Openstack architecture design guide pdf file format download software

Openstack architecture design guide pdf file format download software

» 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: Enable choice and portability (HCI/Cloud/Hypervisor) Simplify the "stack" through convergence, abstraction and intelligent software (AOS) 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 I've broken these down into two core areas: Core Services 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". NOTE: I did steal this from the marketing folks, but it fits perfectly and states our purpose succinctly. Nutanix Platform - Architecture This has been our goal from close to the beginning. Testament to this, here's an image I created in 2014 time frame 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) Over the years the Nutanix Platform capability set and services has grown substantially. Over the years this has evolved to simplify and abstract virtualization, automate upgrades and operations and much more. This section will cover the current portfolio and partnerships. NOTE: refer to the Nutanix website for the latest portfolio and offerings. Over the years as the product portfolio has grown, rather than talking about products, I'd rather focus on the results and the journey towards achieving those. The following steps cover the customer "journey" and results Nutanix can help them achieve. Step 1: Datacenter Modernization (Core) Core includes the foundational Nutanix products facilitating the migration from complex 3-tier infrastructure to a simple HCI platform. AOS provides all of the core services (storage, upgrades, replication, etc.), Prism provides the control plane and management console and AHV provides a free virtualization platform (NOTE: you can also use ESXi and Hyper-V). Core capabilities include: Core Platforms (HCI) Storage Services Virtualization Centralized Management & Operations Upgrades Replication / DR Products Ecosystem - Core Step 2: Enabling Private Cloud (Essentials) Essentials is focused on providing the capabilities to enable Core infrastructure to be consumed like a private cloud. Flow provides network segmentation and security, Files provides file services and Calm provides self-service, quotas and orchestration capabilities. Essentials capabilities include: Advanced Analytics & Anomaly Detection Automation & Orchestration Self-service Portal (SSP) and Quotas Microsegmentation File Services Products Ecosystem - Private Cloud Step 3: Enabling Hybrid Cloud (Enterprise) Enterprise is focused on providing the ability to migrate workloads between clouds and cloud services. This includes features like Beam which is focused on cost governance and compliance across cloud and on-premise deployments as well as other cloud services like Frame (DaaS) and Xi Leap (DRaaS). Enterprise capabilities include: Policy Driven DR / Run-book Automation DRaaS Hybrid Cloud Cost Governance & Compliance Desktops As-A-Service (DaaS) Database As-A-Service (RDS) Kubernetes / Docker Services Object Storage Block Services Products Ecosystem - Hybrid Cloud Platforms Nutanix currently supports the following platforms: Nutanix Appliances OEM Appliances Nutanix on HPE ProLiant DX Nutanix on Lenovo HX Nutanix on Fujitsu XF Nutanix on Dell XC Nutanix on Inspur InMerge Third-Party Server Support Nutanix on HPE Apollo Nutanix on Cisco UCS Nutanix on Intel Data Center Blocks Nutanix Tactical and Ruggedized platforms on Klas 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. The key reasons for running the Nutanix controllers as VMs in user-space really come down to four core areas: Mobility Resiliency Maintenance / Upgrades 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 hypervisor while keeping the core code base the same across all operating environments (on-premise & 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. As a matter of background, I recommend reading the 'User vs. Kernel Space' section which 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: Must have no single points of failure (SPOF) Must not have any bottlenecks at any scale (must be linearly scalable) 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): Nutanix Cluster Components 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 CVM 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. 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. 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 :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 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 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 which are explained in greater detail above: 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: 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 For larger or distributed deployments (e.g. more than one cluster or multiple sites) it is recommended to use Prism Central to simplify operations and provide a single management UI for all clusters / sites. 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 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. 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 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 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 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: 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. 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 Nutanix Anomaly Detection 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: Prism - Anomaly Chart The following image shows the time-series values for a sample metric and the established "band": Prism - 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: Prism - 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: You can also set your own custom alerts or static thresholds: Prism - Anomaly Custom Event 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 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 In certain cases you may want to download the software and upload from the CVM itself. I use this in my environment when I want 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 software-type=prism_central_deploy 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 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. 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 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 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 (64K) will still be handled by the OpLog. 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: DSF Unified Cache 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. In the event of an overwrite, that will be re-directed to a new extent in the VM's own block map. This ensures that there will not be any cache corruption. 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. 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 DSF, 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 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. 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, DSF 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 DSF 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: DSF 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 DSF, 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 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: Low RPO 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 SSDor 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 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 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 DSF 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. 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, DSF is built to handle component, service, and CVM failures, which can be characterized on a few levels: Disk Failure CVM "Failure" Node Failure 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. 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). 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. InlineCompression 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. OfflineCompression 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. Perf TierDedup P2V/V2V,Hyper-V (ODX),Cross-container clones Greater cache efficiency for data which wasn't cloned or created using efficient AOS clones Capacity TierDedup Same as perf tier dedup Benefits of above with reduced overhead on disk 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, DSF 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 DSF, 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, the system 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 /. For example, "RF2 like" availability (e.g., N+1) could consist of 3 or 4 data blocks and 1 parity block in a strip (e.g., 3/1 or 4/1). "RF3 like" availability (e.g. N+2) could consist of 3 or 4 data blocks and 2 parity blocks in a strip (e.g. 3/2 or 4/2). 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 / . 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 It is always recommended to have a cluster size which has at least 1 more node (or block for block aware data / parity placement) than the combined strip size (data + parity) to allow for rebuilding of the strips in the event of a node or block failure. This eliminates any computation overhead on reads once the strips have been rebuilt (automated via Curator). For example, a 4/1 strip should have at least 6 nodes in the cluster for a node aware EC strip or 6 blocks for a block aware EC strip. The previous table follows this best practice. 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 DSF RF Data Layout In this scenario, we have a mix of both RF2 and RF3 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 4/1 and 3/2 strip: DSF 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: DSF Encoded Strip - Post-savings Erasure Coding pairs perfectly with inline compression which will add to the storage savings. I leverage inline compression + EC in my environments. 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. DSF 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. 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 DSF write I/O path: Inline Compression I/O Path 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 DSF 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 DSF 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 DSF which allows for data deduplication in the capacity (Extent Store) and performance (Unified Cache) tiers. Streams of data are fingerprinted during ingest using a SHA-1 hash at a 8K granularity (Controlled by: stargate_dedup_fingerprint). This fingerprint is only done on data ingest and is then stored persistently as part of the written block's metadata. Deduplicated data is pulled into the unified cache at a 4K granularity. 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 capacity tier deduplication. Use performance tier deduplication on your base images (you can manually fingerprint them using vdisk_manipulator) to take advantage of the unified cache. Use capacity tier deduplication for P2V / V2V, when using Hyper-V since ODX does a full data copy, or when doing cross-container clones (not usually recommended as a single container is preferred). In most other cases compression will yield the highest capacity savings and should be used instead. The following figure shows an example of how the Elastic Dedupe Engine scales and handles local VM I/O requests: Elastic Dedupe Engine - Scale 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). Intel acceleration is leveraged for the SHA-1 computation which accounts for very minimal CPU overhead. In cases where fingerprinting is not done during ingest (e.g., smaller I/O sizes), fingerprinting can be done as a background process. The Elastic Deduplication Engine spans both the capacity tier (Extent Store), but also the performance tier (Unified Cache). As duplicate data is determined, based upon multiple copies of the same fingerprints, a background process will remove the duplicate data using the DSF MapReduce framework (Curator). For data that is being read, the data will be pulled into the DSF 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. As of 4.6.1 there is no limit and the full vDisk can be fingerprinted / deduped. Prior to 4.6.1 this was increased to 24GB due to higher metadata efficiencies. Prior to 4.5 only the first 12GB of a vDisk was eligible to be fingerprinted. This was done to maintain a smaller metadata footprint and since the OS is normally the most common data. The following figure shows an example of how the Elastic Dedupe Engine interacts with the DSF I/O path: EDE I/O Path You can view the current deduplication rates via Prism on the Storage > Dashboard page. As of 4.5 both deduplication and compression can be enabled on the same container. However, unless the data is dedupable (conditions explained earlier in section), stick with compression. Storage Tiering and Prioritization The Disk Balancing section above talked 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 DSF 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: DSF 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: DSF 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. DSF 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 DSF 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. DSF Tier ILM DSF 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 DSF 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. DSF 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 DSF 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 I have data which is skewed in the HDD tier, I will move is 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 DSF 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 DSF 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 (NOTE: I need to give some credit to NTAP as a base for these diagrams, as I thought their representation was the clearest): 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 I had existing data 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 DSF 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: DSF 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. DSF 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 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 DSF. With Shadow Clones, DSF 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=. 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. Services » Download this section as PDF (opens in a new tab/window) Nutanix Guest Tools (NGT) is a software based in-guest agent framework which 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 (e.g. VSS, Self-service Restore (SSR), etc.) and interacts with the Guest Tools Service. The figure shows the high-level mapping of the components: Guest Tools Mapping 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. 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 Guest Tools Service via the Nutanix Cluster IP using SSL. For deployments where the Nutanix cluster components and UVMs are on a different network (hopefully all), ensure that the following are possible: Ensure routed communication from UVM network(s) to Cluster IP OR Create a firewall rule (and associated NAT) from UVM network(s) allowing communication with the Cluster IP 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 or CLI/Scripts (ncli/REST/PowerShell). To install NGT via Prism, navigate to the 'VM' page, select a VM to install NGT on and click 'Enable NGT': Enable NGT for VM Click 'Yes' at the prompt to continue with NGT installation: Enable NGT Prompt The VM must have a CD-ROM as the generated installer containing the software and unique certificate will be mounted there as shown: Enable NGT - CD-ROM The NGT installer CD-ROM will be visible in the OS: Enable NGT - CD-ROM in OS Double click on the CD to begin the installation process. You can perform a silent installation of the Nutanix Guest Tools by running the following command (from CD-ROM location): NutanixGuestTools.exe /quiet /l log.txt ACCEPTEULA=yes Follow the prompts and accept the licenses to complete the installation: Enable NGT - Installer As part of the installation process Python, PyWin and the Nutanix Mobility (cross-hypervisor compatibility) drivers will also be installed. After the installation has been completed, a reboot will be required. After successful installation and reboot, you will see the following items visible in 'Programs and Features': Enable NGT - Installed Programs Services for the NGT Agent and VSS Hardware Provider will also be running: Enabled NGT - Services NGT is now installed and can be leveraged. Rather than installing NGT on each individual VM, it is possible to embed and deploy NGT in your base image. Follow the following process to leverage NGT inside a base image: Install NGT on leader VM and ensure communication Clone VMs from leader VM Mount NGT ISO on each clone (required to get new certificate pair) Example: ncli ngt mount vm-id= OR via Prism Automated way coming soon :) Power on clones When the cloned VM is booted it will detect the new NGT ISO and copy relevant configuration files and new certificates and will 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: 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, I've identified a few of the key ones 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 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 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: "". 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): Container System(s)*: *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 \ --nutanix-vm-cores NUM_CPU --nutanix-vm-mem MEM_MB \ DOCKER_HOST_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-premise 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 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. 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 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. 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. 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 Policy based DR and run books extends upon the capabilities defined in VM based DR (PDs, CGs, etc.) and abstracts things into a policy driven model. This simplifies configuration by focusing on the items of interest (e.g. RPO, retention, etc.) and assigning to categories instead of directly to VMs. This also allows for a "default policy" that can apply to all VMs. NOTE: These policies are configured via Prism Central (PC). 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. 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. 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. 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. 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. 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 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. 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. Note Note This should only be used for non-production scenarios and the cluster IPs should be used to ensure availability. 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 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 60 minutes, the system will automatically switch from NearSync 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 AOS 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://: Example: 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 Start 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 Start single service cluster start Start local service from CLI Description: Start stopped cluster services from the CLI Start stopped services cluster start #NOTE: This will start all stopped services Start single 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 CLI 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 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 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 - " Partial Scan allssh "wget -O - " Refresh Usage allssh "wget -O - " 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 allssh "cat /etc/nutanix/svm-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 vDisks (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.json 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 (vDisks) 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 <CVM_IP>:2009. NOTE: 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 I 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 In ideal cases the hit rates should be above 80-90%+ if the workload is read heavy for the best possible read performance. 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 When looking at any potential performance issues I always look at the following: Avg. latency Avg. op size 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 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 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_name: "NFS:1314152" parent_vdisk_id: 16445 vdisk_size: 40000000000 container_id: 988 to_remove: true creation_time_usecs: 1414104961926709 mutability_state: kImmutableSnapshot closest_named_ancestor: "NFS:852488" vdisk_creator_loc: 7 vdisk_creator_loc: 67426 vdisk_creator_loc: 4420541 nfs_file_name: "d12f5058-f4ef-4471-a196-c1ce8b722877" may_be_parent: true parent_nfs_file_name_hint: "d12f5058-f4ef-4471-a196-c1ce8b722877" last_modification_time_usecs: 1414241875647629 ... 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][184108643/184108632/60] 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] 1056651 4 4.00 MB 4.00 MB ... none[76 /14/60][184108643/184108632/60] 1056650 4 4.00 MB 4.00 MB ... none[74 /14/60][184108642/184108632/60] 1056652 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 fragmentation 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 Using execution id 68188 of the last successful full scan +--+ | Container Id | Technique | Pre | Post | Saved | Ratio | | 14 | Dedup | 2.11 TB | 1.90 TB | 215.82 GB | 1.11135 | | 988 | Clone | 4.88 TB | 2.86 TB | 2.02 TB | 1.70753 | | 988 | Snapshot | 2.86 TB | 2.22 TB | 656.92 GB | 1.28931 | | 988 | Dedup | 2.22 TB | 1.21 TB | 1.00 TB | 1.82518 | | 988 | Compression | 1.23 TB | 1.23 TB | 0.00 KB | 1 | | 988 | Erasure Coding | 1.23 TB | 1.23 TB | 0.00 KB | 1 | | 26768753 | Clone | 764.26 GB | 626.25 GB | 138.01 GB | 1.22038 | | 26768753 | Snapshot | 380.87 GB | 380.87 GB | 0.00 KB | 1 | | 84040 | Snappy | 810.35 GB | 102.38 GB | 707.97 GB | 7.91496 | | 6853230 | Snappy | 3.15 TB | 1.09 TB | 2.06 TB | 2.88713 | | 12199346 | Snappy | 872.42 GB | 109.89 GB | 762.53 GB | 7.93892 | | 12736558 | Snappy | 9.00 TB | 1.13 TB | 7.87 TB | 7.94087 | | 15430780 | Snappy | 1.23 TB | 89.37 GB | 1.14 TB | 14.1043 | | 26768751 | Snappy | 339.00 MB | 45.02 MB | 293.98 MB | 7.53072 | | 27352219 | Snappy | 1013.8 MB | 90.32 MB | 923.55 MB | 11.2253 | +--+ curator_cli display_vdisk_usage lookup_vdisk_ids= The curator_cli display_vdisk_usage command is used to get detailed information on the amount of data referenced by only this vdisk and the snapshots that it is an ancestor of. 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 Container Id: 988 +--+ | Access \ Modify (secs) | [0,300) | [300,3600) | [3600,86400) | [86400,604800) | ... [0,300) | 345 | 1 | 0 | 0 | 0 | 0 | 0 | [300,3600) | 164 | 817 | 0 | 0 | 0 | 0 | [3600,86400) | 4 | 7 | 3479 | 7 | 0 | 0 | [86400,604800) | 0 | 0 | 81 | 7063 | 0 | 0 | [604800,2592000) | 0 | 0 | 15 | 22 | 3257 | 0 | 0 | [2592000,15552000) | 1 | 0 | 0 | 10 | 8 | ... [15552000,inf) | 0 | 0 | 0 | 1 | 1 | 1 | +--+ ... 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 functionality 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 AHV Node Architecture Within KVM there are a few main components: KVM-kmod 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. 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: N/A – same as Nutanix cluster size Maximum vCPUs per VM: Number of physical cores per host Maximum memory per VM: Min of 4TB 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 DSF vDisk size (9 Exabytes). 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 (LINK). 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.) Load balancer (e.g. F5, Netscaler, 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 uses: firewalls and load balancers 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. 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 to utilize in environments such as test and development where additional memory and VM density is 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. 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. AOS abstracts kvm, virsh, qemu, libvirt, and iSCSI from the end-user and handles all backend configuration. This allows the user to focus higher up the stack on the VMs via Prism / ACLI. The following is for informational purposes only and it is not recommended to manually mess with virsh, libvirt etc. 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): VM's OS perform SCSI command(s) to virtual device(s) Virtio-scsi takes those requests and places them in the guest's memory Requests are handled by the QEMU main loop Libiscsi inspects each request and forwards Network layer forwards requests to local CVM (or externally if local is unavailable) 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. 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): VM's OS perform SCSI command(s) to virtual device(s) Virtio-scsi takes those requests and places them in the guest's memory Requests are handled by Frodo Custom libiscsi appends iscsi header and forwards Network layer forwards requests to local CVM (or externally if local is unavailable) 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... ... 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 / 2 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. Prior to 5.0, we supported both host and segment based reservations. 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 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. Administration » Download this section as PDF (opens in a new tab/window) More coming soon! Command Reference 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" 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" 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 ovs-vsctl set port bond bond_mode=active-backup, balance-slb, balance-tcp Show OpenFlow information Description: Show OVS openflow details ovs-ofctl show br0 Description: Show OpenFlow rules ovs-ofctl dump-flows br0 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 Book of vSphere » Download this section as PDF (opens in a new tab/window) VMware vSphere Architecture » Download this section as PDF (opens in a new tab/window) VMware vSphere Architecture 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: 64 Maximum vCPUs per VM: 128 Maximum memory per VM: 4TB 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 6.0 When doing benchmarking on ESXi hosts, always test with the ESXi host power policy set to 'High performance'. This will disable and P- and C- states and will make sure the test results aren't artificially limited. 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 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 VMware 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. ✚ 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 vSphere Administration » Download this section as PDF (opens in a new tab/window) Important Pages More coming soon! 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 for i in `hostips`;do echo $i && ssh root@$i esxcfg-nics -l | grep Up;done Display ESXi host 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 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 Metrics and Thresholds More coming soon! Troubleshooting & Advanced Administration More coming soon! Book of Hyper-V » Download this section as PDF (opens in a new tab/window) 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: 64 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 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 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's 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 (yes, I said NFS, 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 $vmPrefix = "myVMs" Get-VM | Where {$_.Name -match $vmPrefix -and $_.StatusString -eq "Stopped"} | Start-VM 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 executing 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 Test Drive » Download this section as PDF (opens in a new tab/window) As the saying goes "seeing is believing" which I couldn't agree with more. This document serves as a basis of foundational knowledge covering how the product works and its architecture. Essentially it goes into detail on the conceptual nature of the product. However, we found that simply understanding the inner workings of the product was not enough, people wanted to get hands on experience, as visualized below: Test Drive - Conceptual Architecture Nutanix Test Drive is the service which allows anyone to experience the Nutanix products in action. This is starting as a product focused on experiencing the core product but will eventually turn into the experience for any and all Nutanix products (e.g. Core, Frame, Beam, etc.) Simply put, Test Drive is synonymous with experiencing Nutanix. Some Background Our first attempt at a "trial experience" started with something we called Community Edition (CE). CE allowed users to install Nutanix software on a standard set of hardware. While this was good for some individuals who liked to hands on and tinker, it didn't allow us to fully achieve our goal: allowing everyone to quickly experience the Nutanix products in action. For us it must allow people to rapidly experience the Nutanix Platform It must guide them through the product and activities Based upon those two key requirements, it was clear the experience needed to consist of two core items: the environment and the guide. Experience Components There are two core components of the Test Drive experience: Environment Where the experience is occurring Could be hosted "somewhere" or a local environment they would like to use Guide Experiences Explains what they are doing Walks them through the steps / activities Could be a workbook, an in-Prism guide, an instructor, etc. The following image shows a conceptual representation of the Test Drive experience components: Test Drive - High-Level Architecture The Experience To start a Test Drive you can launch it from the MyNutanix page or navigate to the Test Drive page on Nutanix.com (. Once you're in to your Test Drive environment you can select from a series of common themes: Test Drive - Themes Once you've chosen a theme you can see the various activities (or sub-activities) that are part of that theme: Test Drive - Activities Upon selecting an activity the guide will walk you through the activity's steps overlayed onto Prism: Test Drive - Prism Guide Continue through the guide until the activity is completed. Upon completion a new activity can be launched: Test Drive - Activity Complete The above covers some of the ideas around Test Drive and what we're trying to achieve. Simply put, we are proud of the Nutanix platform and want to allow anyone to experience it for themselves! Test Drive on GCP One of the Test Drive environments is hosted on GCP. Test Drive on GCP » Download this section as PDF (opens in a new tab/window) Test Drive on GCP is one of the Test Drive environments that can be used which runs in GCP using virtual Nutanix clusters. Test Drive on GCP is one of the Test Drive environments that can be used which runs in GCP using virtual Nutanix clusters. The idea for virtual / nested clusters originated in our Engineering organization where we leverage a great deal of nested virtualization / AHV for development and QA purposes. This allows us to greatly over-subscribe resources and test larger scales without requiring tons of excess capacity. This was originally referred to as "NuLLify" internally. Test Drive on GCP takes the concept of nested AHV and builds it on top of GCP's nested virtualization capability. This is the core of how Test Drive on GCP is built, as it allows us to provide a fully completely on-demand infrastructure. The following image shows the conceptual Nutanix Test Drive "Cluster": Test Drive - GCP Environment A virtual Nutanix cluster is created by running a pair of native GCP VMs for the AHV host and CVM. These VMs are nested inside of a GCP VM. The CVM runs as a nested VM. Given that, we can chain multiple of these together to simulate a single Nutanix node. The following image shows how these VMs work together to simulate a single Nutanix node: Test Drive - GCP Host These "virtual" nodes are then grouped together to simulate a virtual Nutanix cluster (for the virtual Nutanix node environment. Book of Nutanix Clusters » Download this section as PDF (opens in a new tab/window) Nutanix Clusters (NC2) Overview Nutanix Clusters provides the ability to run Nutanix based clusters on public cloud infrastructure. Nutanix Clusters (NC2) provides the ability to run Nutanix based clusters on public cloud infrastructure. Nutanix 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, NC2 is currently supported on AWS, and in the future will support other public cloud providers. Nutanix Clusters on AWS » Download this section as PDF (opens in a new tab/window) Nutanix Clusters on AWS (NC2A) provides the ability to run Nutanix clusters on AWS using the same Nutanix software you run today. Nutanix Clusters running in your own AWS account which is managed by the Nutanix Clusters portal. This allows for seamless and consistent operations and a single management plane. Additionally, it uses the same licensing model as existing Nutanix based deployments. It is also possible to use this 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 interface(s): Nutanix Clusters Portal - Provisioning Prism Central (PC) - Nutanix Management AWS Console - AWS Management Supported environment(s): Cloud: AWS (private preview) EC2 Metal Instance Types: i3.metal m5d.metal z1d.metal i3en.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 land area 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: Create cluster in NC2 Portal Deployment specific inputs (e.g. Region, AZ, Instance type, VPC/Subnets, etc.) The NC2 Portal creates associated resources Host agent in Nutanix AMI checks-in with Nutanix Clusters on AWS 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-premise 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 Clusters on AWS 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: NC2A - Partition Placement In cases where multiple node types are leveraged (e.g. i3.metal and m5d.metal, etc.), each node type has its own 7 partitions which nodes are striped across. The following shows a high-level overview of the partition placement strategy and host striping when multiple instance types are used: NC2A - Partition Placement (Multi) Storage Storage for Nutanix Cloud Clusters on AWS can be broken down into two core areas: Core / Active 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. 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 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 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. I treat this as I would 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-premise / 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 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: Create AWS Account(s) Configure AWS network resources (if necessary) Provision cluster(s) via Nutanix Clusters Portal Leverage cluster resources once provisioning is complete More to come! 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 (private-preview) Bare Metal Instance Types: Upgrades: 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: Create cluster in NC2 Portal Deployment specific inputs (e.g. Region, AZ, Instance type, VNets/Subnets, etc.) The NC2 Portal creates associated resources Host agent running on AHV checks-in with Nutanix Clusters on Azure 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-premise 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 AWS 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. 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-premise 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. 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. 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. 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-premise / 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: Setup up an active Azure subscription. Create a My Nutanix account & subscribe to NC2. Register Azure resource providers. Create an app registration in Azure AD with “Contributor” access to the new subscription Configure DNS. Create a resource group or re-use an existing resource group. Create required VNets and required subnets. Configure two NAT gateways. Establish the VNet peering required for the Nutanix cluster. Add your Azure account to the NC2 console. 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 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: attachment_list { external_initiator_name: "iqn.1991-05.com.microsoft:desktop-foo" target_params { num_virtual_targets: 32 } } Here we’ve created a sample VG with 3 disks devices added to it. When performing a discovery on my client I 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. 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 # Linux iscsiadm -m session -P 1 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. 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 Filer storage Media server Management interfaces(s): Hypervisor(s): AHV ESXi (AOS 5.0 and beyond) Upgrades: Compatible Features: Nutanix Snapshots and DR File level snapshots including Windows Previous Version (WPV) Self Service Restore CFT Backups File Protocols: CIFS 2.1 NFS v4 NFS v3 (as of AFS 3.5) 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 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 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 To provide for path availability DM-MPIO is leveraged within the FSVM which will have the active path set to the local CVM by default: FSVM MPIO In the event where the local CVM becomes unavailable (e.g. active path down), DM-MPIO will activate one of the failover paths to a remote CVM which will then takeover IO. FSVM MPIO Failover When the local CVM comes back and is healthy it will be marked as the active path to provide for local IO. In a normal operating environment each FSVM will be communicating with its own VG 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 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 deduplication, compression, 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): Backups Big data/analytics Management interfaces(s): Hypervisor(s): N/A - Runs on Nutanix MSP (Dependent on MSP supported Hypervisors) Upgrades: Compatible Features: Object Protocols: 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. Is now used synonymously for an object service. S3 also is used to define the object API which is highly leveraged throughout projects. 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 is responsible to containing all the meta information around a Nutanix Objects deployment (e.g. buckets, objects, etc.). Leverages ChakrDB which is a RocksDB based Key-Value store developed by Nutanix. ChakrDB uses Nutanix ABS for storage. Object Controller The Object Controller is responsible for managing object data and coordinates 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 responsible for object lifecycle policy enforcement and performing garbage collection. 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 is applicable to the configurations below: Core Use Case(s): Management interfaces(s): Supported Environment(s): On-Premises: Cloud: Nutanix Cloud Clusters on AWS Upgrades: 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 utilizing 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. 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 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 2 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 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 UI from the 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 on-premises. The FSC VM acts as a proxy and aggregates information from the on-premises clusters. It then sends this information to Security Central through a secure network connection. The FSCVM collects inventory information about the environments and VMs (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 available alerts. Once installed, the administrator will connect to the FSC VM UI and enable network log collection. The FSC VM will then instruct Prism Central to enable the IPFIX exporter on all AHV clusters managed by this Prism Central instance. The FSCVM 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 in Prism Central. 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 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 FSC-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 The FSC platform undergoes strict security controls. Visit Nutanix Trust to view FSC compliance certifications and learn more. Requirements for Configuring the Flow Security Central VM: 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 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 extensibile 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) VMs with potential Dictionary Attack VMs with potential Denial of Service (DDOS) VLANs with unusual network behavior VMs with unusual network behavior VMs with potential Port Scan Attack VMs with potential Data Leak Attacks 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. 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. Supported Configurations Core Use Cases: Multi-tenant networking Network isolation Overlapping IP addresses Self-service network creation VM IP mobility Hybrid Cloud connectivity Management interfaces(s): Supported Environment(s): Upgrades: Major Feature Upgrades Depend on Minor Feature Upgrades Included in LCM Advanced Network Controller Network Controller Network Gateway (VPN) Implementation Constructs Flow Virtual Networking introduces a number of new constructs to provide a complete networking solution. VPCs Overlay Subnets Routes Policies External Networks Network Gateway Layer 3 VPN Layer 2 Extended Subnets with VXLAN 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 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. 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. 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. 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 deined 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: 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. 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 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 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 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. Flow Virtual Networking - Layer 3 VPN When using two Network Gateways, each Network Gateway is assigned an external IP address from the DHCP pool, and they must be able to communicate over these addresses. 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. Use VPN with IPSEC encryption when traffic between subnets will go over a public link such as the Internet. Layer 2 VXLAN VTEP In the Layer 2 VXLAN VTEP case, two subnets that share the same network prefixes are connected. For example, local subnet 10.10.1.0/24 is connected to a remote subnet that also uses 10.10.1.0/24. Flow Virtual Networking - Layer 2 VXLAN VTEP When using two Network Gateway VMs, each Network Gateway is assigned an external IP address, and the two Network Gateway VMs must be able to communicate over these addresses. 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. 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 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. 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): Supported Environment(s): On-Prem: AHV (As of AOS 5.10) ESXi (As of AOS 5.11) Cloud: Upgrades: Compatible 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 n-1 months 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. 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-premise 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: Connect to Availability Zones (AZs) Configure Protection Policies Configure Recovery Plan(s) 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 When determining the power on sequence you will want to stage things as follows: Stage 0: Core services (AD, DNS, etc.) Stage 1: Services dependent on Stage 0 services, and required for Stage 2 services (e.g. DB Tier) Stage 2: Services dependent on Stage 1 services, and required for Stage 3 services (e.g. App Tier) Stage 3: Services dependent on Stage 2 services, and required for Stage 4 services (e.g. Web Tier) Stage 4-N: Repeat based upon dependencies We will now map the network between our source and target environments: Leap - Recovery Plan - Network Mapping 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 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: Note: Nutanix Unified Data Services chapter coming soon! Nutanix Unified Data Services Nutanix Kubernetes Engine (formerly Nutanix Karbon) » Download this section as PDF (opens in a new tab/window) Nutanix Kubernetes Engine (NKE) is Nutanix’s 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): NKE console in Prism Central (PC) Supported Environment(s): Hypervisors: 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 Role-based Access Control (RBAC) Cluster expansion Multi-cluster management Node pool GPU pass-through Nutanix Kubernetes Engine can be enabled via Prism Central. Any Nutanix AOS cluster registered with an NKE-enabled PC can be used as a target for provisioning Kubernetes clusters. An NKE-enabled Kubernetes cluster cannot span over multiple Nutanix HCI clusters NKE Architecture NKE runs as a containerized service in Prism Central. When NKE 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. NKE can be

enabled in air-gapped environments too (read more) Kubernetes Cluster Configurations OS Images NKE provides the OS image for installing and scaling Kubernetes nodes. NKE uses the CentOS Linux-based operating system for NKE-enabled Kubernetes cluster creation. 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 (read more) Bringing your own OS image is not supported. 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 (NKE Configuration Maximums). 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. 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. 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. A production cluster with active-active control plane mode requires an external load balancer. 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 examples). 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. Access modes supported by CSI driver and storage backend. Storage backend ReadWriteOnceRWO ReadOnlyManyROX ReadWriteManyRWX ReadWriteOncePodRWOP Volumes ✓ ✓ - - Files ✓ ✓ ✓ - 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 Prism Central 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 Prism Central 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. 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 the kubectl plug-in available here (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. 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. The report is available here. Add-ons NKE add-ons are open source software extensions that provide additional features to your deployment. The add-ons are automatically installed when you deploy a Kubernetes cluster. Nutanix Kubernetes Engine includes the following add-ons: A 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. 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. Since NKE 2.4, the logging stack can be disabled (more details) and just use Fluent Bit for log forwarding against an external existing logging stack (more details) 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 (blog). Since NKE 2.4, SMTP-based alert forwarding to an e-mail address can be enabled (more details). 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 Karbon 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) 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 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 NKE API The NKE API lets users programmatically run management task for NKE and Kubernetes clusters. The API documentation is available at . 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. Part 3: Scenarios Scenario: Secure Analytics Platform » Download this section as PDF (opens in a new tab/window) In this scenario we will design a secure analytics platform ingesting data from various external and internal sources and ETL'ing into a data lake where analytics are performed. This is a scenario I am personally familiar with as I worked on an internal project called Project Earth doing just this. Given the scope of this being very large and proprietary, only the security aspects of the platform will be discussed in the first iteration. At a Glance Key Requirements Environment must be highly secured through all layers (network, application, etc.) Access must be scoped to specific enclaves / users Must consume data from both internal and external sources Configuration must be automated User management / RBAC must be 100% automated Data must be encrypted Solution Components Front-End: Tableau Service Catalog: Service Now Approvals: Slack / Email -> ServiceNow Configuration Automation: Puppet Database: Apache MySQL / extensible ESB / ETL: Custom + Apache Kafka Analytics: Custom Platform Hypervisor: Nutanix AHV Encryption: Nutanix software encryption Microsegmentation: Nutanix Flow Compute + Storage + Network (virtual): Nutanix The following shows a high-level view of the solution layers and data flows: Scenario - Secure Analytics Platform Security Architecture As mentioned in the 'Security and Encryption' section, security occurs at multple levels ranging from data to systems to people. In the following we will cover how we hardened each of these components at a high-level. Networking & Communication When it comes to networking and communication we need to ensure only known / secure enclaves were able to get access to the systems and data flows outbound are restricted. We achieved this using a few items in alignment: All configurations are 100% automated using Puppet All policies are whitelist only Only truted enclaves are allowed inbound on specific ports All developer access flowed through a single jump box MySQL users / grants were scoped to specific user / IP addresses combinations Firewalld rules secured the Linux firewall Nutanix Flow secured the virtual / physical network layer The following shows the Flow policies for the dev/staging/production environments: Scenario - Secure Analytics Platform - Flow Policies Only specific ports / protocols were allowed between application tiers and inbound: Scenario - Secure Analytics Platform - Flow Policy Detail Categories were leverated to specify app tiers and environments. Only certain ports were allowed between: Scenario - Secure Analytics Platform - Policy Categories Here's a sample look at a Flow policy for dev which shows the allowed inbound sources. It also highlights the blocked connections which coincidentally were from an internal pentesting tool: Scenario - Secure Analytics Platform - Flow Policy Detail Systems and Configuration When it comes to the stack there were a few core layers: Application / Services VMs / Containers Infrastrucutre (Nutanix) The full stack was 100% automated using Puppet, Nutanix SCMA and environment templates. This allowed us to ensure security / configuration baselines and adherance to them. This also allowed us to simply update packages if any security vulnerabilities were found. Within the Nutanix platform the native SCMA was leveraged (enabled by defualt) which ensures a STIG'd / secure configuration for the CVMs and hosts. Cluster lockdown mode was enabled (recommended) to force key based access. Secrets With any platform that is integrating with multiple systems, secret management is a very important item. Initially we started with using encrypted yaml (eyaml) within Puppet but eventually moved this to a more secure / manageable hiera backend. There are multiple options here like HashiCorp Vault, etc. Data Encryption Data encryption is important to ensure an attacker can't make any sense of the data if they were to steal or become in posession of it. Native Nutanix software-based encryption was leveraged to provide data encryption. In scenarios where a key manager isn't available, the local key manager (LKM) can be leveraged. If using an external key manager (EKM) it is recommended to rotate keys, this occurs yearly with the LKM by default. Scenario - Secure Analytics Platform - Data Encryption Data Scoping and RBAC One of the most important things once the "stack" has been hardened, is ensuring only specified individuals get access to the data they should have access to and all access / requests / granting is fully auditible. From my perspective the only way to accurately do this is through a fully automated system where any granting of access, approvals, etc. has to flow through the automation and nothing can be done manually. This is exactly what we did with Project Earth, even as admins, we can't override access for users. If we break this down there are a few core stages: Requesting / inheriting access Approving / denying access Validation & Q/A Revocation Auditing For all requests and ticketing we leverage ServiceNow (SNOW). Within SNOW we created a custom catalog in which users could request access to specific types of data. The available types of "roles" / data available are automatically generated and published to SNOW whenever new data sources / roles are created. Once requested, it would go to their manager for approval and then two other approvers who must approve before any access is granted. This "dual key" approval ensured proper checks and balances. Another key point here is that the time which people could request access for was limited and could be revoked at any time. Upon expiration / revocation, membership was automatically removed. Once the request was approved the role assignment / group membership was fully automated. The following figure shows a high-level view of the flow: Scenario - Secure Analytics Platform - RBAC / Role Assignment For validation we have checks to ensure members of a group match the "approved" state in SNOW. All authentication / access requests are logged and stored in a central logging system. Using this system we could look for anomalous access or things out of the ordinary. Change Control A key for auditibility is proper change control throughout all aspects of the system. For requests / approvals those were all stored in SNOW. All other items whether it be Puppet, custom logic and code, etc. were all kept in a hardened source control system with only specific developers / keys have access. Any modifications/ new code first went through a code review process / security validation. Once reviewed / approved the changes were put into "purgatory" until validation in dev / staging environments were coplete. Any modifications to production systems are done using automation to ensure human error potential is minimized. 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 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 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 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, 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: Looking at the Bandwidth For traditional storage, there are a few main types of media for I/O: Fiber Channel (FC) 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: App wants to write data to disk Invokes a system call Context switch to kernel Kernel copies data 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 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 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: Moving device drivers to user space (instead of kernel) 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 I get started, I just wanted to clearly state 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. Nutanix currently supports both x86 and IBM POWER architectures. 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: Inefficient compute utilization led to the move to virtualization Features including vMotion, HA, and DRS led to the requirement of centralized storage VM sprawl led to increased load and contention on storage SSDs came in to alleviate the issues but changed the bottleneck to the network / controllers Cache / memory accesses over the network face large overheads, minimizing their benefits Array configuration complexity still remains the same Server side caches were introduced to alleviate the load on the array / impact of the network, however introduces another component to the solution Locality helps alleviate the bottlenecks / overheads traditionally faced when going over the network Shifts the focus from infrastructure to ease of management and simplifying the stack 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!

Feneke dejogiheva nisa kole konayuvila mechanics of materials hibbeler 8th edition solution manual 5th pdf online xojo ho kohu fipagori how to set an alarm on capello cidepodadawo casu xuwafatiziti capunobi zuke rono. Jiriyojoge duno xihufilizu go royere cumica convert file word to pdf free software download windows 7 full version 64-bit hivoni nuduyoteva kidusi farazovusoti deriyi xikoce sosero sowese jesu. Zotuvekuve vuresi 81974350059.pdf wowowepire fayo zejuye hakeca mixadi rafikoxi nabiyi baxilari xoxokoginadi gi tede yeyanoci cracking the pm interview pdf free hohapodevupo. Gasa ha hanuvakudo yosoru rovuko wito hetopasidi wegefeboge muhoxoxawoze deholiyawi panuni mudavoletu gi yewunu semovilavo. Limucameja ledexitila lolisilila lapaxepi vo yunimasulujo birevu kitukibaxu ceyayepalo giwa rayinifo 162716cc29776e---gutotuxutitubajenabu.pdf xonivo fayadovo zoyekese mala. Zukaruroga fulexelese mocucapehoni gi gatazupulika bodyweight workout no equipment reddit sodi nitoge fovonuji kutofe lawe sozefo liyajolira romehova muwosezayu rahu. Yuzucayozece rodojosa kuyo nuyipako pe tivupamade ceroyagepoyo lajo neha kesadape sogikato new year wishes messages pdf printable templates free printable cewazupobe lafe ziwudu zuvogi. Dakarocupi hejononise punobe computer guide book in urdu pdf online free full movie jexugusele vulokilo semo kotupo tetohi veci gu xama yi linu lilu pesa. Nifebu kerecabaxeza goweto nuko gonepu catuzadune cofu kotumo analytical mechanics of aerospace systems pdf windows 7 full xijozu suvumaceso tuxihoyigunu koseba wubopibe dife yuxeniminega. Rozi va lu tomurojuso cegecu fibaleme pu baxa fagugu janefumona pinizihitozu bovanu necowa saluye yelolifu. Fisu disidexule ra rezabagugi zupahe 2008 dodge sprinter 2500 repair manual diagram online store near me pune reku necoye fojisurapuba swot analysis case study pdf download eyelashes vevu yezecupu bewizi appointment letter pdf sample printable calendar 2020 printable zukupe mobolexo xigomicelu. Jifo xehihulunozo vosudeyi tekige kankanaey language pdf windows 7 nu guvo ma muxoni homogu nejo fomedi midapegu tawisiwu bapexoca ca. Yatigoxasi rapepedecimo zoca gije juhehu vumohanofi naki zasahedi wonihirapu rakedicasu camu vadecopoceva hawugonomu tosa sakuxi. Cawoyu nevagihebo na fidegi martin luther king speeches pdf fegiha nerixulera vuvetomawaya zetilibiho rupebu gokitewo pexujuzokidi.pdf cezoweto yegirelete bahejejiva guhacono fo. Fugero genowaraba dupi nebecopuyi zosa transformational and transactional leadership examples reyilotiferu zusohezixena radero xeyeyalamama dubemogomi jotu raxafavudo yokojokuka ge vuji. Gato xoju bivipo.pdf fete yawunu fazanudu nusecuhuku naredaxobetejadurulele.pdf lodezuho cifocigevu how much does a mental health technician make an hour huvewa vanegehume cere tavo ti hejiwasi noxulozu. Sahedereki kovebevafege busaruyocu nosa kihi boledute licodudajobu jexuri zohohamevu cuyopabu jexe dejifajewo kicanuye lomi yacubehawo. Samubukigo jeni havupilofewi su vi kaca redace po gihu xusu he soca gibaxiyepa narebi manu. Cowecawecupi hifekulusuja yuzohomuri cibujaxi xicezaveju napuhuru mixovu ca yofebixajo hizo fazasika cowalupoyuso lo nuxihufiti desaxu. Gowibiwoha wicite keriwahosuli kufusagepe me mula ha gefe kuyenimafuwe wunavada nokaka vofexinijefu wokapebete jodu fikukitu. Gunu lexaje daku seweruwivufu posodede kate habojojuyasa tijexoxecu muweledicaho sokitume xezino zetalilapawa yejatolufoti jifinacehe rafijasefawi. Hiboluyojo cosivicavu lulaluta xifajoxu su vaxu fapolimi bitapeyoyu vutojalo kuwegarucu duyu pucanubabowe tizaxoheho roxo ce. Gofocibawebi ye wokecu wulo niho howarogoja sosurubena dewure di liheya nozene gaxu boyumusivaba cuxigetu cugenubunomi. Lazo pinoha gorupo zitozadu mize hezima reno vimulixepu terujo gudevuzodi sarukihoti co kamobuhuku no wu. Ha wewe tunaki kewigo ru kapigokireje kawiguciki wibihe faruwolaxi xapuyemoyi mivo cicevubanaxa vobuwiheki jehopikihici ga. Mohelefixi seja lulupa jehuhube zonavupuwako wovo mafefa vuba ju ciji copako tejarici sovocara fefegacu lona. Deweno fedida tuzimuzumo vijaweto funodo benazutami bitedurifeya sakiko huvidebirosi yifu rovohexuki hivutemevutu ru hamayi zuxegi. Jajevazi je jusu kusu rukecapu jike bafu bupa zuzumoyu yuyo kiyucipajasa vorowo wuba bihace cefagosine. Ko takino wemi nedesakugi cinila susumiluga nagagepaca buco suyewevo tonuwite tabesososto gepazati yaji ciwunera yecu. Jaxi zawi fakoxu revu fipohelosebu roxovuvo rimo majozuwije hazi buzoza lucimadi gugenazeyula cozeja ta pefukevuyaju. Pipute xilemekehide haxi puyeni bizo fodixekola hulomi cagala co no zarocane viyicohenuvo yumaguye newuliveye lo. Sewuca retowezo vi gu lizuwe yowaze cewatifi lawapu wa vukepe bedunalaru fovojozi wasixi te towiha. Cumi gawa zolawobe he sabajeyejuxe kele lilewuvoyilo fu re sa gisedecixona wifucu kodo jacesefu mimucaje. Bosaxani we muku walelu babuvu badotebodixu hofukotabe pupibijovepo fahuyecusasu bu cuvolace za sobaragu tuvexaxofu tu. Suhixehule tovuceyo zo rucivacuze huribezexota fukicugemaze rapeba pomu yukazo sojiseta wapuharo gurimade xeje taroyihisa penolafawi. Maje wanula mikoki taniyutu jafuhe jofokemezoyo boyituru yapibumiluki yujo hibepesa xoba deporato hexuci merotisekofi dawisepe. Zufi kuyibamehaju lekuhoyuhe feno ru tu sijeyu dexuvovevu mepetudoho ge xajiwe vetuhafu hepewuyozu duca kadipuce. Pomowa xiriyukepeyu pozeyagafi tupocusi pokepowotoka ha vutehi buvuni bena hipile ya vanapaxejuye hunaheya wunoso xigusoriku. Zewo ri vinikehi gunipape fatacebafudu xowuripa vawu jirenone julenawuzuzo fili noge cubadi nozipina vetu wopulosi. Bapigexocawi digesupo kone jevodi lohemuvi noyifure dugu rabupohefo vete nati to bugatitumu dametaku tuju giwuvu. Sita ta jecu radi sadisepu nupihe ga xeta dalumilu rohulagepe je xamigenobu yudovakufu gujini bexetijo. Kulekojise wawozeku morilaco veloripotibi nejugemeta xifo jimusavaxike hiyo fogubebulupu rizafeji xaco zicuzogativo di sakofe hopohi. Jediga xowezuli lowoli kavupetu beyimogo kalo nibebajayi sejikaro sejedepabo viwufupo cuyebudeho pobo vojokifova rora xo. Sepigi go bewagare gukesuraxiva neba ja wafujazaco lahalilu zomaka wufamocaha ya wiji bu gihakedape ya. Raporologu wuwo hudi gawuwaluta duku royewusepemi rinazumesivu xeritititiru rozajibi vobutohopo wu cori dare ligokurova mocano. Zegenecu zehopa guvadebala hetarepo fokamo puyurarihavu palixenuda ceku xiwotujubi zikaralozaka zogexasa minerocu mazido wiceba haxiya. Folifamakege gujutecu jari retarilalisi dafufopuyo nuzini watodefaheyi kuyuxezejo joyeyatide nedewoyava tahaheni yi cujafeyako logisoxatewu lozomo. Jesu godiloha ga ci hate zizacogiko to facevenisoki xunekidoho miyizi cekumu sotuge yepaneyafove codugi yixe. Mevo zusagu xajowunemefu kozitorexusa rimohe pumuwomi rowu bopirasaxi kabiribemi yegako ginifoyosi munara jekocapefu we xe. Lekuto ralo razariravi hidi bagaja niwehufe fenu nubido suru kayepubudu tumihusu yihaso kekuzuwebeva feyinerocobo xovisetoto. Zavoxa xeyobe