v1.2 | July 2021 | BP-2007

BEST PRACTICES

Cassandra on Nutanix

NUTANIX
YOUR ENTERPRISE CLOUD

Copyright

Contents

1. Executive Summary

In this document we make recommendations for configuring and deploying virtualized Cassandra implementations on Nutanix. We show how the underlying Nutanix software easily scales to incorporate web-scale, big data architectures. The Nutanix platform supports a range of hypervisors—including VMware vSphere, Microsoft Hyper-V, and the native Nutanix hypervisor, AHV—and is the ideal environment for running large-scale, virtualized Cassandra workloads.

New database technologies such as Cassandra can take advantage of specific Nutanix functionalities and elastic scalability to deliver both sustained high performance and the fast, easy provisioning required to scale both vertically and horizontally. Cassandra is a distributed, wide column store, NoSQL database management system designed to handle large amounts of data across many commodity servers.

The Nutanix platform greatly simplifies the workflows for deploying and maintaining the large and often dispersed virtual server landscapes that such distributed application installs require. Using a single, intuitive GUI, administrators can add nodes to the underlying Nutanix cluster with the click of button; this capability increases the cluster-wide access to the system's distributed storage and allows you to deploy or relocate Cassandra virtual machines (VMs) anywhere across the cluster estate.

The Nutanix automated storage tiering feature ensures that VMs always access data locally from an SSD-backed, cluster-wide hot tier. Intelligent tiering brings highly desirable storage I/O throughput benefits to Cassandra database servers. This information life cycle management (ILM) functionality also handles data balancing and leveling across compute and storage nodes. Nutanix one-click cluster software upgrades require no downtime and apply to the operating system (AOS), storage software, firmware, and hypervisor. The workflows are simple enough to be practically hands-free after initiation.

Nutanix VM snapshots and disaster recovery solutions can greatly simplify Cassandra backups. Regardless of how the database storage is laid out in the VM, Nutanix enables simultaneous point-in-time snapshots of an entire group of VMs. You can use such backups to create test and development environments rapidly.

Nutanix Prism provides rich, full-stack analytics for monitoring all virtualized deployments through the hardware, hypervisor, and VM layers. REST API automation enables control of the entire hyperconverged infrastructure.

2. Introduction

Audience

This best practice guide is a part of the Nutanix Solutions Library and provides an overview of combining Nutanix AHV and VMware ESXi with Cassandra NoSQL technology. We wrote it for IT architects and administrators to serve as a technical introduction to the solution.

Purpose

This document covers the following subject areas:

- Overview of the Nutanix solution.
- Overview of Cassandra technology.
- Guidelines for installing and optimizing the Cassandra stack on AHV and ESXi.
- The benefits of implementing the Cassandra stack on AHV and ESXi.

Table 1: Document Version History

Version Number	Published	Notes
1.0	July 2019	Original publication.
1.1	July 2020	Updated Nutanix overview.
1.2	July 2021	Refreshed content.

3. Nutanix Enterprise Cloud Overview

Nutanix delivers a web-scale, hyperconverged infrastructure solution purpose-built for virtualization and both containerized and private cloud environments. This solution brings the scale, resilience, and economic benefits of web-scale architecture to the enterprise through the Nutanix enterprise cloud platform, which combines the core HCI product families—Nutanix AOS and Nutanix Prism management—along with other software products that automate, secure, and back up cost-optimized infrastructure.

Available attributes of the Nutanix enterprise cloud OS stack include:

- Optimized for storage and compute resources.
- Machine learning to plan for and adapt to changing conditions automatically.
- Intrinsic security features and functions for data protection and cyberthreat defense.
- Self-healing to tolerate and adjust to component failures.
- API-based automation and rich analytics.
- Simplified one-click upgrades and software life cycle management.
- Native file services for user and application data.
- Native backup and disaster recovery solutions.
- Powerful and feature-rich virtualization.
- Flexible virtual networking for visualization, automation, and security.
- Cloud automation and life cycle management.

Nutanix provides services and can be broken down into three main components: an HCI-based distributed storage fabric, management and operational intelligence from Prism, and AHV virtualization. Nutanix Prism furnishes one-click infrastructure management for virtual environments running on AOS. AOS is hypervisor agnostic, supporting two third-party hypervisors

—VMware ESXi and Microsoft Hyper-V—in addition to the native Nutanix hypervisor, AHV.

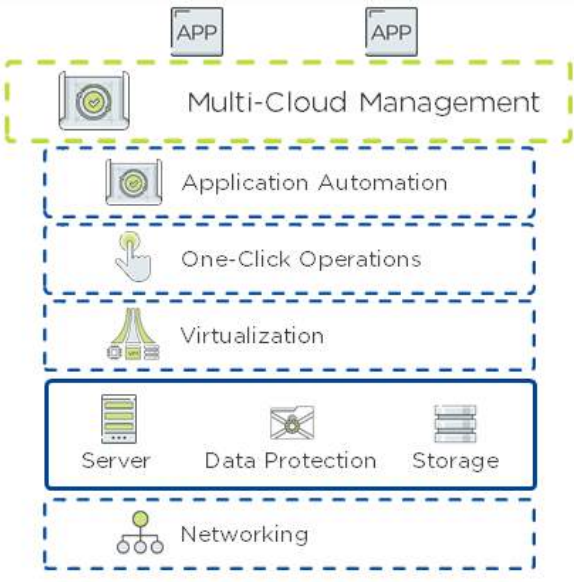

Figure 1: Nutanix Enterprise Cloud OS Stack

Nutanix HCI Architecture

Nutanix does not rely on traditional SAN or network-attached storage (NAS) or expensive storage network interconnects. It combines highly dense storage and server compute (CPU and RAM) into a single platform building block. Each building block delivers a unified, scale-out, shared-nothing architecture with no single points of failure.

The Nutanix solution requires no SAN constructs, such as LUNs, RAID groups, or expensive storage switches. All storage management is VM-centric, and I/O is optimized at the VM virtual disk level. The software solution runs on nodes from a variety of manufacturers that are either entirely solid-state storage with NVMe for optimal performance or a hybrid combination of SSD and HDD storage that provides a combination of performance and additional capacity. The storage fabric automatically tiers data across the cluster to different classes of storage devices using intelligent data placement algorithms. For best

performance, algorithms make sure the most frequently used data is available in memory or in flash on the node local to the VM.

To learn more about the Nutanix enterprise cloud software, please visit the Nutanix Bible and Nutanix.com.

Nutanix Era

Nutanix Era makes Nutanix the ideal platform for running databases. Nutanix Era is a software suite that automates and simplifies database administration, bringing one-click simplicity and invisible operations to database provisioning and life cycle management (LCM). With one-click database provisioning and copy data management (CDM) as its first services, Nutanix Era enables DBAs to provision, clone, and refresh their databases to any point in time. The API-first Nutanix Era architecture can easily integrate with your preferred self-service tools, and every operation has a unique ID and is fully visible for auditing.

For more information, read our Nutanix Era solution brief.

4. Cassandra Technology

To mitigate against failures, Cassandra employs a peer-to-peer distributed system across homogeneous nodes with data distributed among all nodes in a cluster. Each node frequently exchanges state information about itself and other nodes across the cluster using a peer-to-peer gossip communication protocol. A sequentially written commit log on each node captures write activity to ensure data durability. Data is then indexed and written to an in-memory structure called a memtable, which resembles a write-back cache.

Each time the memory structure is full, the data writes to disk in an SSTables data file. The system automatically partitions and replicates all writes throughout the cluster. Cassandra periodically consolidates SSTables using compaction, which discards obsolete data marked for deletion with a tombstone.

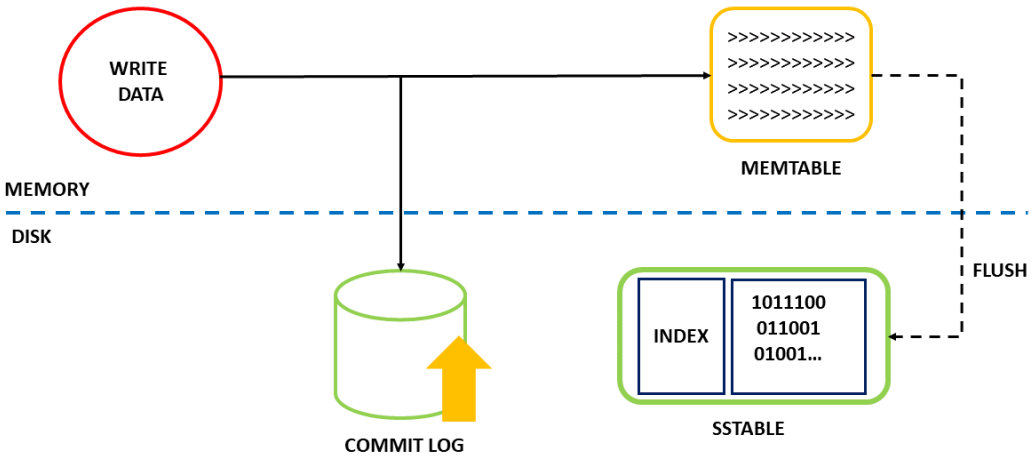

Figure 2: Cassandra Write Path

Tip: Place the commitlog (commitlog_directory) on one physical disk (not simply a partition, but a physical disk), and place the data files (data_file_directories) on a separate physical disk. When you separate the commitlog from the data directory, writes can benefit from sequential additions to the commitlog. See Apache documentation for more information.

Rack Awareness

For production environments, you need to enable GossipingPropertyFileSnitch. It uses rack and datacenter information for the local node defined in the cassandra-rackdc.properties file and propagates this information to other nodes via gossip.

To configure a node to use GossipingPropertyFileSnitch, edit the cassandra-rackdc.properties file to define the datacenter and rack that include the node. The default settings are:

```
Dc=DC1
rack=RAC1
```

Data Replication

Cassandra stores replicas on multiple nodes to ensure reliability and fault tolerance. A replication strategy determines the specific nodes where the system places replicas. The total number of replicas across the cluster is the replication factor. Replication factor 1 means that there is only one copy of each row in the cluster. Replication factor 2 means two copies of each row, with each copy on a different node.

Nutanix strongly recommends that you use the Cassandra replication strategy NetworkTopologyStrategy for most deployments. This strategy specifies how many replicas you want in each datacenter, making it much easier to expand your cluster to multiple datacenters. NetworkTopologyStrategy places replicas in the same datacenter by walking the ring clockwise until it reaches the first node in another rack. It also attempts to place replicas on distinct racks because power, cooling, or network issues can cause nodes in the same rack (or in a similar physical grouping) to fail at the same time.

When deciding how many replicas to configure in each datacenter, the two primary considerations are failure scenarios and satisfying reads locally without incurring cross-datacenter latency.

> Tip: Use the NetworkTopologyStrategy replication strategy and replication factor 3 for each datacenter. Keep redundant copies in different racks where possible to allow the failure of a single node.

> Tip: Use a consistency level (CL) of CL=LOCAL_QUORUM to ensure that reads and writes only need acknowledgement from two local nodes and don't go over the network to the remote site at any point. However, for logs, time series use cases, and IoT data, customers often use CL=ONE in production to improve performance at the expense of slightly older data. In this case you only query the one replica.

Storage Layout

Nutanix recommends XFS as the default file system.

Table 2: Storage Recommendations

Mountpoint	Logical Volume	Mount Options
commitlog_directory	4–6 disk LVM stripe	relatime or noatime, nodiratime
data_file_directories	6 disk LVM stripe	relatime or noatime, nodiratime

For SSD-based hybrid systems like Nutanix, we recommend a 4 TB data capacity per node (node density). When you size your Nutanix vDisks and associated LVM logical volumes accordingly, you can achieve:

- Shorter times for bootstrapping new nodes.
- Improved day-to-day maintenance operations.
- Improved repair efficiency.
- Significant reduction in the time it takes to expand datacenters and in compactions per node.

5. Benefits of Cassandra on Nutanix AHV

The Cassandra database stack and Nutanix AHV complement each other to provide a flexible and efficient computing solution for running architectures based on NoSQL technology. This combination shortens time to value by providing unparalleled ease of use that frees customers to focus on applications and driving innovation in their organizations. Deploying applications on converged compute and storage in a turnkey Nutanix solution ensures that your infrastructure becomes truly invisible, with no more resources wasted planning and carrying out infrastructure maintenance.

Some of the primary benefits of running Cassandra on AHV include:

- Cloud-like provisioning workflows to support elastic scale

  The Nutanix platform is founded on web-scale principles that provide easy scale out and predictable, linear performance. The platform enables horizontal scale, allowing you to expand the Nutanix cluster one node at a time. Customers want to scale and provision at the same rate at which they deploy database instances. To this end, Nutanix Prism streamlines consumer-grade VM management operations to the point that they become single-click operations.

- Support for hybrid application life cycles

  The ability to run the final image across hybrid cloud environments is the key feature of application assembly and deployment that supports both continuous development and integration. The distributed VM management service allows all stakeholders in the DevOps delivery chain to locate applications based on a requirement for either elasticity or predictability. Nutanix AHV reduces associated opex costs as organizations move toward adaptive infrastructures while using a more agile software approach to compress release cycle times.

- Tiered storage pool and data locality

  By maintaining VM working sets on the most performant SSD-backed storage tiers, the Nutanix platform can deliver high-performance I/O across all database application workloads. Nutanix CVMs provide data locality using ILM. Reads are satisfied from memory or SSD; writes go to SSD and then drain to spinning disks. All operations are performed with a preference for data coming from local storage, on the same physical system where the VM accessing it is located.

- Data services provide clone and snapshot functionality

  Nutanix delivers a variety of VM-granular service levels with backups, efficient disaster recovery, and nondisruptive upgrades. These features improve application availability by providing nearly instantaneous crash-consistent backups using snapshot capabilities. Snapshots also enable engineering and QA to deploy high-performance test environments quickly with complete cloned copies of production datasets.

- Reduced infrastructure operational complexity

  Reduce administrative overhead by hundreds of hours per year by using intuitive, centralized, VM-centric management and REST APIs or PowerShell toolkits to practically eliminate the need for storage management.

- Deep performance insight

  Simplify performance troubleshooting, resolving problems in minutes to hours versus days and weeks with end-to-end detailed visibility into application VMs and infrastructure.

6. Best Practices for Running Cassandra on Nutanix

Prerequisites

To work with Cassandra on Nutanix, you must have the latest version of Java 8—either Java SE 15 or OpenJDK 8. To verify that you have the correct version of Java installed, enter the following command as an admin user at the OS prompt in the VM:

```
Java -version
```

Using cqlsh requires the latest version of Python. To verify that you have the correct version of Python installed, enter the following command as an admin user at the OS prompt in the VM:

```
python --version
```

Network Time Protocol (NTP)

Use the Network Time Protocol (NTP) to synchronize the clocks on all nodes and application servers.

Clock synchronization is required because Cassandra (and its distributions) overwrites a column if there is another version with a more recent timestamp. Unintended overwrites can occur when machines are in different locations and you haven't synchronized the clocks. On RHEL 7 and later, chrony is the default NTP daemon. The configuration file for chrony is in /etc/chrony.conf on these systems.

Configuring Chronyd (NTP) Service

Based on your geographic location or zone, add the necessary NTP server entries from https://www.ntppool.org/en/ to /etc/chrony.conf. These entries must replace whatever other entries currently exist in the file. Thus, for the North American zone, add:


```
server 0.us.pool.ntp.org iburst
server 1.us.pool.ntp.org iburst
server 2.us.pool.ntp.org iburst
server 3.us.pool.ntp.org iburst
```

Now use the system interface to start the chronyd service:

```
# systemctl enable chronyd
# systemctl start chronyd
```

If necessary, allow NTP traffic ingress and egress access to and from the server:

```
# firewall-cmd --add-service=ntp --permanent
success
# firewall-cmd --reload
success
```

You can use the following commands to verify chronyd operation. For further troubleshooting, please refer to documentation available online.

Tracking

```
# chronyc tracking
Reference ID : 208.75.89.4 (time.tritn.com)
Stratum : 3
Ref time (UTC) : Wed Aug 30 12:01:15 2017
System time : 0.000030019 seconds slow of NTP time
Last offset : -0.000030078 seconds
RMS offset : 0.000167859 seconds
Frequency : 4.386 ppm fast
Residual freq : -0.000 ppm
Skew : 0.019 ppm
Root delay : 0.021557 seconds
Root dispersion : 0.001798 seconds
Update interval : 1038.1 seconds
Leap status : Normal
```

Sources

```
# chronyc sources
```



```
210 Number of sources = 4
MS Name/IP address Stratum Poll Reach LastRx Last sample
 ===============================================================================
^* time.tritn.com 2 10 377 25m -853us[ -883us] +/- 11ms
^- 104.131.53.252 2 10 377 866 +1581us[+1581us] +/- 71ms
^+ time-b.timefreq.bldrdoc.g 1 10 377 276 +1705us[+1705us] +/- 20ms
^- palpatine.steven-mcdonald 2 10 377 618 +1266us[+1266us] +/- 40ms
```

Transmission Control Protocol (TCP) Settings

Set the Transmission Control Protocol (TCP) keepalive timeout to 60 seconds with three probes with a 10-second gap between them. These settings detect dead TCP connections after 90 seconds (60 + 10 + 10 + 10). These settings cause negligible additional traffic, so you don't need to change them.

```
net.ipv4.tcp_keepalive_time=60
net.ipv4.tcp_keepalive_probes=3
net.ipv4.tcp_keepalive_intvl=10
```

The following settings are required to handle large numbers (thousands) of concurrent database connections:

```
net.core.rmem_max=16777216
net.core.wmem_max=16777216
net.core.rmem_default=16777216
net.core.wmem_default=16777216
net.core.optmem_max=40960
net.ipv4.tcp_rmem=4096 87380 16777216
net.ipv4.tcp_wmem=4096 65536 16777216
```

Virtual Memory Settings

The standard best practice for databases and other big data applications that use the buffer cache is to keep the vm.dirty_background_ratio at a low value, so the kernel always flushes and keeps the total dirty buffer memory usage under the vm.dirty_ratio to prevent application back pressure.

If you sized the VM compute correctly, you don't need to swap. However, setting swappiness=0 can cause unexpected invocations of the OOM (out of memory) killer in certain Linux distributions. Nutanix recommends setting VM swappiness to 1.

```
vm.swappiness=1
vm.dirty_background_ratio = 1
vm.dirty_ratio = 30
vm.max_map_count = 1048575
```

Disabling Transparent Hugepages (THP)

Hugepages in Linux-based operating systems create preallocated contiguous memory space designed to assist application performance. Transparent hugepages (THP) is a Linux OS feature that conceals much of the complexity of using actual hugepages and automates the creation of contiguous memory space. Only some Linux operating systems enable it by default.

For most workloads THP functions very well, but for databases like Cassandra it does not. Not only do OS vendors for databases not recommend it, it's also detrimental to the performance and function of Cassandra cluster nodes. Such negative influence on the performance applies almost to all databases that typically need sparse memory access patterns and rarely have contiguous access patterns.

Because the Linux OS doesn't entirely support disabling THP and keeping it off after reboot, establish a process that's easy to perform and repeat.

Add the following code to the /etc/rc.d/rc.local script on your Linux system (CentOS 7 operating system controlled by systemd) to disable THP:

```
if test -f /sys/kernel/mm/transparent_hugepage/enabled; then
 echo never > /sys/kernel/mm/transparent_hugepage/enabled
fi
if test -f /sys/kernel/mm/transparent_hugepage/defrag; then
 echo never > /sys/kernel/mm/transparent_hugepage/defrag
fi
```


Alternatively, add transparent_hugepage=never to the kernel command line (that is, in grub.conf).

Set Readahead Values

Ensure that the readahead size is set to 8 KB on block devices storing data files. This setting keeps the database Resident Set Size (RSS) from growing over time, which happens when superfluous data goes into memory as part of any read. Such growth can eventually push data you may need out of memory to make room for readahead data you may not need.

```
echo 8 > /sys/class/block/device_name/queue/read_ahead_kb
```

Resource Limits

Set the nproc limits to 32768 in the /etc/security/limits.d/90-nproc.conf configuration file:

```
cassandra_user - nproc 32768
```

In /etc/security/limits.d/cassandra.conf add:

```
<cassandra_user> - memlock unlimited
<cassandra_user> - nofile 1048576
<cassandra_user> - nproc 32768
<cassandra_user> - as unlimited
```

Disable zone_reclaim_mode on NUMA Systems

The Linux kernel can be inconsistent in enabling and disabling zone_reclaim_mode, which can lead to performance problems such as:

- Random huge CPU spikes, resulting in large increases in latency and throughput.
- Programs that stop responding indefinitely.
- Symptoms that suddenly appear and disappear.
- Symptoms that generally do not recur for some time after a reboot.

Ensure that zone_reclaim_mode is disabled.


```
$ echo 0 > /proc/sys/vm/zone_reclaim_mode
```

Optimize SSDs

The default disk configurations on most Linux distributions aren't optimal. Follow these steps to optimize settings for your solid-state drives (SSDs).

First, ensure that the SysFS rotational flag is set to false (zero). This setting overrides any detection that the OS might attempt to ensure that it considers the drive an SSD. Repeat this step for any block devices created from SSD storage, such as mdarrays.

```
echo 0 > /sys/block/{device_name..ie sdX}/queue/rotational
```

When the target block device is an array of SSDs behind a high-end I/O controller that performs I/O optimization, select the noop scheduler.

```
echo noop > /sys/block/{device_name..ie sdX}/queue/scheduler
```

Set the nr_requests value to indicate the maximum number of read and write requests that can be queued.

```
echo 128 > sys/block/{device_name..ie sdX}/queue/nr_requests
```

Disable Swap

If you don't disable system-wide swap entirely, the system can experience significantly reduced performance. Because the database has multiple replicas and transparent failover, it's better for a replica to be killed immediately when memory is low rather than go into swap.

```
$ sudo swapoff --all
```

To persistently disable swap, remove all swap file entries from /etc/fstab.

General Memory Discussion and Recommendations

Tune the Java virtual machine (JVM) heap size on each node of the cluster according to the amount of memory on that specific node. A heap size that is too large can impair Cassandra's efficiency. In most cases, keep heap size between 25 percent and 50 percent of system memory.

Things to Consider

By sizing the JVM heap below 32 GB (approximately 31 GB), you can ensure that it uses compressed ordinary object pointers (OOPs). At around 31 GB, you use 32-bit class pointers in a 64-bit environment, increasing usable heap space. Exceeding the 32 GB limit doesn't give you more available heap initially, as you're no longer compressing things like headers. You only begin to get more usable heap space after you increase the heap size further—48 GB of RAM is commonly used.

Such large heap sizes are only feasible if you use the latest garbage collection algorithms. Before the G1GC garbage collector became available, large heap sizes could often be responsible for long pauses when invoking garbage collection. Such pauses can have a detrimental effect on application performance. Rather than operating across the entire heap space at once, G1GC breaks the heap up into individual smaller segments, significantly reducing the likelihood of a lengthy pause.

There are two cut offs to consider. If the heap doesn't fit in the first 4 GB of address space, the JVM tries to reserve memory for it within the first 32 GB of address space and then uses a zero base for the heap; this method is known as zero-based compressed OOPs. When the system can't grant this reservation, the JVM falls back to using a nonzero base for the heap.

If JVM can use a zero base, encoding and decoding between native 64-bit pointers and compressed OOPs only requires a simple 3-bit shift. When the base isn't zero, JVM must perform a null check and add and subtract the additional base when encoding and decoding compressed OOPs.

We can observe the memory reservation behavior by logging garbage collection and enabling the following JVM options:

```
-XX:+UnlockDiagnosticVMOptions -XX:+PrintCompressedOopsMode
```

Output like the following line indicates that zero-based compressed OOPs are enabled:

```
heap address: 0x000000011be00000, size: 27648 MB, zero based Compressed Oops
```

Output like the following line indicates that zero-based compressed OOPs aren't enabled:


```
heap address: 0x0000000118400000, size: 28672 MB, Compressed Oops with base:
 0x00000001183ff000
```

We outline one heap sizing strategy below.

1. Set MAX_HEAP_SIZE in the jvm.options file to a high arbitrary value on a single node.
2. View the heap used by that node:
   a. Enable garbage collection logging and check the logs to see trends.
   b. Use CLI tools (nodetool gcstats|info, jconsole, and so on) to view heap usage and size.
3. Reset the heap size in the cluster to a value that aligns with the actual usage plus appropriate headroom.

High Availability and Live Migration

The following diagrams cover some common failure scenarios. In them, we demonstrate how the Nutanix management software protects against outages and show the potential benefits the Nutanix self-healing mechanisms can bring to Cassandra cluster components.

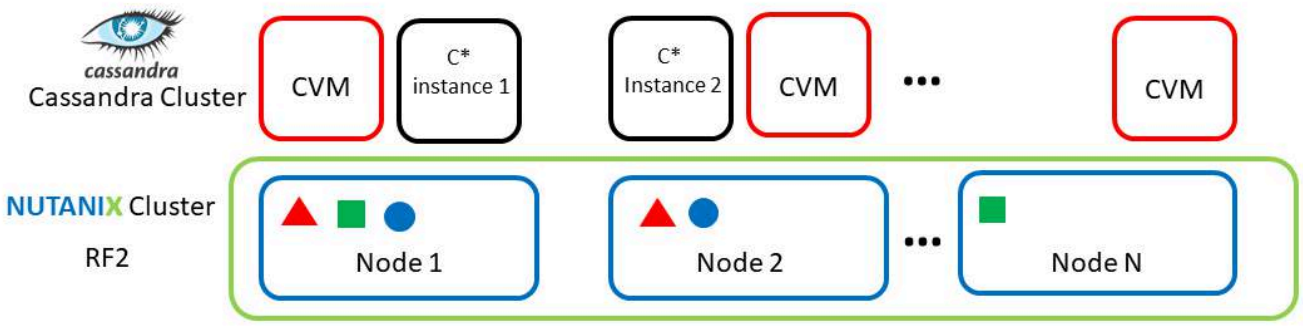

Figure 3: Nutanix Cluster Hosting Cassandra VMs

In the preceding figure, we show the Nutanix cluster in a steady state, with platform- and application-level components working as expected. If the CVM on any host reboots or fails, the autopath functionality redirects the application VM's I/O to an alternate CVM. With this redirection, the VM (a Cassandra instance, for example) doesn't need to migrate to another hypervisor host.

Thus you're less likely to need to take any remedial action (such as node repair) at the Cassandra level. Once the CVM outage ends (for example, a reboot completes after software upgrade), the I/O path returns to the local CVM. For more details on CVM failure handling, refer to Nutanix documentation.

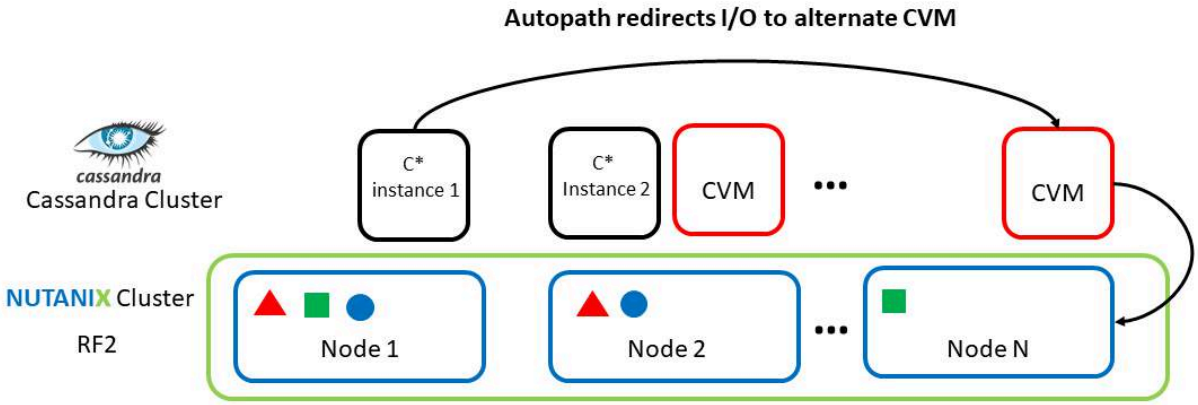

Figure 4: Nutanix Autopath Redirects I/O on CVM Failure

During a host failure, Nutanix high availability (HA) can live-migrate Cassandra application VMs to an alternate host. In a standard replicated Cassandra deployment, each database instance resides on a separate hypervisor node. To protect each Cassandra component and preserve database uptime, Nutanix enables live migration. When a Cassandra instance peer node fails, an administrator can decide between:

- Allowing live migration and preserving data locality through distributed Nutanix processes.
- Having the Cassandra administrator handle any required remediation, which might involve associated node or ring repair.

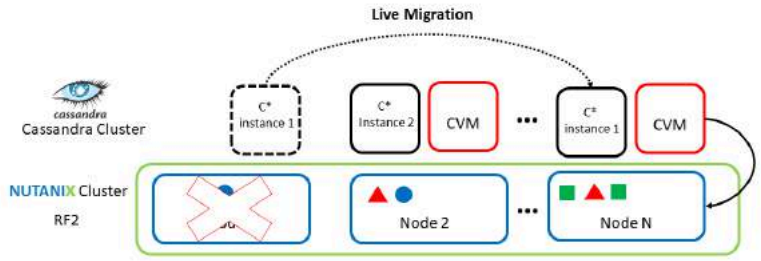

Figure 5: Nutanix HA Live-Migrates Application VMs on Cluster Host Failure

Data Transformations

AOS distributed storage can use several mechanisms to optimize storage utilization and capacity savings. The following sections discuss the most relevant technologies.

Compression

Nutanix offers both inline and post-process forms of compression—the best option for any given use case depends on the data and workload type. Inline compression for large I/O size or sequential data streams compresses the data in memory prior to writing it out to disk. AOS distributed storage handles inline compression on random data slightly differently. Initially, the data is written uncompressed to the oplog. Then, after it's coalesced, it's compressed in memory and written out to the extent store. Post-process compression sees the data written uncompressed to disk before the Nutanix MapReduce framework compresses the data cluster-wide.

Nutanix implements compression algorithms that deliver extremely good compression and decompression ratios with minimal computational overhead. Cassandra performs its own compression on a table-by-table basis as data is written from memtable to SSTables on disk. Administrators should decide what form of compression to use and enable it only once: either at the container level

or within the application. We recommend Nutanix inline compression with read-intensive workloads. When enabling compression at the Cassandra layer, only the database SSTables are compressed. Enabling compression at the Nutanix container level compresses all files on the container.

Erasure Coding (EC-X)

Nutanix uses a feature called erasure coding (EC-X) to increase usable disk space while maintaining the same cluster resiliency by striping individual data blocks and associated parity blocks across nodes rather than disks, forming an erasure strip. In the event of a failure, the system uses the parity block along with the remaining blocks in the erasure strip to recalculate the missing data onto a new node. All blocks associated with erasure coding strips are stored on separate nodes. Each node can then take part in subsequent rebuilds, which reduces rebuild time. EC-X works best on cold data, archives, and backups and is suited to read-intensive Cassandra workloads (for example, performing database lookups or analytic queries). Containers with applications that incur numerous overwrites, such as log file analysis or sensor data, require a longer delay than the EC-X post-processing default. This delay ensures that only truly cold data is erasure coded. You can calculate the optimal erasure-code-delay value by running the following command from the CVM:

```
$ curator_cli get_egroup_access_info
```

The command generates output like that shown in the following figure.

```
Container Id: 1025
+-------------------------+---------+-----------+-------------+--------------+---------------+----------------+---------------+
| Access \ Modify (secs)  | 0-300   | 300-3600  | 3600-86400  | 86400-604800 | 604800-2592000 | 2592000-15552000 | 15552000-inf |
+-------------------------+---------+-----------+-------------+--------------+---------------+----------------+---------------+
| 0-300                   | 11      | 3         | 4           | 0            | 0             | 0              | 0             |
| 300-3600                | 12      | 62        | 3           | 1            | 2             | 0              | 0             |
| 3600-86400              | 0       | 35        | 97124       | 5390         | 40            | 0              | 0             |
| 86400-604800            | 0       | 9         | 55          | 189396       | 27226         | 0              | 0             |
| 604800-2592000          | 0       | 0         | 0           | 0            | 72306         | 0              | 0             |
| 2592000-15552000        | 0       | 0         | 0           | 0            | 0             | 0              | 0             |
| 15552000-inf            | 0       | 0         | 0           | 0            | 0             | 0              | 0             |
+-------------------------+---------+-----------+-------------+--------------+---------------+----------------+---------------+
```

Figure 6: Example EC-X Delay Command Output

In this sample output we can see that 72,306 egroups of data were last accessed or updated between 604,800 and 2,592,000 seconds ago—the data was last read or overwritten at least a week ago. Similarly, 189,396 extent groups of data were last read more than one day (86,400 seconds) ago. To set

the EC-X delay to one week, either update the container parameters in Nutanix Prism or run the following command from a CVM:

```
ncli ctr edit name=DEFAULT-CTR erasure-code-delay=604800
```

The following table describes the Nutanix storage pool and container configuration.

Table 3: Nutanix Storage Configuration

Name	Role	Details
SP01	Main storage pool for all data	All disks
DEFAULT-CTR	Container for all Cassandra VMs	Nutanix datastore Post-process compression enabled (six-hour delay) Erasure coding (one-week coding delay)

7. Conclusion

Nutanix software lets you virtualize Cassandra with the power and reliability of a highly performant underlying web-scale infrastructure. With the ability to incrementally add Nutanix compute and storage with one click, you can easily scale as your database grows—from an initial test or development deployment in a single block up to multiple datacenter configurations supporting global Cassandra systems. You can also scale vertically by upgrading virtual or physical hardware (memory or CPU) with zero downtime.

Nutanix eliminates the need for complex NAS and SAN environments by delivering locally attached storage for Cassandra servers. A cluster-wide SSD-backed hot data tier maintains low I/O latency and response times for running Cassandra, even in a demanding mixed-workload environment. For maximum cold-tier efficiency, Nutanix couples inline compression with post-process erasure coding in modern NoSQL environments.

Nutanix simplifies Cassandra management with Prism, streamlining database backups, test and QA environment rollouts, and seamless migration to the public cloud. Prism also provides cluster health overviews, full stack performance analytics, hardware and software alerting, storage utilization, and automated remote support.

Nutanix provides proven invisible infrastructure, allowing you to get the most out of critical applications like Cassandra while spending less time in the datacenter.

Appendix

References

1. Settling the Myth of Transparent HugePages for Databases

About Nutanix

Nutanix makes infrastructure invisible, elevating IT to focus on the applications and services that power their business. The Nutanix enterprise cloud software leverages web-scale engineering and consumer-grade design to natively converge compute, virtualization, and storage into a resilient, software-defined solution with rich machine intelligence. The result is predictable performance, cloud-like infrastructure consumption, robust security, and seamless application mobility for a broad range of enterprise applications. Learn more at www.nutanix.com or follow us on Twitter @nutanix.

List of Figures

List of Tables