DATASTAX

DataStax Enterprise Reference Architecture

Table of Contents

Abstract

This white paper outlines reference architectures for deploying Apache Cassandra™ and DataStax Enterprise (DSE) within an organization and establishes a starting point for users contemplating hardware choices to host DataStax solutions. This paper offers guidance for system architects and administrators during the planning stages of development, test and production environments, whether in-house or at a private or public data center. It explores common scenarios and configuration options for various deployments.

Introduction

Innovations in database technologies continue to power and disrupt today's always-on, online world. There is an increasing focus with modern enterprises utilizing data as a strategic asset to compete. Companies are moving towards more "near term" analytics that can provide data insights in real time, so they can respond quickly to needs of customers. Because of this, online applications that interact with customers and collect data have zero tolerance for downtime and must be capable of reaching and interacting with their customer's data no matter where they are located. Decision makers are facing tremendous pressure to choose the right set of database tools and configuration to deliver these requirements in a scalable yet cost-efficient manner. DataStax Enterprise (DSE) addresses the requirements of modern, Internet Enterprise applications with its distributed and scalable database management platform built on Apache Cassandra™ with integrated analytics, enterprise search, security and in-memory capabilities.

Apache Cassandra is an open-source, NoSQL platform designed from the ground up to handle concurrent requests with fast writes and provide a low latency response for widely distributed users. DataStax Enterprise built on Cassandra offers flexible analytics so that users can leverage Spark and Shark to run "near real-time" analytics on Cassandra data or utilize Hadoop tools such as Hive, Pig and Mahout to carry out the analysis on Cassandra data (via integrated Hadoop or external Hadoop). In addition, DSE's integrated search (built on Solr) delivers features such as full-text search, faceting, hit highlighting and more.

Choosing the right database technology for an application is imperative for the success and scale of an online application. Planning for hardware and accompanying software ensures that the system remains robust when scaled. This whitepaper provides tips to proactively configure and deploy a DataStax Enterprise system. Use cases play an important role in determining specific information system requirements, application access patterns and mix of workloads that will be present in the end result.

DataStax Enterprise Architecture

Built on a production-certified version of Apache Cassandra, DataStax Enterprise (DSE) encapsulates a distributed peer-to-peer architecture in which all nodes are equal, ensuring cluster resilience to an arbitrary number of node or data center failures. DSE transparently distributes data across the cluster permitting cluster performance to scale linearly with the number of nodes, at a petabyte scale.

As illustrated in Figure 1, Keyspaces are abstractions similar to schemas in relational databases. Keyspaces are used to group tables together where data is organized by table and identified by a primary key. The primary key determines the node(s) in the cluster on which data is stored.

Cassandra stores copies called replicas, of each row based on the row key. The number of replicas is defined when a keyspace is created using replica placement strategy. This strategy sets the distribution of the replicas across the nodes in the cluster depending on the cluster's topology.

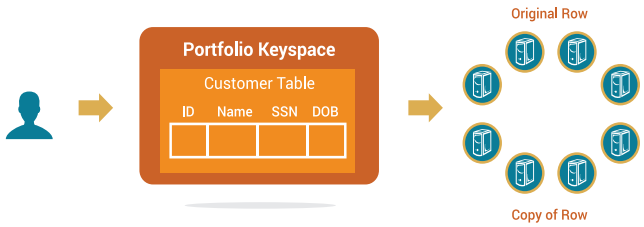

Figure 1 - Keyspace and Cassandra Ring

An insert operation is first written to commit log for durability and then to a memtable as shown in Figure 2. It is later flushed to an SStable once the memtable is full.

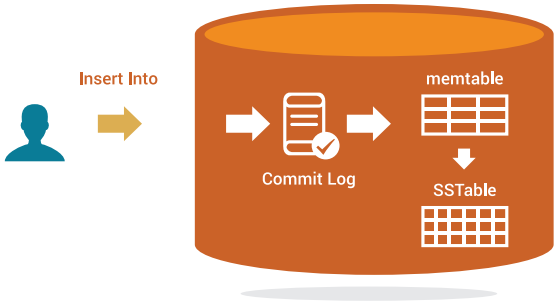

Figure 2 - Writes in Cassandra

Each node can be assigned a combination of dedicated functions from real-time, analytics, and search processing as shown in Figure 3. Data is distributed and replicated across the real-time / online and analytic groups. Solr nodes also use Cassandra for storage, with data inserted on Cassandra nodes being automatically replicated, indexed and made available for search operations on Solr nodes.

Figure 3 – DataStax Enterprise Cluster

OpsCenter: Easy-to-Use, Visual Management Interface

DataStax OpsCenter is a visual management and monitoring interface that administers Cassandra and DSE clusters. Each node in a DSE cluster has a DataStax agent installed by default, which is used by OpsCenter to manage and monitor the cluster. OpsCenter automates many system and database administrator activities effectively through its web-based, visual interface.

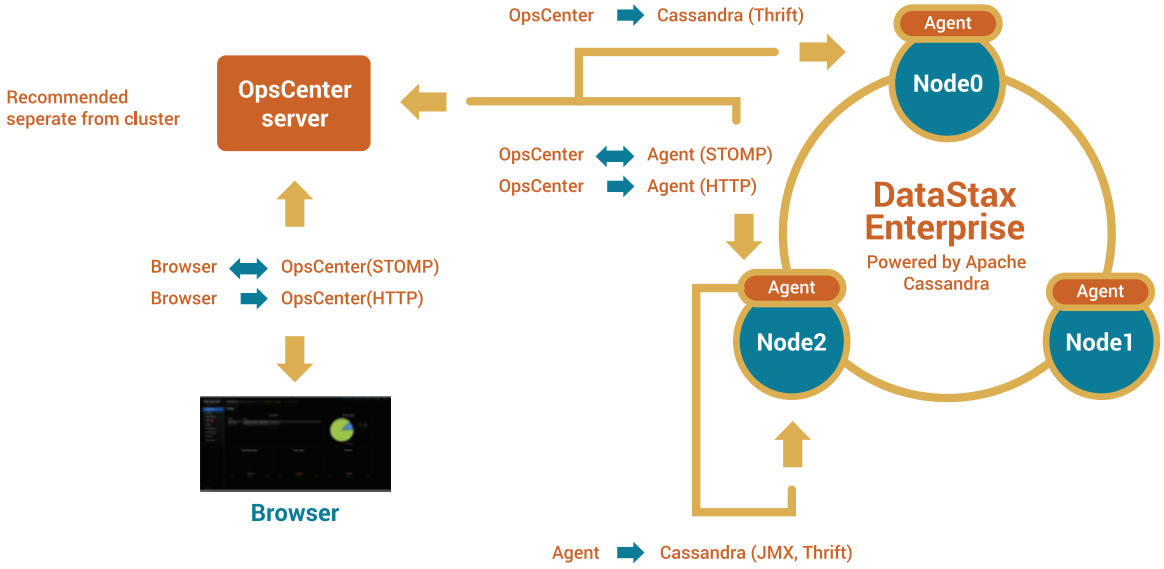

Figure 4 – OpsCenter Architecture

Capacity Planning

Storage, memory and CPU are key during the initial capacity-planning phase. Monitoring storage and memory usage on existing nodes and adding more capacity when needed can easily be done in a number of ways. Per-node data and memory usage can be monitored visually with DataStax OpsCenter or via the command line with Cassandra's nodetool utility, and proactively take action when a predetermined threshold is met (OpsCenter enables you to configure alerts that notify you when this happens).

The Capacity Service of DataStax Enterprise is an automated service helpful for already existing clusters. Capacity Services uses trend analysis to help understand how a cluster is performing within its current environment and workload. It uses historical data to help understand future resource needs of a cluster along with a timeline for this need. Admins can forecast, for example, disk usage for upcoming months (based on the prior three months of usage) to determine additional capacity needs.

Storage

Estimating transaction arrival rates and retained data volume are key factors for computing data storage. Many companies forecast months ahead to determine anticipated storage. Selecting the type of storage medium is also important: SSD and magnetic hard disks have different performance characteristics, and should be considered carefully based on the workload. The key point to acknowledge is the elimination of seek time and the penalty associated with non-contiguous data on physical media. While Cassandra helps mitigate these issues by writing sequentially only, SSD drives can scale up to cope with larger compaction overheads while simultaneously serving many random reads.

Shared Storage – An Anti-Pattern to Avoid

DataStax ***strongly*** recommends that shared storage (e.g. SAN, NAS, etc.) for Cassandra and DataStax Enterprise ***not*** be used. Use of shared storage defeats the purpose of Cassandra's continuous availability feature set and delivers performance that will disappoint in most every use case as compared to deployments that use local storage.

Free Disk Space Planning for Compaction

An operational storage requirement is the need to keep sufficient disk space free and available for Cassandra's compaction. Compaction is a mechanism used to improve data locality and flush stale data from the system. The default compaction strategy is "size-tiered" (STCS), which temporarily may require up-to double the size of the table it is compacting. Since multiple tables may be compacted simultaneously, it is prudent to keep free space in excess of your live data volume, to ensure optimal performance.

Leveled Compaction Strategy (LCS) dramatically reduces the temporary disk space requirements, while providing tighter bounds on the latency of read operations. The tradeoff is that LCS generally produces twice as many write operations over time. LCS is generally not recommended for write-heavy workloads for this reason. To get a better handle on the tradeoff, it is possible to forecast compaction I/O impact in 2.x Cassandra releases if you have an existing, running cluster to use as a basis for estimation. Please contact DataStax Technical Support to discuss your LCS-based use case and determine the best free disk space headroom estimate.

Hadoop workloads that use built-in DSE Hadoop components often create large volumes of intermediate results that are stored in the Cassandra File System (CFS). If you plan to use Analytics nodes i.e. Hadoop that run built-in DSE Hadoop tasks, you will want to review the CFS Compaction strategy and compare with your analytics workloads. On a per table basis, free disk space headroom for the CFS use case is no worse in magnitude than the LCS use case.

Free Disk Space Planning for Snapshots

Cassandra backs-up data by taking a snapshot of all on-disk data files (SSTables) stored in the data directory. Users can take a snapshot of all keyspaces, a single keyspace, or a single table while the system is online. Snapshots can be taken automatically depending on your configuration and should be actively managed. (Refer to auto_snapshot and snapshot_before_compaction settings in cassandra.yaml.) Administrators need to factor in sufficient storage capacity for the length and frequency of retention required. Sometimes a huge amount of free space is not necessary if the workload is read-heavy.

Secondary Indexes and Solr Index Overhead

Though highly dependent on the schema design, if secondary indexes on Cassandra tables are deployed there will be more disk space needed, proportional to the columns indexed. In the worst case, if each column had a secondary index, one might need double the normal disk space.

For Search/Solr nodes, indexed content is stored in Cassandra tables, so roughly twice the disk space is needed for storing the content alone. This is due to the space needed for a Lucene keyword index on each node. DataStax Customer Operations or Technical Support helps customers arrive at reasonable per-node data storage targets for their particular use case.

CPU

Insert-heavy workloads are CPU-bound in Cassandra before becoming memory-bound. That is, all writes go to the commit log, but Cassandra is so efficient in writing that the CPU is the limiting factor. Cassandra is highly concurrent and uses as many CPU cores as available. Hence, if an application uses search/Solr nodes, plan for increased CPU consumption during indexing, where an index backlog is constructed from unique keys of data stored in Cassandra. One or more CPU cores will be fully utilized when the indexing service works off the backlog. The indexing is a multi-threaded operation and can be configured to use your system resources optimally, without impacting other workloads.

If an application uses DSE Spark Streaming/Analytic nodes, plan for high CPU (16 cores or more). This generally depends on the workload, but if the job requires a high amount of processing of a small amount of data, high CPU should be the goal.

Memory

It is important to ensure that sufficient RAM is available on a per node basis. RAM is used by Java Virtual Machine (VM), by the filesystem buffer cache, by the operating system and by Cassandra. Some of the big data structures include the bloom filter, row cache, key cache, compression metadata and memtables. The operating system, Java VM and DSE all have fixed overhead. For memory capacity planning purposes, allow 4 GB for fixed overhead. A general guideline is to allow plenty of memory for the filesystem buffer cache. Another rule of thumb is to devote half the server memory to the Java VM used by Cassandra, with a cap of 8 GB.

Analytics Considerations

For Hadoop, memory requirements stem from the number of tasks to be executed in parallel. With DSE's new Spark analytics, it is much lighter on memory than Hadoop as it does not use separate JVMs per task. Allow at least 50% more memory or 2x more for Spark alone than the size of the data you plan to cache in order to fully leverage Spark's in-memory caching capabilities. Heap size must be monitored as larger heaps can introduce GC pauses/issues that can increase wait times.

Shark is a Hive-compatible system built on Spark that stores results of a query in memory. It is recommended to allocate additional memory when in-built caching is enabled.

DSE-Spark Streaming requires more memory when holding large amounts of data in memory for analysis. Allocating sufficient RAM (64GB or more) based on the use case is recommended.

Bloom Filter Considerations

The bloom filter size is directly proportional to the number of unique row keys in the Cassandra database. For any given node and table, the local hash-table-based bloom filter is checked for the possible existence of a row given its row key value. If present in the bloom filter, the index cache and/or file are read and finally the row is read from the SSTable on disk. Bloom filter, index and row cache sizes are tunable.

Other General Recommendations

When considering the transient memory needed for a given mix of workloads, it will be best to engage with DataStax Customer Operations or Technical Support to formulate the best advice for a particular situation. The following guidelines are generalizations meant as qualitative advice:

- 32 GB or more are practical server memory capacities for Cassandra use cases
- 32 GB to 64 GB are appropriate server memory capacities for many search use cases
- 32 GB, 64 GB and 128 GB are appropriate memory capacities for light, moderate and heavy analytics use cases, respectively
- Depending on the number of map and reduce workers configured per node, 8 GB to 32 GB can be needed for analytics
- To keep Java VM "garbage collection runs" to a manageable time frame (under a quarter of a second), it is advantageous to target an 8 GB or smaller heap.
- With reference to configuring DataStax OpsCenter, having dedicated hardware for the core service is recommended with 2 cores and 2 GB of RAM.

DSE Reference Architecture

The following section provides a guide for common architecture deployments as dedicated hardware systems or in the cloud.

Physical Architecture (On-premise deployment)

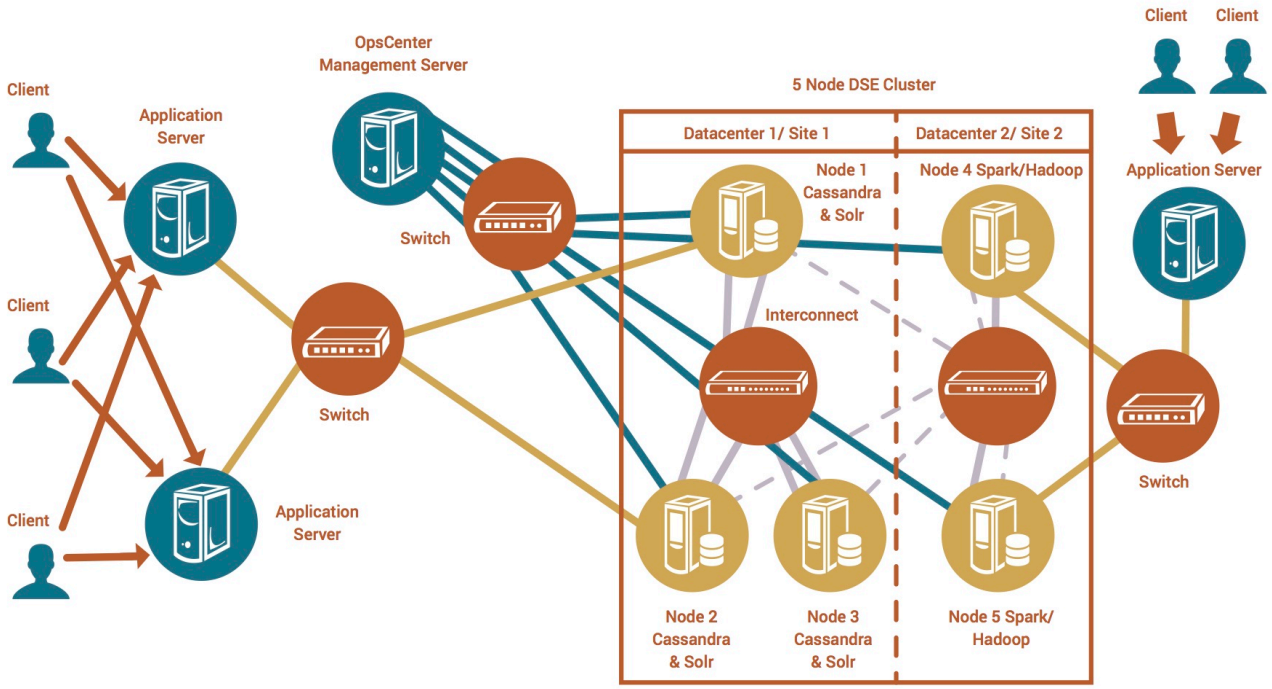

Figure 5 – Physical Architecture (on-premise deployment)

More nodes (Node 6, Node 7…Node N) may be added to the architecture for additional capacity and throughput. These nodes can span different racks across different data centers. Data centers can either be in the same physical location (for failover/upgrades etc.) or can span different physical locations. Cassandra & Solr (node 1, 2 and 3) are running in the same JVM.

Dedicated Components

Hardware Components
Intel Xeon or AMD processor based (64 bit)
1 GbE or 10 GbE switch
SATA3 or SAS or SSD for storage
SATA3 or SAS for commit log

Software Components
Cassandra 2.x, DataStax Enterprise 3.2 or beyond
OpsCenter 4.0 or beyond
CentOS/Debian/Ubuntu/Red Hat/Suse/Oracle Linux

Platforms support
http://www.datastax.com/what-we-offer/products-services/datastax-enterprise/platforms#navtop

Physical Architecture (Cloud Deployment)

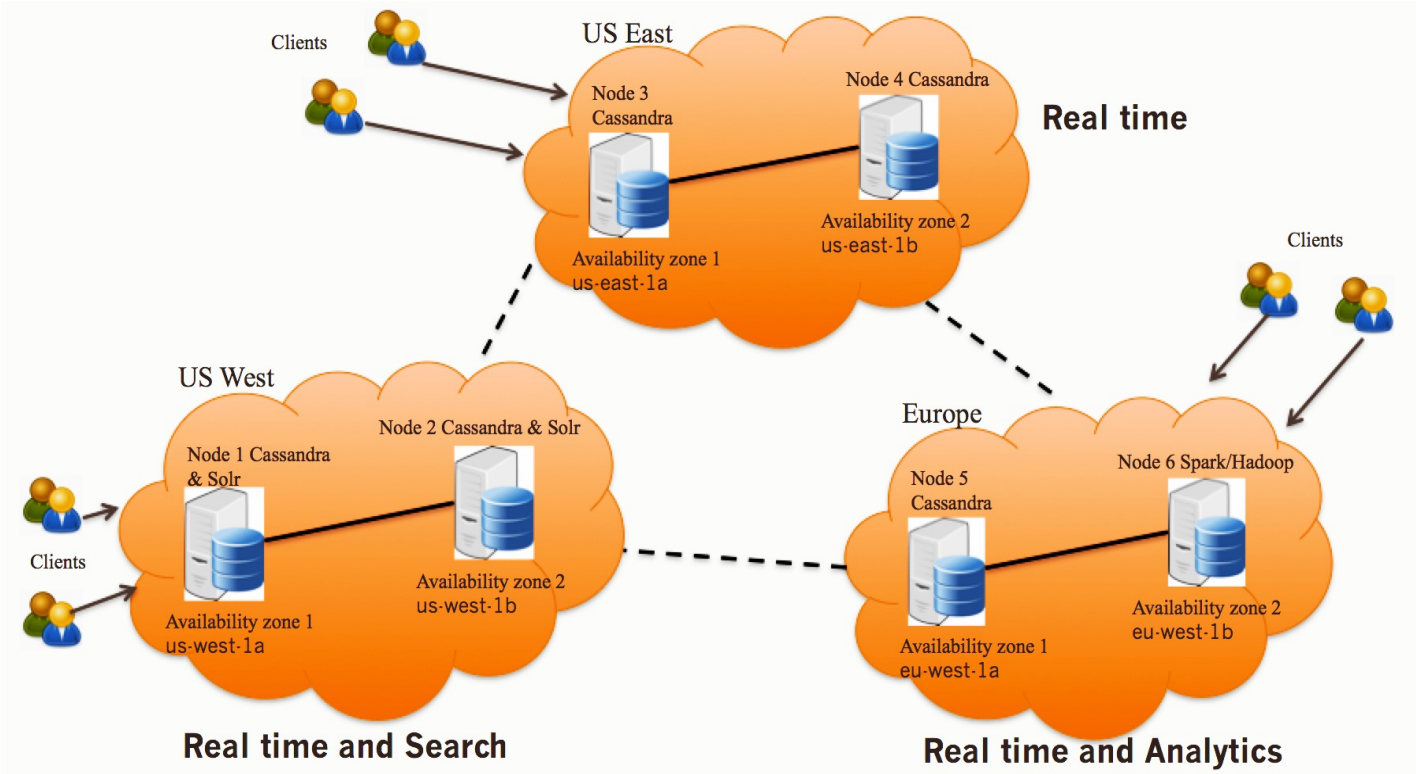

Figure 6 – Physical Architecture (Cloud Deployment)

Cloud Components

Hardware Components
m1.XLarge instance or more
1 GbE or 10 GbE switch
Regular or SSD for storage

Software Components
Cassandra 2.x, or DataStax Enterprise 3.2 or beyond
OpsCenter 4.0 or beyond
CentOS/Debian/Ubuntu/Red Hat

Deployment Scenarios

Low End Specification (Cassandra Only)

Server	Intel Xeon or AMD modern processor (64 bit) 8 cores or more
RAM	**16 GB to 32 GB of ECC DRAM per node
Storage	SATA III hard disks or SAS drives
Network	Gigabit Ethernet
Data Size	300 GB to 1 TB per node on SATA III Up to 1 TB per node on SAS

** While some users report good results with 16GB RAM, DataStax recommends 32GB as minimum RAM support for production workload.

A typical cluster running Cassandra may have four or more nodes, with a replication factor of 3 (RF=3). The replicated data often spans two physically separated data centers.

Given the low-specification hardware mentioned above, when file systems are striped across additional SATA III hard drives, architects must manage a data size of 500 GB (including the extra copies due to replication factor) per node

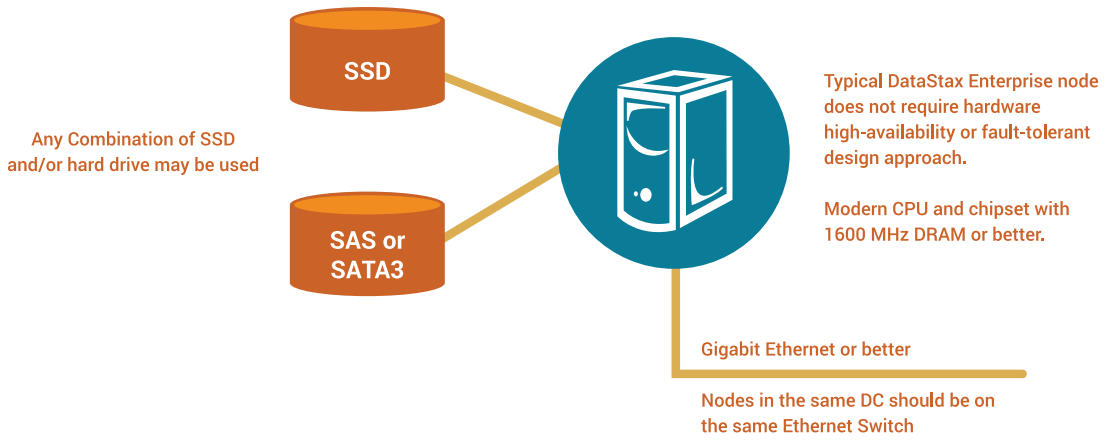

Figure 7 – Deployment Scenarios

Up to 1 TB per node is achievable with SAS drives because SAS drives are generally lower capacity. More spindles are likely to be used relative to SATA. SAS drives typically perform better with lower latency and higher throughput than SATA counterparts. Three-way striping is recommended if volume management is utilized.

Rotating magnetic hard drives should be striped three ways or more. DSE supports distributing I/O across multiple data directories when volume management is not utilized. Please note that RAID 1 or RAID 1+0 can be used if RF=1. If RF=3 or higher, the system administrator can generally rely on database-level replication for data durability.

Solid State Disk (SSD) may be striped two ways, however, separate PCI-e controllers should be used for each SSD. SSD's always outperform their mechanical counterparts but are typically used in performance-oriented, higher-specification examples as mentioned below.

High End Specification (Cassandra Only)

Server	Intel Xeon or AMD modern processor (64 bit) 16 cores or more
RAM	128 GB to 256 GB of ECC DRAM per node
Storage	SATA III hard disks or SAS drives or SSDs
Network	10-Gigabit Ethernet
Data Size	300 GB to 1 TB or more per node Up to 5 TB per node using SSD exclusively, for keybased access pattern

Demanding applications are frequently focused on minimal latency for end-user requests, or maximum throughput for batch processing. Both approaches typically result in the selection of high end, state-of-the-art equipment.

DataStax recommends using Solid State Disk (SSD). 400 MB/sec or more can be expected for a modern SSD drive. Additionally, DataStax recommends striping across more than one SATA controller for the most demanding applications. Use of a multi-lane SATA controller (PCIe 2.0 x4 or better) may be worth the premium to ensure maximum disk throughput, especially with multiple SSDs. Solid State Disk (SSD) may be striped two ways and care should be taken to ensure that the controller is not a bottleneck.

Transactional volume per cluster depends on the use case (like average record or row size). Users must estimate and calculate the number of records and size of each record to come to a conclusion on transaction rate.

Cloud Specification (Cassandra only)

This section covers the basics of sizing in cloud deployments. A cloud specification assumes low performance hardware (as with the low specification mentioned above), on per server or per blade basis. Examples below are for public clouds, such as Amazon AWS EC2 and Google Cloud Compute Engine (GCE).

Amazon AWS EC2

Data per node	Instance Type	Memory	Disk	Network Performance
< 100 GB	m1.xlarge	15 GB	Ephemeral	High
< 100 GB	c3.2xlarge	15 GB	SSD	High
< 1 TB	i2.2xlarge	60 GB	SSD	High

Google Cloud Compute Engine

Data per node	Instance Type	Memory	Disk
< 200 GB	n1-standard-8	30 GB	2 TB persistent disk
< 1 TB	n1-highmem-16	104 GB	4 TB persistent disk

Private Cloud Hardware

Data per node	CPU/Chipset	Memory	Disk	Network
< 500 GB	4 physical core, single socket motherboard	30 GB	SSD	GigE e.g. 1000base-T
< 1 TB	2 socket motherboard Ivy Bridge 4 physical core	64 GB	SSD	GigE e.g. 1000base-T or GBIC
< 5 TB	4 socket Sandy Bridge, 6 physical core	128 GB	Multi-controller SSD	10GbE e.g. 10GBase-x, SFP+

Workload-Specific Examples

DataStax Enterprise addresses different needs depending upon the application requirements and usage for DSE-Cassandra, DSE-Search and DSE-Analytics.

DSE/Cassandra	
Workload	Writes/second: From thousands per node to millions per cluster, depending on row size, block I/O subsystem performance and CPU integer performance. Queries: Sub-second to sub-millisecond response times Data sizing: Hundreds of TB, up to PB range
Use Case	High transaction-rate, write-mostly or balanced read and write transaction workloads characterize real-time use cases. This is the typical web application object persistence mechanism for many modern and larger-scale web apps. For example, Netflix streaming media metadata, including users, accounts, genres, preferences, sessions, bookmarks—but not movie content—are managed in DSE/Cassandra, supporting terabytes of streaming media (30% of the internet on Friday nights), at write speeds exceeding 15M writes/second. Reference customers - http://www.datastax.com/customers.
Simple scale-out patterns	Cassandra was originally optimized for x86 commodity servers. The storage engine leverages log structured merge trees, which result in purely sequential writes to maximize performance, and inter-node operations enable fast interactions across commodity networks. Leverage these basic building blocks to grow/shrink deployments in a stepwise manner for both on-premise and cloud deployments. **Scaling:** Auto-scale based on transaction arrival rates first, then the data storage volume as you learn the data access patterns and how they depend on data volume.
High-density patterns	Cassandra customers looking to achieve operational efficiency in terms of minimizing energy usage and floor space. These customers typically deploy non-redundant hardware and utilize high-capacity SSD exclusively (as mentioned in the high end specification section above).

DSE/Search (with Solr)

Workload	**Documents:** Hundreds to billions **Queries:** Sub-second on down to sub-millisecond response times **Data sizing:** Tens of TB
Use Case	Current Solr use cases can be broadly characterized as either write-heavy or read-heavy. For example, a major U.S. retailer supports a universal product catalog (every SKU, every currency, every language, with 40% change per day) and all searches on their website now traverse DSE/Solr.
Simple scale-out patterns	DSE/Solr exerts high CPU demands and memory (heap and OS buffers) due to ingest/indexing and for complex queries. The benefit of smaller nodes is that Solr auto-scales/auto-shards as the DSE cluster grows/shrinks. Avoid vnodes (num_tokens > 1) on any DSE/Search nodes.
Storage-density patterns	100 GB per node or more is a good target for nominal hardware running DSE Search. An equal amount of room must be dedicated to the Solr/Lucene search index as well as to data in Cassandra.

	DSE Analytics (via Spark/Hadoop)
Workload	**Batch-based job processing:** Near real-time analytics with Spark (seconds, minutes). Hours/days of elapsed time to compute analytics using integrated/external Hadoop. **Data sizing:** TB to PB
Use Case	Analytics workload requirements vary by data model and by the type of question asked. Machine learning such as click prediction, spam filters, personalization and search ranking are an ideal fit for DSE-Spark Analytics. DSE's Spark Streaming enables faster analytical capabilities for a variety of use cases such as near real time fraud detection, ad targeting and anomaly detection. Summary roll-ups, statistical measures or multivariate regression (trend determination) are familiar patterns with a predictable workload characteristic. Any automatable task can be realized in Java, compiled and submitted as a Hadoop job, which then iterates over a dataset. Constant Contact, for example, uses DSE/Analytics to refine customers' email campaign through a broad array of batch analytics.
Simple scale-out patterns	DSE customers running analytics typically have a purpose-built Spark/Hadoop clusters running alongside DSE. Spark is provided for near real-time analytics on Cassandra data and Hadoop is provided to run batch analytics on Cassandra data. DSE's integrated Hadoop uses the Cassandra File System (CFS) as a foundation layer for Hadoop support. This capability should not be used as a general Hadoop data warehouse with cold storage as CFS is just an alternative to HDFS and not a replacement. Performance is a function of the algorithm executed and typically characterized by high memory demands. High CPU demands may also be present depending on the nature of the Map/Reduce algorithm.
High-density patterns	Customers running analytics on massive scale (PB range) should utilize external Hadoop integration offered by DataStax. This integration allows running batch analytics on Cassandra data from external Hadoop clusters through Map Reduce, Hive, Pig or Mahout. The results of MapReduce can be stored in Cassandra or Hadoop.

Conclusion

This reference architecture paper is intended to deliver a foundation for choosing hardware components and for educating users on the right deployment strategy for specific use cases. DataStax recommends engaging with our technical experts for support and guidance on specific scenarios.

About DataStax

DataStax, the leading distributed database management system, delivers Apache Cassandra to the world's most innovative enterprises. Datastax is built to be agile, always-on, and predictably scalable to any size.

DataStax has more than 500 customers in 45 countries including leaders such as Netflix, Rackspace, Pearson Education and Constant Contact, and spans verticals including web, financial services, telecommunications, logistics, and government. Based in Santa Clara, Calif., DataStax is backed by industry-leading investors including Lightspeed Venture Partners, Meritech Capital, and Crosslink Capital. For more information, visit DataStax.com or follow us @DataStax and @DataStaxEU

.