public admonition of a feature trade-off or a design decision drives the entire company on Yammer in quick time, and before long, we’ve a conclusion on whether it is a genuine weakness or a true strength that someone is fear-mongering on. Nutanix Bible, in essence, protects us from drinking our own kool aid. That is the power of an honest discourse with our customers and partners.
>
> This ever-improving artifact, beyond being authoritative, is also enjoying wide readership across the world. Architects, managers, and CIOs alike, have stopped me in conference hallways to talk about how refreshingly lucid the writing style is, with some painfully detailed illustrations, visio diagrams, and pictorials. Steve has taken time to tell the web-scale story, without taking shortcuts. Democratizing our distributed architecture was not going to be easy in a world where most IT practitioners have been buried in dealing with the “urgent”. The Bible bridges the gap between IT and DevOps, because it attempts to explain computer science and software engineering trade-

> offs in very simple terms. We hope that in the coming 3-5 years, IT will speak a language that helps them get closer to the DevOps' web-scale jargon.
>
> – Dheeraj Pandey, Former CEO of Nutanix

Have feedback? Find a typo? Send feedback to biblefeedback at nutanix dot com

To learn more about Nutanix, check it out for yourself by taking a Nutanix Test Drive!

Thank you for reading The Nutanix Bible! Stay tuned for many more upcoming updates and enjoy the Nutanix platform!

This Classic Edition of The Nutanix Bible is an homage to the OG, Steve Poitras who started the Nutanix Bible ten years ago!

The hypervisors are supported by Nutanix AOS

- Nutanix AHV
- VMware vSphere ESXi
- Microsoft Hyper-v