Unit-4
Cloud Storage

-Dr. Radhika V. Kulkarni

Associate Professor, Dept. of Computer Engineering,

Vishwakarma Institute of Technology, Pune.

DISCLAIMER

This presentation is created as a reference material for the students of TY-Comp. Engg., VIT (AY 2022-23 Sem-2).

It is restricted only for the internal use and any circulation is strictly prohibited.

Syllabus

Unit-IV Cloud Storage

[CO4PO1, PO2, PO3, PO4, PO5, PO6, PO9- Strength 3,2,2,2,3,3,3]

Storage options in the cloud, Structured and unstructured storage in the cloud, unstructured storage using Cloud Storage, SQL managed services, Exploring Cloud SQL, Cloud Spanner as a managed service, NoSQL managed service options, Cloud Datastore, a NoSQL document store, Cloud Bigtable as a NoSQL option. OpenStack: NOVA, Neutron, Keystone Cinder, Swift and Glances, VMware Suit, Apache Cloud Stack [4 Hrs]

Storage Options in the Cloud

What is cloud storage?

- **Cloud storage** is a cloud computing model that enables storing data and files on the internet through a cloud computing provider that you access either through the public internet or a dedicated private network connection.

- **The provider** securely stores, manages, and maintains the storage servers, infrastructure, and network to ensure you have access to the data when you need it at virtually unlimited scale, and with elastic capacity.

- **Cloud storage removes** the need to buy and manage your own data storage infrastructure, giving you agility, scalability, and durability, with any time, anywhere data access.

Why is cloud storage important?

- Cloud storage delivers cost-effective, scalable storage.
- You no longer need to worry about running out of capacity, maintaining storage area networks (SANs), replacing failed devices, adding infrastructure to scale up with demand, or operating underutilized hardware when demand decreases.
- Cloud storage is elastic, meaning you scale up and down with demand and pay only for what you use.
- It is a way for organizations to save data securely online so that it can be accessed anytime from any location by those with permission.
- Whether you are a small business or a large enterprise, cloud storage can deliver the agility, cost savings, security, and simplicity to focus on your core business growth.
- For small businesses, you no longer have to worry about devoting valuable resources to manage storage yourself, and cloud storage gives you the ability to scale as the business grows

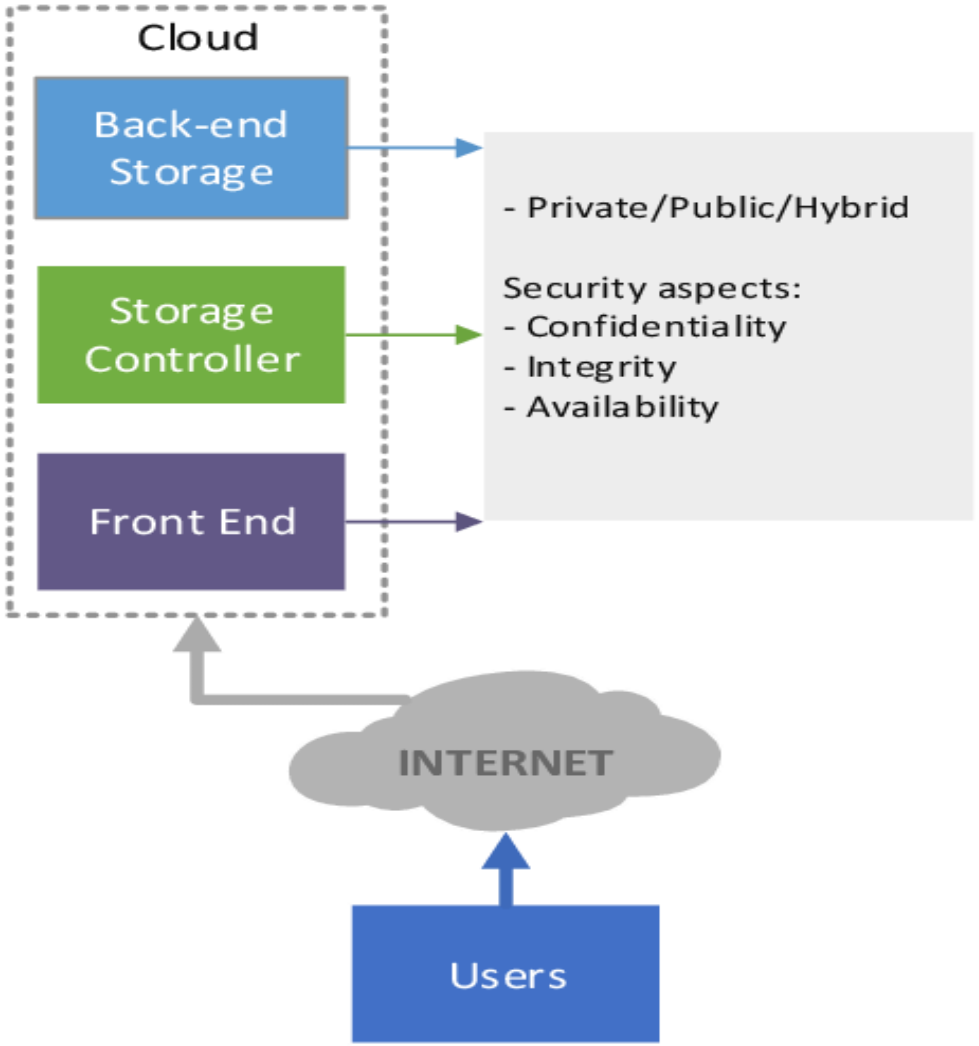
Why is cloud storage important?

- Cost effectiveness
- Increased agility
- Faster deployment
- Efficient data management
- Virtually unlimited scalability
- Business continuity

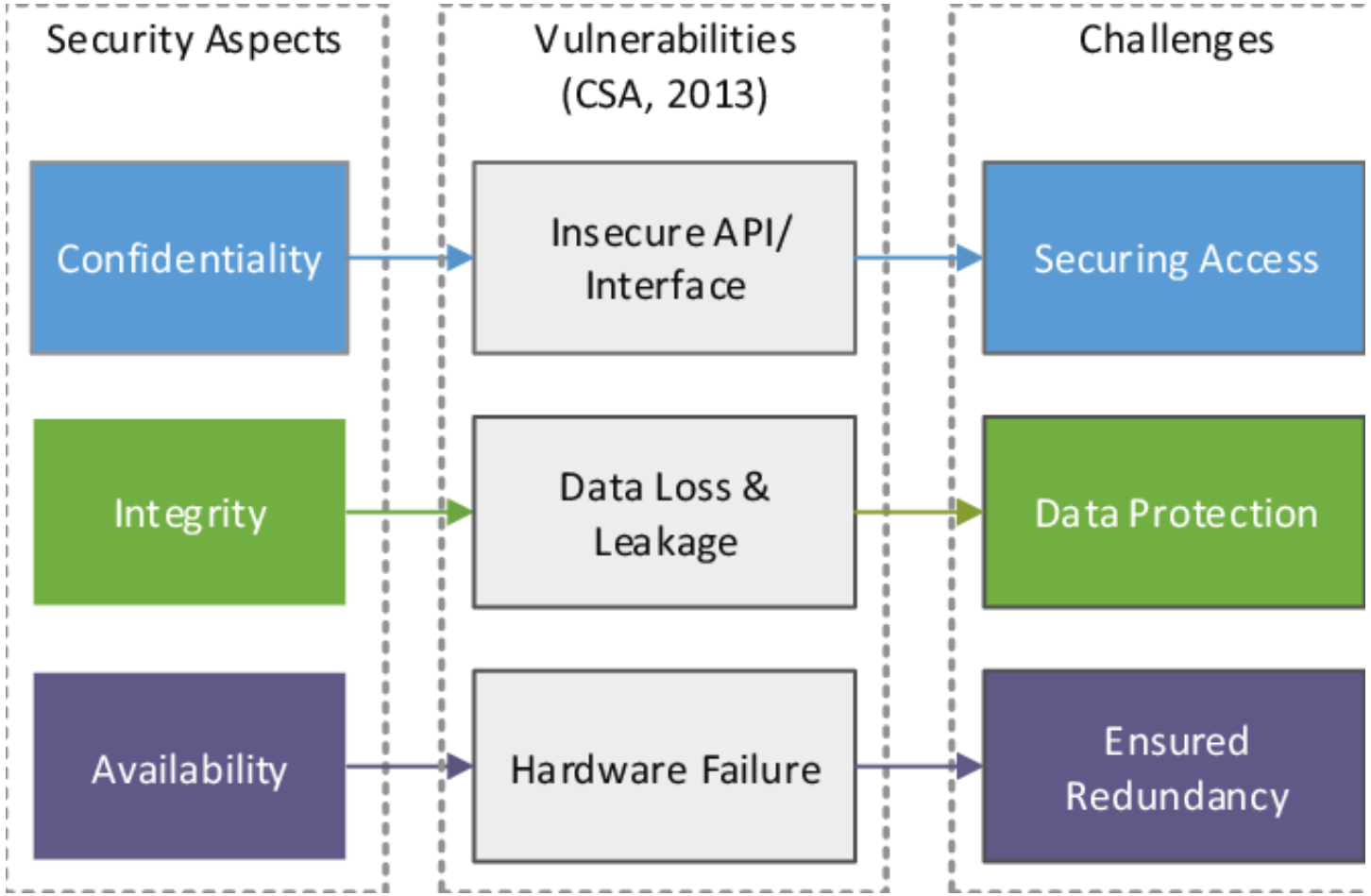
With cloud storage services, you can:

- Cost-effectively protect data in the cloud without sacrificing performance.
- Scale up your backup resources in minutes as data requirements change.
- Protect backups with a data center and network architecture built for security-sensitive organizations.

Generic cloud storage architecture



Cloud Storage Security Aspects, Vulnerabilities & Challenges



How does cloud storage work?

- Cloud storage is delivered by a cloud services provider that owns and operates data storage capacity by maintaining large datacenters in multiple locations around the world.
- Cloud storage providers manage capacity, security, and durability to make data accessible to your applications over the internet in a pay-as-you-go model.
- Typically, you connect to the storage cloud either through the internet or through a dedicated private connection, using a web portal, website, or a mobile app.
- When customers purchase cloud storage from a service provider, they turn over most aspects of the data storage to the vendor, including capacity, security, data availability, storage servers and computing resources, and network data delivery.
- Your applications access cloud storage through traditional storage protocols or directly using an application programming interface (API).
- The cloud storage provider might also offer services designed to help collect, manage, secure, and analyze data at a massive scale.

What are the types of cloud storage?

There are three main cloud storage types:

1. **object storage,**
2. **file storage,**
3. **block storage**

Object storage

- Organizations have to store a massive and growing amount of unstructured data, such as photos, videos, machine learning (ML), sensor data, audio files, and other types of web content, and finding scalable, efficient, and affordable ways to store them can be a challenge.
- Object storage is a data storage architecture for large stores of unstructured data.
- Objects store data in the format it arrives in and makes it possible to customize metadata in ways that make the data easier to access and analyze.
- Instead of being organized in files or folder hierarchies, objects are kept in secure buckets that deliver virtually unlimited scalability. It is also less costly to store large data volumes. Applications developed in the cloud often take advantage of the vast scalability and metadata characteristics of object storage.
- Object storage solutions are ideal for building modern applications from scratch that require scale and flexibility, and can also be used to import existing data stores for analytics, backup, or archive.

File storage

- File-based storage or file storage is widely used among applications and stores data in a hierarchical folder and file format.
- This type of storage is often known as a network-attached storage (NAS) server with common file level protocols of Server Message Block (SMB) used in Windows instances and Network File System (NFS) found in Linux.

Block storage

- Enterprise applications like databases or enterprise resource planning (ERP) systems often require dedicated, low-latency storage for each host.
- This is analogous to direct-attached storage (DAS) or a storage area network (SAN). In this case, you can use a cloud storage service that stores data in the form of blocks. Each block has its own unique identifier for quick storage and retrieval.

What are cloud storage use cases?

- Cloud storage has several use cases in application management, data management, and business continuity. Let's consider some examples below.

Analytics and data lakes

- Traditional on-premises storage solutions can be inconsistent in their cost, performance, and scalability — especially over time.
- Analytics demand large-scale, affordable, highly available, and secure storage pools that are commonly referred to as data lakes.
- Data lakes built on object storage keep information in its native form and include rich metadata that allows selective extraction and use for analysis.
- Cloud-based data lakes can sit at the center of multiple kinds of data warehousing and processing, as well as big data and analytical engines, to help you accomplish your next project in less time and with more targeted relevance.

Cloud storage use cases (cont..)

Backup and disaster recovery

- Backup and disaster recovery are critical for data protection and accessibility, but keeping up with increasing capacity requirements can be a constant challenge.
- Cloud storage brings low cost, high durability, and extreme scale to data backup and recovery solutions.
- Embedded data management policies can automatically migrate data to lower-cost storage based on frequency or timing settings, and archival vaults can be created to help comply with legal or regulatory requirements.
- These benefits allow for tremendous scale possibilities within industries such as financial services, healthcare and life sciences, and media and entertainment that produce high volumes of unstructured data with long-term retention needs.

Cloud storage use cases (cont..)

Software test and development

- Software test and development environments often require separate, independent, and duplicate storage environments to be built out, managed, and decommissioned.
- In addition to the time required, the up-front capital costs required can be extensive.
- Many of the largest and most valuable companies in the world create applications in record time by using the flexibility, performance, and low cost of cloud storage.
- Even the simplest static websites can be improved at low cost.
- IT professionals and developers are turning to pay-as-you-go storage options that remove management and scale headaches.

Cloud storage use cases (cont..)

Cloud data migration

- The availability, durability, and low cloud storage costs can be very compelling. On the other hand, IT personnel working with storage, backup, networking, security, and compliance administrators might have concerns about the realities of transferring large amounts of data to the cloud.
- For some, getting data into the cloud can be a challenge.
- Hybrid, edge, and data movement services meet you where you are in the physical world to help ease your data transfer to the cloud.

Cloud storage use cases (cont..)

Archive

- Enterprises today face significant challenges with exponential data growth. Machine learning (ML) and analytics give data more uses than ever before.
- Regulatory compliance requires long retention periods
- Customers need to replace on-premises tape and disk archive infrastructure with solutions that provide enhanced data durability, immediate retrieval times, better security and compliance, and greater data accessibility for advanced analytics and business intelligence.

Hybrid cloud storage

- Many organizations want to take advantage of the benefits of cloud storage, but have applications running on premises that require low-latency access to their data, or need rapid data transfer to the cloud.
- Hybrid cloud storage architectures connect your on-premises applications and systems to cloud storage to help you reduce costs, minimize management burden, and innovate with your data.

Cloud storage use cases (cont..)

- **Database storage**
- Because block storage has high performance and is readily updatable, many organizations use it for transactional databases.
- **With its limited metadata, block storage is able to deliver the ultra-low latency required for high-performance workloads and latency sensitive applications like databases**.
- Block storage allows developers to set up a robust, scalable, and highly efficient transactional database.
- As each block is a self-contained unit, the database performs optimally, even when the stored data grows

Structured Vs Unstructured Storage in Cloud

For more details refer to: IBM Cloud Databases , IBM Blog, Google Cloud Databases

Structured Data Storage

- **Structured data-** typically categorized as quantitative data — is highly organized and easily decipherable by machine learning algorithms. Developed by IBM in 1974, structured query language (SQL) is the programming language used to manage structured data. By using a relational (SQL) database, business users can quickly input, search and manipulate structured data.
- **Pros and cons of structured data**
- Examples of structured data include dates, names, addresses, credit card numbers, etc. Their benefits are tied to ease of use and access, while liabilities revolve around data inflexibility:
- **Pros**
  - **Easily used by machine learning (ML) algorithms:** The specific and organized architecture of structured data eases manipulation and querying of ML data.
  - **Easily used by business users:** Structured data does not require an in-depth understanding of different types of data and how they function. With a basic understanding of the topic relative to the data, users can easily access and interpret the data.
  - **Accessible by more tools:** Since structured data predates unstructured data, there are more tools available for using and analyzing structured data.
- **Cons**
  - **Limited usage:** Data with a predefined structure can only be used for its intended purpose, which limits its flexibility and usability.
  - **Limited storage options:** Structured data is generally stored in data storage systems with rigid schemas (e.g., “data warehouses”). Therefore, changes in data requirements necessitate an update of all structured data, which leads to a massive expenditure of time and resources.

Structured Data Tools & Use Cases

- **Structured Data Tools**
  - **OLAP:** Performs high-speed, multidimensional data analysis from unified, centralized data stores.
  - **SQLite:** Implements a self-contained, serverless, zero-configuration, transactional relational database engine.
  - **MySQL:** Embeds data into mass-deployed software, particularly mission-critical, heavy-load production system.
  - **PostgreSQL:** Supports SQL and JSON querying as well as high-tier programming languages (C/C+, Java, Python, etc.).

- **Use cases for structured data**
  - **Customer relationship management (CRM):** CRM software runs structured data through analytical tools to create datasets that reveal customer behavior patterns and trends.
  - **Online booking:** Hotel and ticket reservation data (e.g., dates, prices, destinations, etc.) fits the “rows and columns” format indicative of the pre-defined data model.
  - **Accounting:** Accounting firms or departments use structured data to process and record financial transactions.

Unstructured Data Storage

- **Unstructured data-** typically categorized as qualitative data, cannot be processed and analyzed via conventional data tools and methods. Since unstructured data does not have a predefined data model, it is best managed in non-relational (NoSQL) databases. Another way to manage unstructured data is to use data lakes to preserve it in raw form.
- The importance of unstructured data is rapidly increasing. Recent projections indicate that unstructured data is over 80% of all enterprise data, while 95% of businesses prioritize unstructured data management..
- **Pros and cons of unstructured data**
- Examples of unstructured data include text, mobile activity, social media posts, Internet of Things (IoT) sensor data, etc. Their benefits involve advantages in format, speed and storage, while liabilities revolve around expertise and available resources:
- **Pros**
  - **Native format:** Unstructured data, stored in its native format, remains undefined until needed. Its adaptability increases file formats in the database, which widens the data pool and enables data scientists to prepare and analyze only the data they need.
  - **Fast accumulation rates:** Since there is no need to predefine the data, it can be collected quickly and easily.
  - **Data lake storage:** Allows for massive storage and pay-as-you-use pricing, which cuts costs and eases scalability.

Unstructured Data Storage (cont..)

- **Cons**
  - **Requires expertise:** Due to its undefined/non-formatted nature, data science expertise is required to prepare and analyze unstructured data. This is beneficial to data analysts but alienates unspecialized business users who may not fully understand specialized data topics or how to utilize their data.
  - **Specialized tools:** Specialized tools are required to manipulate unstructured data, which limits product choices for data managers.

Unstructured Data Tools & Use Cases

- **Unstructured Data Tools**
  - **MongoDB:** Uses flexible documents to process data for cross-platform applications and services.
  - **DynamoDB:** Delivers single-digit millisecond performance at any scale via built-in security, in-memory caching and backup and restore.
  - **Hadoop:** Provides distributed processing of large data sets using simple programming models and no formatting requirements.
  - **Azure:** Enables agile cloud computing for creating and managing apps through Microsoft's data centers.

- **Use cases for unstructured data**
  - **Data mining:** Enables businesses to use unstructured data to identify consumer behavior, product sentiment, and purchasing patterns to better accommodate their customer base.
  - **Predictive data analytics:** Alert businesses of important activity ahead of time so they can properly plan and accordingly adjust to significant market shifts.
  - **Chatbots:** Perform text analysis to route customer questions to the appropriate answer sources.

Structured V/s Unstructured Data

- While structured (quantitative) data gives a “birds-eye view” of customers, unstructured (qualitative) data provides a deeper understanding of customer behavior and intent. Let’s explore some of the key areas of difference and their implications:
  - **Sources:** Structured data is sourced from GPS sensors, online forms, network logs, web server logs, OLTP systems, etc., whereas unstructured data sources include email messages, word-processing documents, PDF files, etc.
  - **Forms:** Structured data consists of numbers and values, whereas unstructured data consists of sensors, text files, audio and video files, etc.
  - **Models:** Structured data has a predefined data model and is formatted to a set data structure before being placed in data storage (e.g., schema-on-write), whereas unstructured data is stored in its native format and not processed until it is used (e.g., schema-on-read).
  - **Storage:** Structured data is stored in tabular formats (e.g., excel sheets or SQL databases) that require less storage space. It can be stored in data warehouses, which makes it highly scalable. Unstructured data, on the other hand, is stored as media files or NoSQL databases, which require more space. It can be stored in data lakes which makes it difficult to scale.
  - **Uses:** Structured data is used in machine learning (ML) and drives its algorithms, whereas unstructured data is used in natural language processing (NLP) and text mining.

SQL

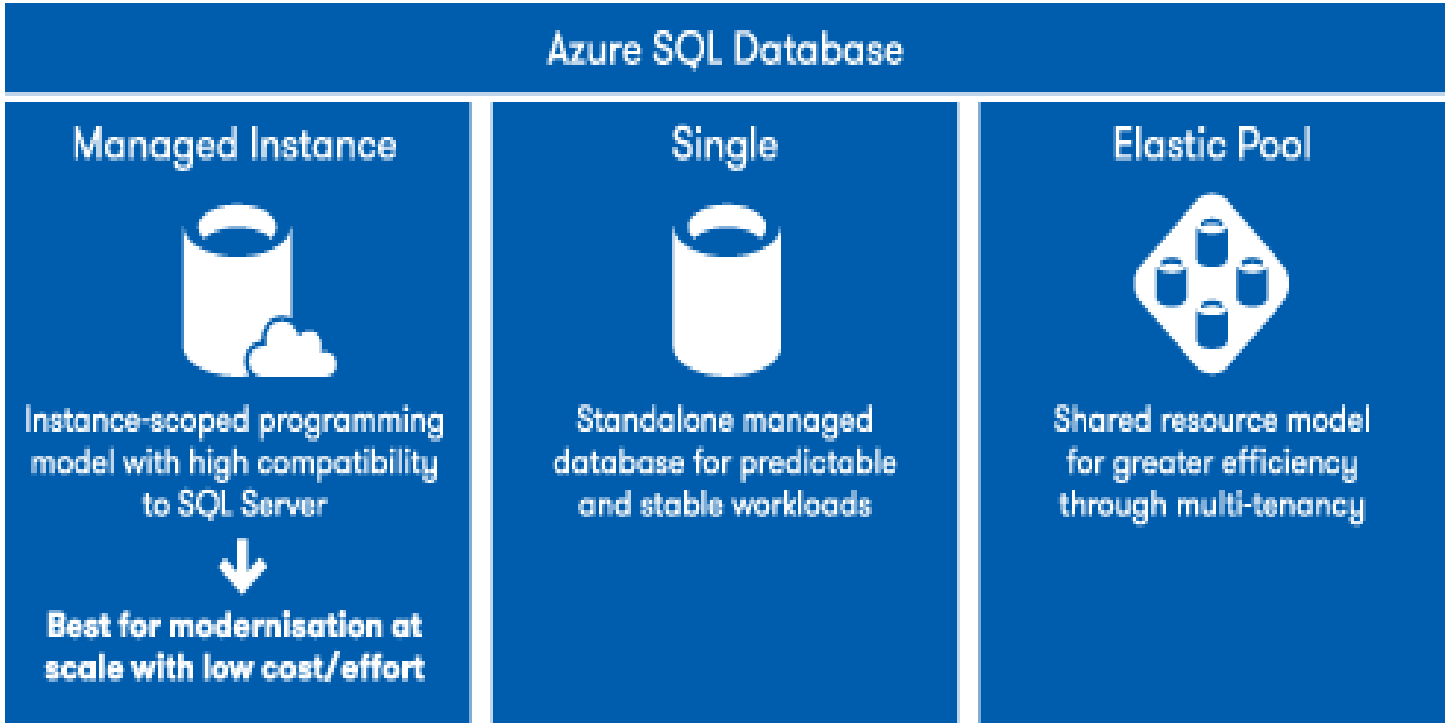
Microsoft SQL Server

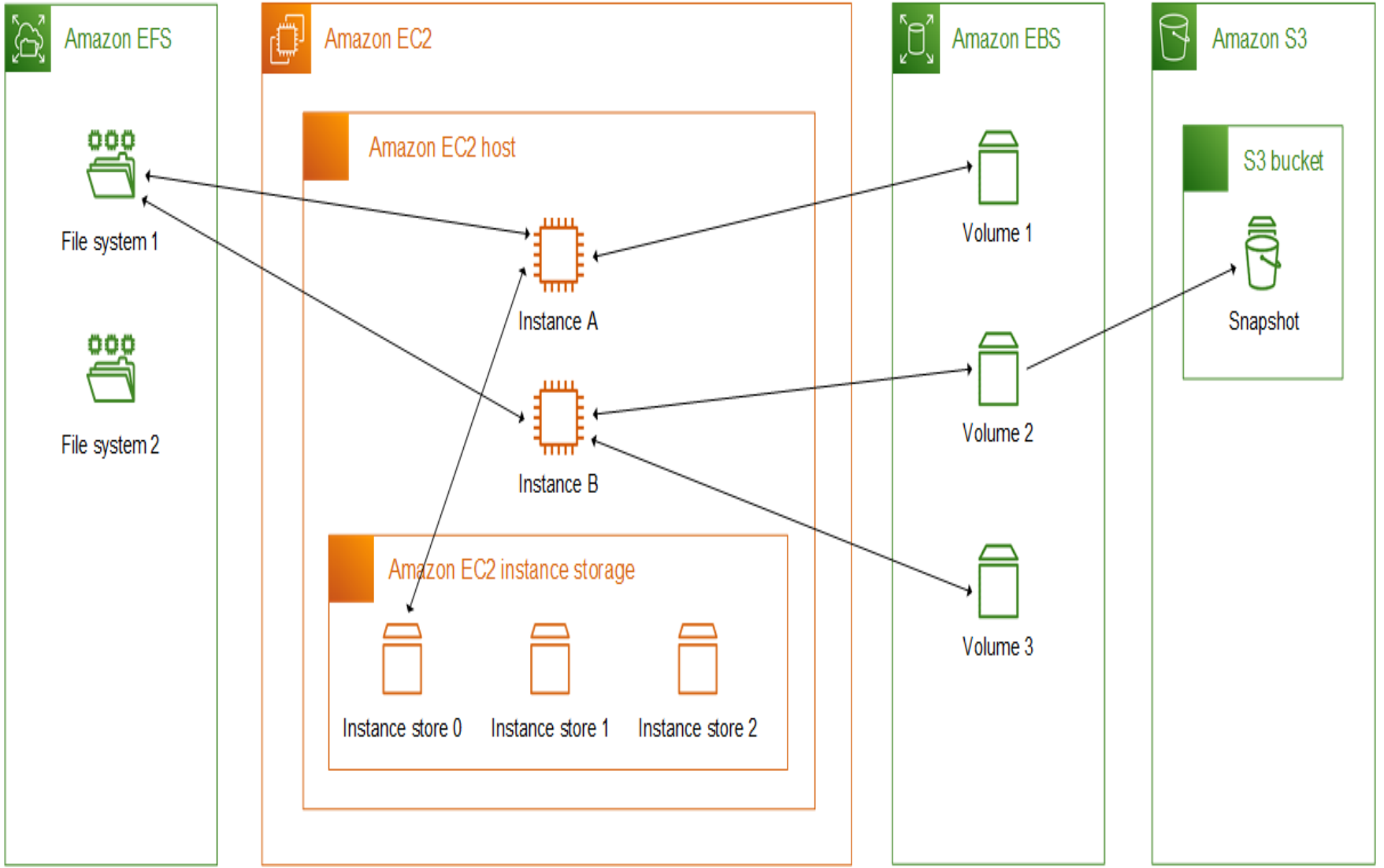
- **Microsoft SQL Server Master Data Services (MDS)** is a relational database management system developed by Microsoft.
- As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications—which may run either on the same computer or on another computer across a network (including the Internet)
- Microsoft SQL Server 2016 introduced enhancements to Master Data Services, such as improved performance and security, and the ability to clear transaction logs, create custom indexes, share entity data between different models, and support for many-to-many relationships.

Terminology

- *Model* is the highest level of an MDS instance. It is the primary container for specific groupings of master data. In many ways it is very similar to the idea of a database.
- *Entities* are containers created within a model. Entities provide a home for members, and are in many ways analogous to database tables. (e.g. Customer)
- *Members* are analogous to the records in a database table (Entity) e.g. Will Smith. Members are contained within entities. Each member is made up of two or more attributes.
- *Attributes* are analogous to the columns within a database table (Entity) e.g. Surname. Attributes exist within entities and help describe members (the records within the table). Name and Code attributes are created by default for each entity and serve to describe and uniquely identify leaf members. Attributes can be related to other attributes from other entities which are called 'domain-based' attributes. This is similar to the concept of a foreign key.

Microsoft SQL Server -SQL Managed Instance




Amazon EFS
Amazon EC2
Amazon EBS
Amazon S3
Amazon EC2 host
S3 bucket
File system 1
File system 2
Instance A
Instance B
Volume 1
Volume 2
Volume 3
Snapshot
Amazon EC2 instance storage
Instance store 0
Instance store 1
Instance store 2

Cloud Spanner

For more details refer to:https: //cloud.google.com/spanner/

Cloud Spanner

- Fully managed relational database with unlimited scale, strong consistency, and up to 99.999% availability.

- **Key Features:**

1. Relational database, built for scale:
   - Everything you would expect from a relational database—schemas, SQL queries, and ACID transactions—battle-tested and ready to scale for both reads and writes globally.

2. 99.999% availability:
   - Industry-leading high availability (up to 99.999%) for multi-regional instances with TrueTime atomic clocks and transparent, synchronous replication. 100% online schema changes and maintenance while serving traffic with zero downtime.

3. Automatic database sharding:
   - Optimize performance by automatically sharding the data based on request load and data size. As a result, you can scale your database without disruptive re-architecture, and focus on growing your business.

4. Fully managed:
   - Easy deployment at every stage and for any size database. Synchronous replication and maintenance are automatic and built in.

Cloud Spanner Key Features (cont..)

5. Strong transactional consistency:
   - Purpose-built for industry-leading external consistency without compromising on scalability or availability.

6. Granular instance sizing:
   - Start with Spanner with a granular instance for only $65/month and scale it based on your needs without downtime and with no need for re-architecting.

7. PostgreSQL interface:
   - Combine the scalability and reliability of Spanner with the familiarity and portability of PostgreSQL. Use the skills and tools that your teams already know, future-proofing your investment for peace of mind.

8. Regional and multi-regional configurations:
   - No matter where your users may be, apps backed by Spanner can read and write up-to-date strongly consistent data globally. Additionally, when running a multi-region instance, your database is protected against a regional failure and offers industry-leading 99.999% availability.

9. Unified analytics and AI on transactional data:
   - Query data in Spanner from BigQuery in real time without moving or copying the data, bridging the gap between operational data and analytics and creating a unified data life cycle. Invoke Vertex AI models in transactions in Spanner using a simple SQL query.

Cloud Spanner Key Features (cont..)

10. Built on Google Cloud network:
    - Cloud Spanner is built on Google's dedicated network that provides low-latency, security, and reliability for serving users across the globe.

11. Enterprise-grade security and controls:
    - Customer-managed encryption keys (CMEK), data-layer encryption, IAM integration for access and controls, and comprehensive audit logging. Support for VPC-SC, Access Transparency and Access Approval. Fine-grained access control lets you authorize access to Spanner data at the table and column level.

12. Backup and Restore, point-in-time recovery (PITR):
    - Backup your database to store a consistent copy of data and restore on demand. PITR provides continuous data protection with the ability to recover your past data to a microsecond granularity.

13. Rich application and tool support:
    - Meet development teams where they are with native client libraries for Java/JDBC, Go, Python, C#, Node.js, PHP, Ruby and C++ as well as the most popular ORMs, including Hibernate and Entity Framework.

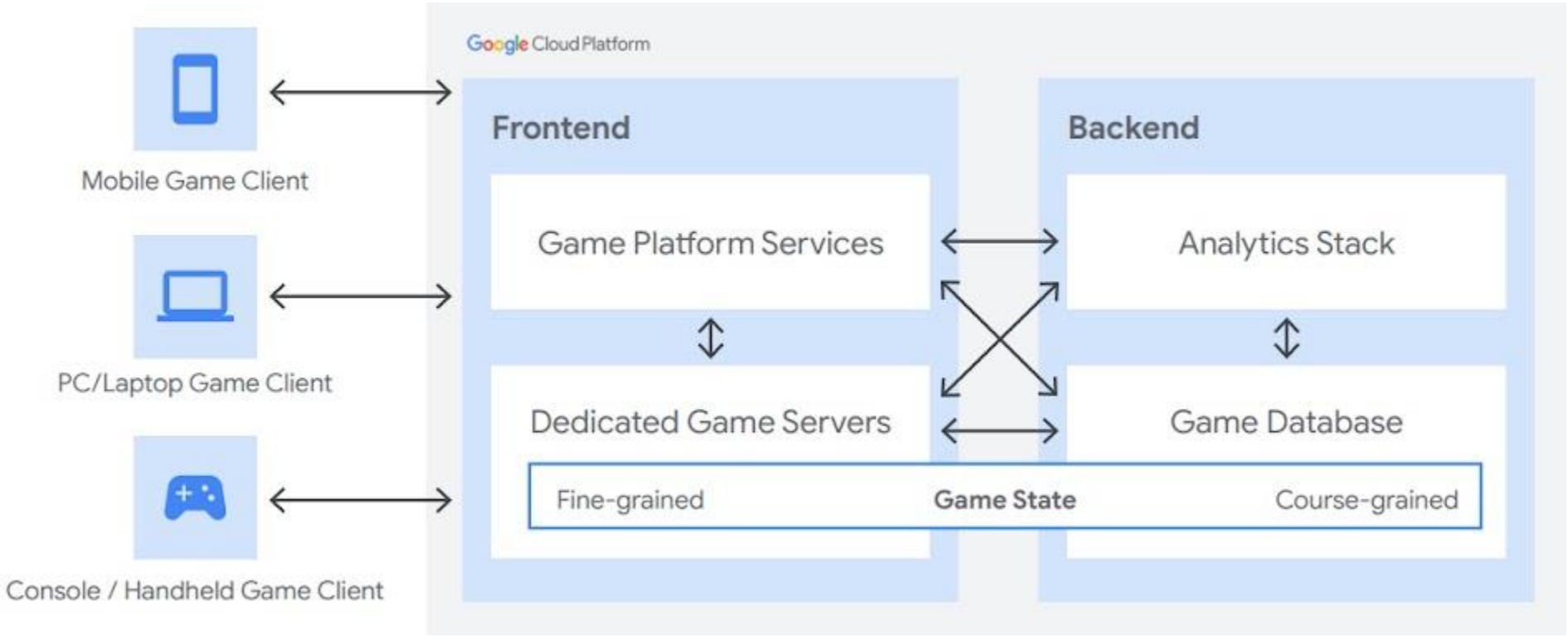
Cloud Spanner Key Features (cont..)

14. Real-time change data capture and replication:
    - Use Datastream to deliver change data from Oracle and MySQL databases into Spanner for up-to-date information. Use Spanner change streams to capture change data from Spanner databases and integrate it with other systems for analytics, event triggering, and compliance.

15. Observability:
    - Monitor performance of Spanner databases with metrics and stats. Analyze usage patterns in Spanner databases with Key Visualizer, an interactive monitoring tool. Use Query Insights for visualizing query performance metrics and debugging issues in the Google Cloud console.

Cloud Spanner Use case

- Develop global multiplayer games with Spanner:
- Spanner is a distributed, globally scalable SQL database service that decouples compute from storage, which makes it possible to scale processing resources separately from storage.
- This distributed scaling nature of Spanner's architecture makes it an ideal solution for unpredictable workloads such as online games.



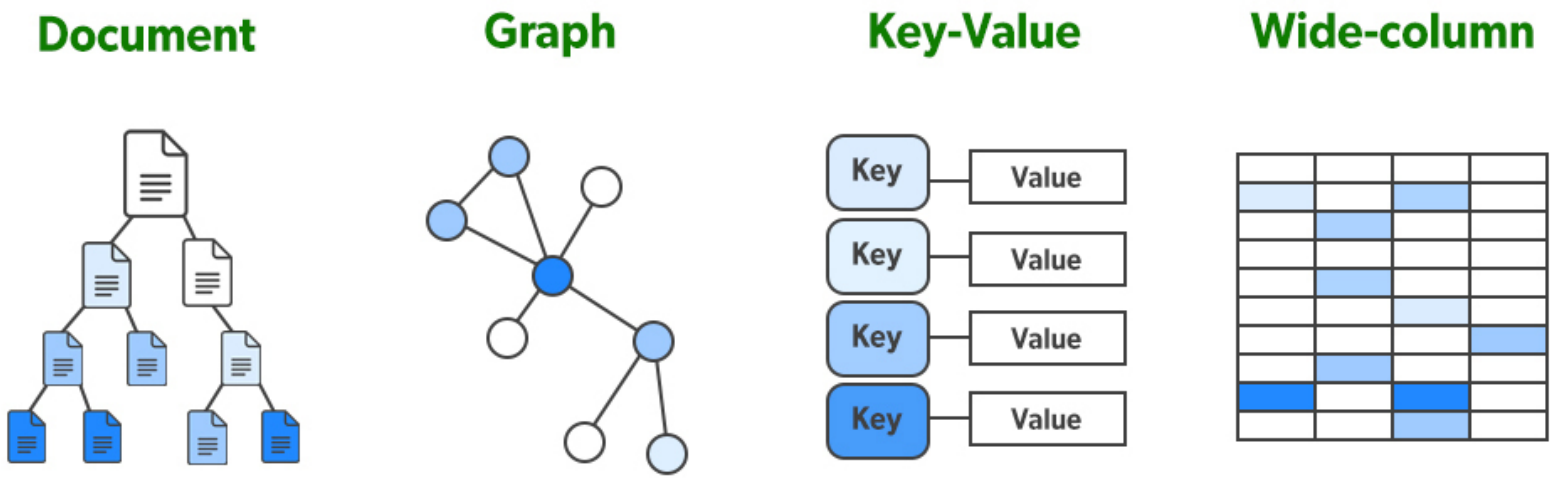
NoSQL

NoSQL Cloud Storage Services

- NoSQL Cloud Database Services are cloud-based database services that provide scalable, high-performance, and cost-effective solutions for storing and retrieving data.
- NoSQL (Not Only SQL) databases are designed to handle large volumes of unstructured, semi-structured, and structured data, and can easily scale horizontally to accommodate increased data volumes.
- **Cloud-based NoSQL databases offer several advantages** over traditional on-premise databases. These include:
  - Scalability: Cloud-based NoSQL databases can easily scale horizontally by adding more servers to the cluster. This allows for seamless scalability as data volumes increase.
  - High availability: NoSQL cloud databases are designed to be highly available and can provide reliable uptime and performance, which is critical for many applications.
  - Reduced cost: Cloud-based NoSQL databases can be more cost-effective than traditional on-premise databases because they eliminate the need for expensive hardware and infrastructure. This can be particularly beneficial for small to medium-sized businesses that do not have the resources to invest in expensive hardware.
  - Improved performance: Cloud-based NoSQL databases can provide high performance and low latency, making them well-suited for applications that require fast and efficient data access.
  - Flexibility: Cloud-based NoSQL databases are designed to handle unstructured, semi-structured, and structured data, making them a flexible solution for a wide range of applications.

Types of NoSQL databases

1. Column-based
2. Key-value store
3. Graph databases
4. Document-Based



Popular NoSQL Cloud Services

1. Amazon DynamoDB: A fully managed NoSQL database service offered by Amazon Web Services (AWS) that provides fast and predictable performance with seamless scalability.
2. Google Cloud Datastore: A NoSQL document database service that is fully managed and offers automatic scaling, high availability, and low latency.
3. Microsoft Azure Cosmos DB: A globally distributed, multi-model database service that provides high availability, low latency, and flexible data modeling.
4. MongoDB Atlas: A fully managed global cloud database service for MongoDB that provides automated backups, advanced security, and easy scalability.

- Overall, NoSQL Cloud Database Services provide a flexible, scalable, and cost-effective solution for storing and retrieving data in the cloud.
- They offer several advantages over traditional on-premise databases and can be an excellent choice for businesses of all sizes that need to store and manage large volumes of data.

Google Cloud NoSQL

- Google's cloud platform (GCP) offers a wide variety of database services. Of these, its NoSQL database services are unique in their ability to rapidly process very large, dynamic datasets with no fixed schema.
- Google Cloud provides the following NoSQL database services:
  - **Cloud Firestore**—a document-oriented database storing key-value pairs. Optimized for small documents and easy to use with mobile applications.
  - **Cloud Datastore**—a document database built for automatic scaling, high performance, and ease of use.
  - **Cloud Bigtable**—an alternative to HBase, a columnar database system running on HDFS. Suitable for high throughput applications.
  - **MongoDB Atlas**—a managed MongoDB service, hosted by Google Cloud and built by the original makers of MongoDB.

Google Cloud Bigtable

For more details refer to: https://cloud.google.com/bigtable/docs/overview

Cloud Bigtable

- It is a sparsely populated table that can scale to billions of rows and thousands of columns, enabling you to store terabytes or even petabytes of data.
- It is HBase-compatible, enterprise-grade NoSQL database service with single-digit millisecond latency, limitless scale, and 99.999% availability for large analytical and operational workloads.
- A single value in each row is indexed; this value is known as the row key.
- Bigtable is ideal for storing large amounts of single-keyed data with low latency.
- It supports high read and write throughput at low latency, and it's an ideal data source for MapReduce operations.
- Bigtable is exposed to applications through multiple client libraries, including a supported extension to the Apache HBase library for Java. As a result, it integrates with the existing Apache ecosystem of open source big data software.
- Bigtable is ideal for applications that need high throughput and scalability for key/value data, where each value is typically no larger than 10 MB.
- Bigtable also excels as a storage engine for batch MapReduce operations, stream processing/analytics, and machine-learning applications.

Cloud Bigtable (cont..)

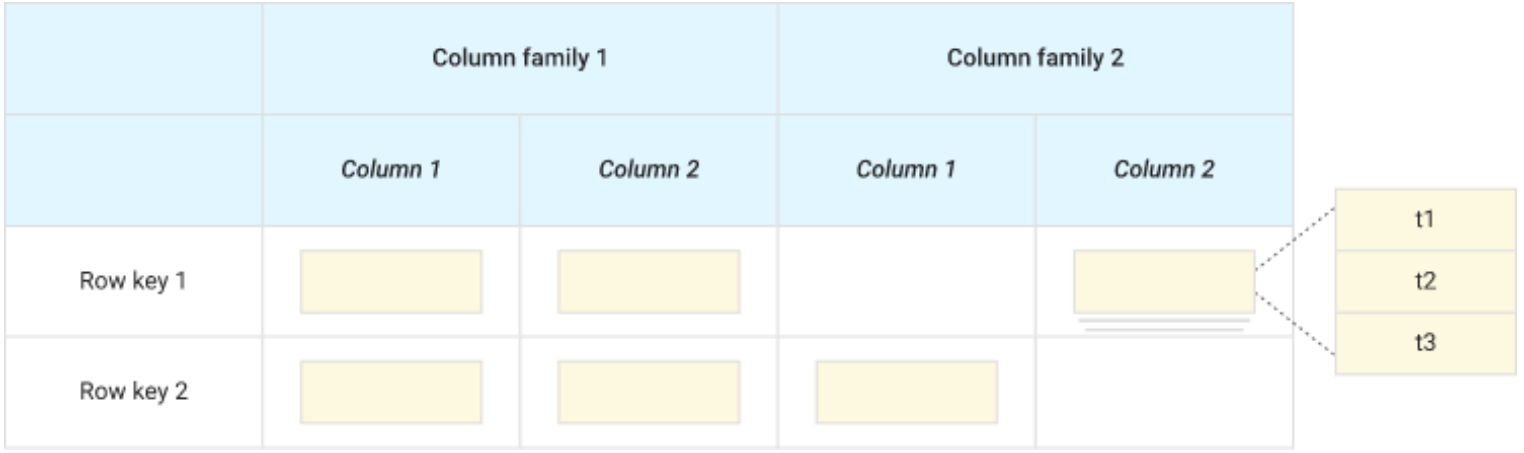
- Bigtable can **be used to store and query all of the following types of data**:
  - **Time-series data,** such as CPU and memory usage over time for multiple servers.
  - **Marketing data,** such as purchase histories and customer preferences.
  - **Financial data,** such as transaction histories, stock prices, and currency exchange rates.
  - **Internet of Things data,** such as usage reports from energy meters and home appliances.
  - **Graph data,** such as information about how users are connected to one another.

- Bigtable's powerful backend servers offer several **key advantages** over a self-managed HBase installation:
  - **Incredible scalability:** Bigtable scales in direct proportion to the number of machines in your cluster. A self-managed HBase installation has a design bottleneck that limits the performance after a certain threshold is reached. Bigtable does not have this bottleneck, so you can scale your cluster up to handle more reads and writes.
  - **Simple administration:** Bigtable handles upgrades and restarts transparently, and it automatically maintains high data durability. To replicate your data, simply add a second cluster to your instance, and replication starts automatically. No more managing replicas or regions; just design your table schemas, and Bigtable will handle the rest for you.
  - **Cluster resizing without downtime:** You can increase the size of a Bigtable cluster for a few hours to handle a large load, then reduce the size of the cluster again—all without any downtime. After you change a cluster's size, it typically takes just a few minutes under load for Bigtable to balance performance across all of the nodes in your cluster.

Cloud Bigtable Storage Model

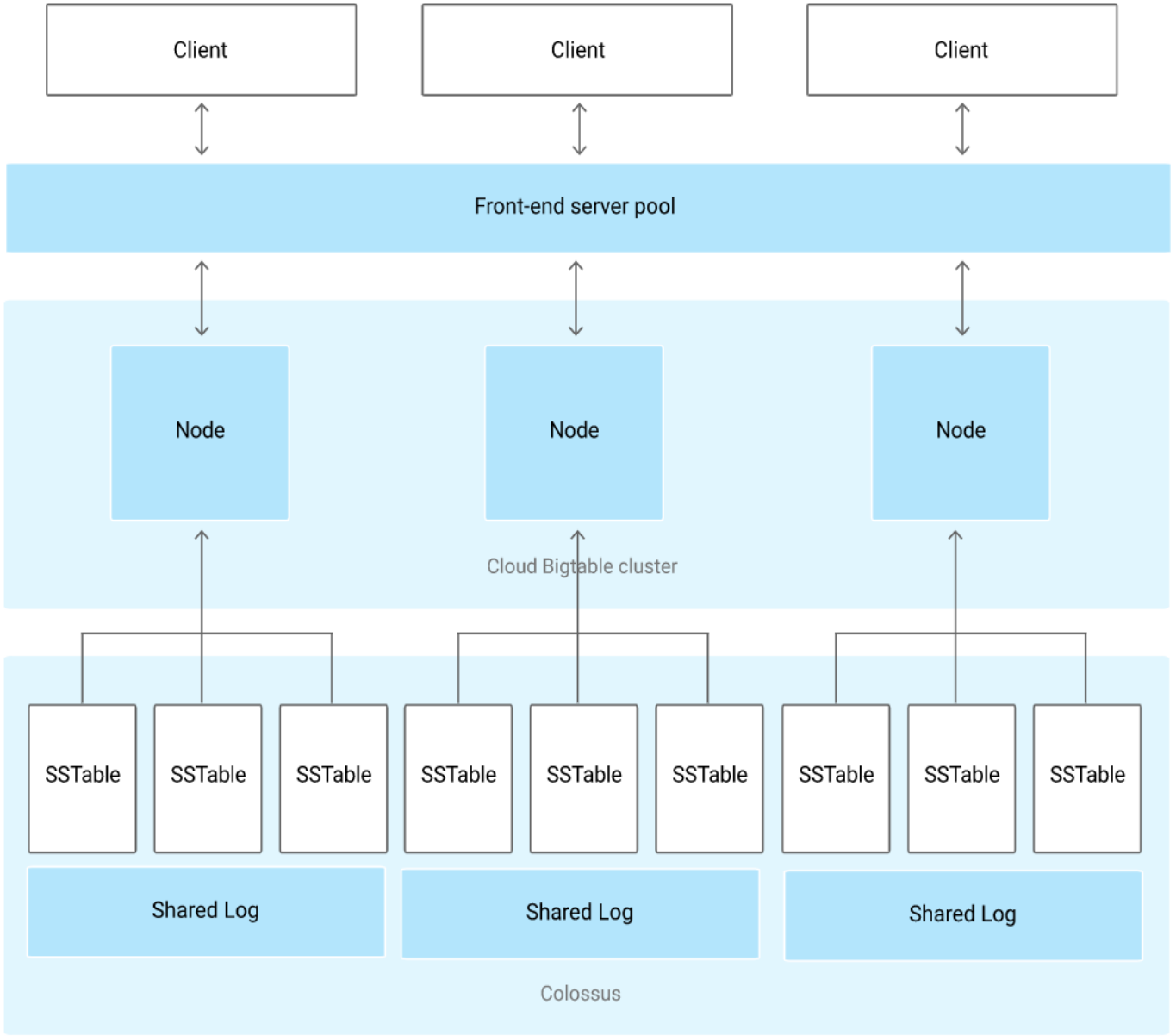
- Bigtable stores data in massively scalable tables, each of which is a sorted key/value map. The table is composed of rows, each of which typically describes a single entity, and columns, which contain individual values for each row.
- Each row is indexed by a single row key, and columns that are related to one another are typically grouped into a column family. Each column is identified by a combination of the column family and a column qualifier, which is a unique name within the column family.
- Each row/column intersection can contain multiple cells. Each cell contains a unique timestamped version of the data for that row and column.
- Storing multiple cells in a column provides a record of how the stored data for that row and column has changed over time. Bigtable tables are sparse; if a column is not used in a particular row, it does not take up any space.

- A few things to notice in this illustration:
  - Columns can be unused in a row.
  - Each cell in a given row and column has a unique timestamp (t).



Cloud Bigtable Architecture

- As the diagram illustrates, all client requests go through a frontend server ("tablet server") before they are sent to a Bigtable node.
- The nodes are organized into a Bigtable cluster, which belongs to a Bigtable instance, a container for the cluster.
- Each node in the cluster handles a subset of the requests to the cluster. By adding nodes to a cluster, you can increase the number of simultaneous requests that the cluster can handle. Adding nodes also increases the maximum throughput for the cluster.
- If you enable replication by adding additional clusters, you can also send different types of traffic to different clusters. Then if one cluster becomes unavailable, you can fail over to another cluster.



Cloud Bigtable Architecture (cont..)

- A Bigtable table is sharded into blocks of contiguous rows, called tablets, to help balance the workload of queries. (Tablets are similar to HBase regions.)
- Tablets are stored on Colossus, Google's file system, in SSTable format. An SSTable provides a persistent, ordered immutable map from keys to values, where both keys and values are arbitrary byte strings.
- Each tablet is associated with a specific Bigtable node. In addition to the SSTable files, all writes are stored in Colossus's shared log as soon as they are acknowledged by Bigtable, providing increased durability.
- Importantly, data is never stored in Bigtable nodes themselves; each node has pointers to a set of tablets that are stored on Colossus. As a result:
  - Rebalancing tablets from one node to another happens quickly, because the actual data is not copied. Bigtable simply updates the pointers for each node.
  - Recovery from the failure of a Bigtable node is fast, because only metadata must be migrated to the replacement node.
  - When a Bigtable node fails, no data is lost.

Cloud Bigtable Architecture (cont..)

- A Bigtable **instance** is a container for your data.
  - Instances have one or more clusters, located in different zones. Each cluster has at least 1 node.
  - A table belongs to an instance, not to a cluster or node.
  - If you have an instance with more than one cluster, you are using replication. This means you can't assign a table to an individual cluster or create unique garbage collection policies for each cluster in an instance.
  - You also can't make each cluster store a different set of data in the same table.
  - An instance has a few important properties that you need to know about:
    - The storage type (SSD or HDD)
    - The application profiles, which are primarily for instances that use replication
- **Clusters:**
  - A cluster represents the Bigtable service in a specific location. Each cluster belongs to a single Bigtable instance, and an instance can have clusters in up to 8 regions. When your application sends requests to a Bigtable instance, those requests are handled by one of the clusters in the instance.
  - Each cluster is located in a single zone. An instance can have clusters in up to 8 regions where Bigtable is available. Each zone in a region can contain only one cluster.

Cloud Bigtable Architecture (cont..)

- **Nodes:**
  - Each cluster in an instance has 1 or more nodes, which are compute resources that Bigtable uses to manage your data.
  - Behind the scenes, Bigtable splits all of the data in a table into separate tablets. Tablets are stored on disk, separate from the nodes but in the same zone as the nodes. A tablet is associated with a single node.
  - Each node is responsible for:
    - Keeping track of specific tablets on disk.
    - Handling incoming reads and writes for its tablets.
    - Performing maintenance tasks on its tablets, such as periodic compactions.

- A cluster must have enough nodes to support its current workload and the amount of data it stores. Otherwise, the cluster might not be able to handle incoming requests, and latency could go up. Monitor your clusters' CPU and disk usage, and add nodes to an instance when its metrics exceed the recommendations and limits listed below.

Cloud Bigtable Use Cases

1. Bigtable for Cassandra users:

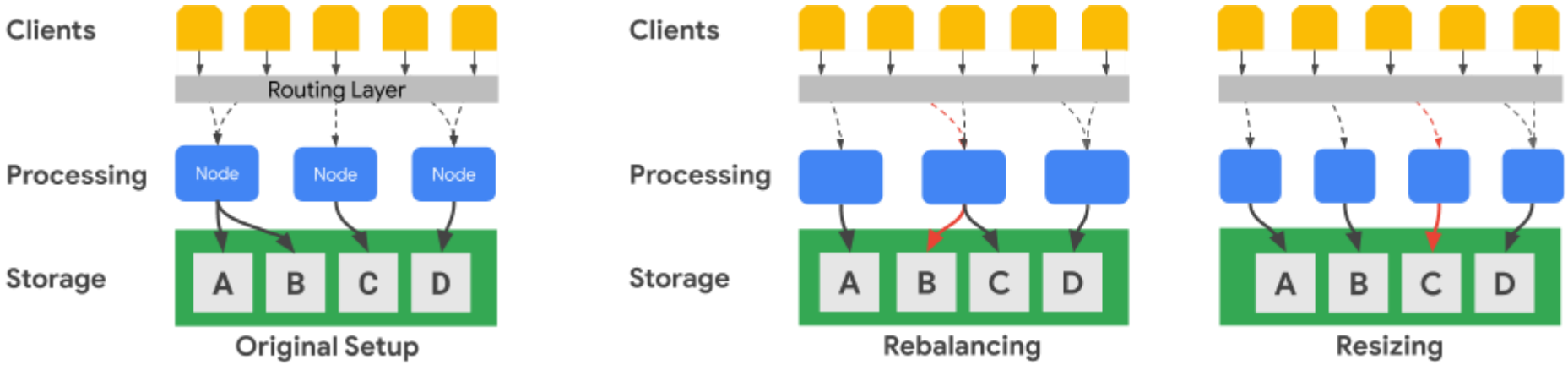
- Bigtable and Cassandra are distributed databases. They implement multidimensional key-value stores that can support tens of thousands of queries per second (QPS), storage that scales up to petabytes of data, and tolerance for node failure.
- Bigtable separates the compute nodes, which serve client requests, from the underlying storage management. Data is stored on Colossus.
- The storage layer automatically replicates the data to provide durability that exceeds levels provided by standard Hadoop Distributed File System (HDFS) three-way replication.
- This architecture provides consistent reads and writes within a cluster, scales up and down without any storage redistribution cost, and can rebalance workloads without modifying the cluster or schema.
- If any data processing node becomes impaired, the Bigtable service replaces it transparently.
- Bigtable also supports asynchronous replication between geographically distributed clusters in topologies of up to four clusters in different Google Cloud zones or regions throughout the world.
- In addition to gRPC and client libraries for various programming languages, Bigtable maintains compatibility with the open source Apache HBase Java client library.

- The following diagram shows how Bigtable physically separates the processing nodes from the storage layer:

Cloud Bigtable Use Cases (cont..)

1. Bigtable for Cassandra users (cont..)

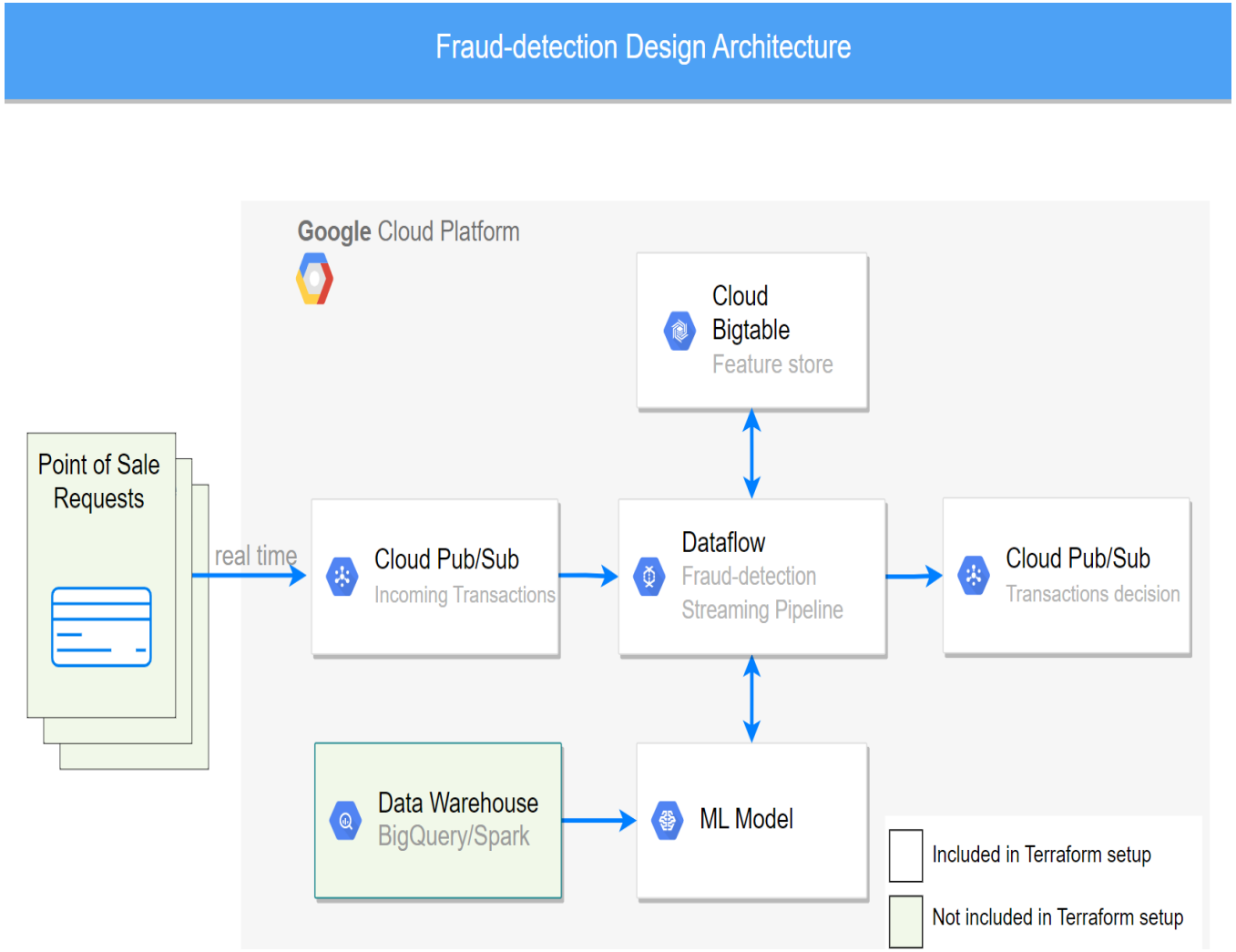
- The following diagram shows how Bigtable physically separates the processing nodes from the storage layer.
- In the preceding diagram, the middle processing node is only responsible for serving data requests for the C dataset in the storage layer.
- If Bigtable identifies that range-assignment rebalancing is required for a dataset, the data ranges for a processing node are straightforward to change because the storage layer is separated from the processing layer. The last two diagrams show, in simplified terms, a key range rebalancing and a cluster resizing.



Cloud Bigtable Use Cases

1. Credit card fraud detection using Cloud Bigtable

- This sample application aims to build a fast and scalable fraud detection system using Cloud Bigtable as its feature store.
- The feature store holds customer profiles (customer ids, addresses, etc.) and historical transactions.
- In order to determine if a transaction is fraudulent, the feature store queries the customer profile information and transaction history.
- Cloud Bigtable stores data in tables, each of which is a sorted key/value map.
- The table is composed of rows, each of which typically describes a single entity, and columns, which contain individual values for each row.
- Each row/column intersection can contain multiple cells. Each cell contains a unique timestamped version of the data for that row and column.



Cloud Bigtable Use Cases (cont..)

1. Credit card fraud detection using Cloud Bigtable (cont..)

- This design uses a single table to store all customers' information. The table is structured as follows:

row key	customer_profile column family	historical transactions column family
user_id 1	Customer’s profile information	Transaction details at time 10
		Transaction details at time 7
		Transaction details at time 4
		...
user_id 2	Customer’s profile information	Transaction details at time 8
		Transaction details at time 7
		...

- Row Key: The row key is the unique userID.
- Timestamps: Cloud Bigtable Native timestamps are used rather than putting the timestamp as the row key suffix.

Openstack Storage

NOVA, Neutron, Keystone Cinder, Swift and Glances, VMware Suit, Apache Cloud Stack

For more details refer to: OpenStack Storage

OpenStack

- The OpenStack project is an open source cloud computing platform that supports all types of cloud environments.
- The project aims for simple implementation, massive scalability, and a rich set of features. Cloud computing experts from around the world contribute to the project.

- OpenStack provides an Infrastructure-as-a-Service (IaaS) solution through a variety of complementary services. Each service offers an Application Programming Interface (API) that facilitates this integration.

- OpenStack has multiple storage realms to consider:
  - Block Storage (cinder)
  - Object Storage (swift)
  - Image storage (glance)
  - Ephemeral storage (nova)

Nova

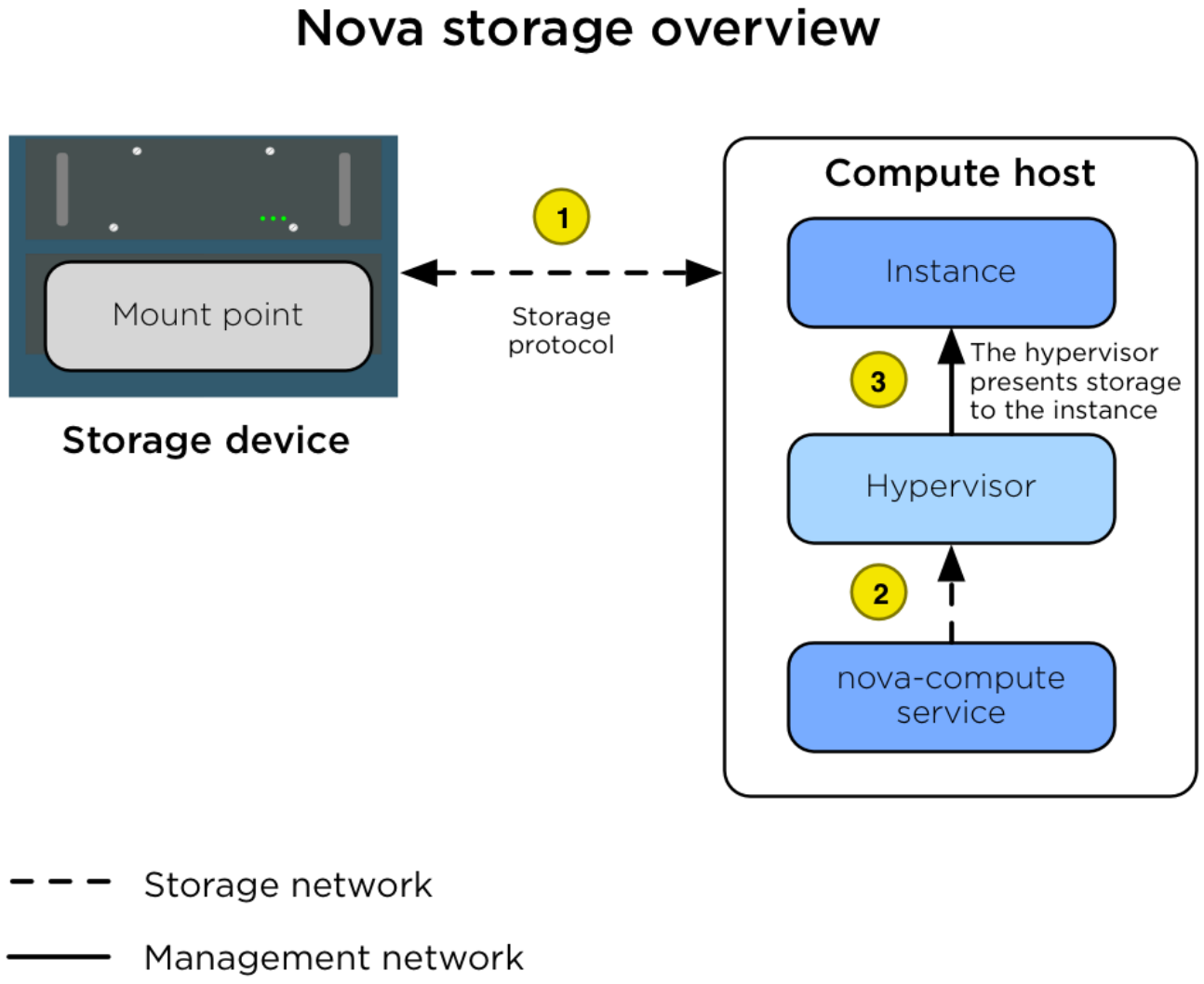
For more details refer to: Nova

Nova

- Nova is the OpenStack Ephemeral storage service.
- When the flavors in the Compute service are configured to provide instances with root or ephemeral disks, the nova-compute service manages these allocations using its ephemeral disk storage location.
- In many environments, the ephemeral disks are stored on the Compute host's local disks, but for production environments we recommend that the Compute hosts be configured to use a shared storage subsystem instead.
- A shared storage subsystem allows quick, live instance migration between Compute hosts, which is useful when the administrator needs to perform maintenance on the Compute host and wants to evacuate it.
- Using a shared storage subsystem also allows the recovery of instances when a Compute host goes offline. The administrator is able to evacuate the instance to another Compute host and boot it up again.

Nova (cont..)

- The Nova Storage Overview diagram illustrates the interactions between the storage device, the Compute host, the hypervisor, and the instance.
- The diagram shows the following steps.
    1. The Compute host is configured with access to the storage device. The Compute host accesses the storage space via the storage network (br-storage) by using a storage protocol (for example, NFS, iSCSI, or Ceph RBD).
    2. The nova-compute service configures the hypervisor to present the allocated instance disk as a device to the instance.
    3. The hypervisor presents the disk as a device to the instance.



Neutron

For more details refer to: Neutron

Neutron

- Neutron is an OpenStack project to provide "network connectivity as a service" between interface devices (e.g., vNICs) managed by other OpenStack services (e.g., nova). It implements the OpenStack Networking API.

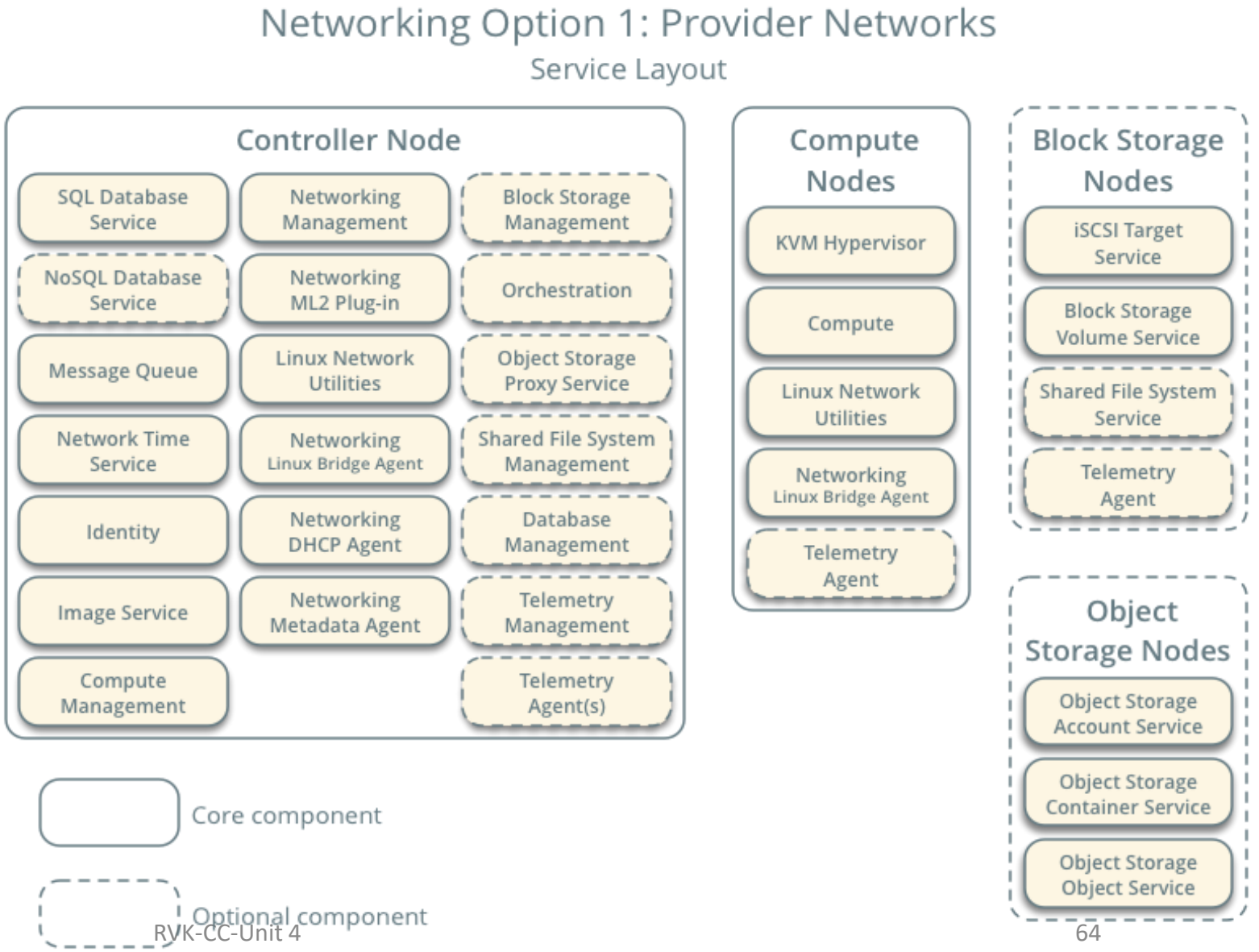
- It manages all networking facets for the Virtual Networking Infrastructure (VNI) and the access layer aspects of the Physical Networking Infrastructure (PNI) in your OpenStack environment. OpenStack Networking enables projects to create advanced virtual network topologies which may include services such as a firewall, a load balancer, and a virtual private network (VPN).

- Networking also supports security groups. Security groups enable administrators to define firewall rules in groups. A VM can belong to one or more security groups, and Networking applies the rules in those security groups to block or unblock ports, port ranges, or traffic types for that VM.

- Each plug-in that Networking uses has its own concepts. While not vital to operating the VNI and OpenStack environment, understanding these concepts can help you set up Networking. All Networking installations use a core plug-in and a security group plug-in (or just the No-Op security group plug-in). Additionally, Firewall-as-a-Service (FWaaS) and Load-Balancer-as-a-Service (LBaaS) plug-ins are available.

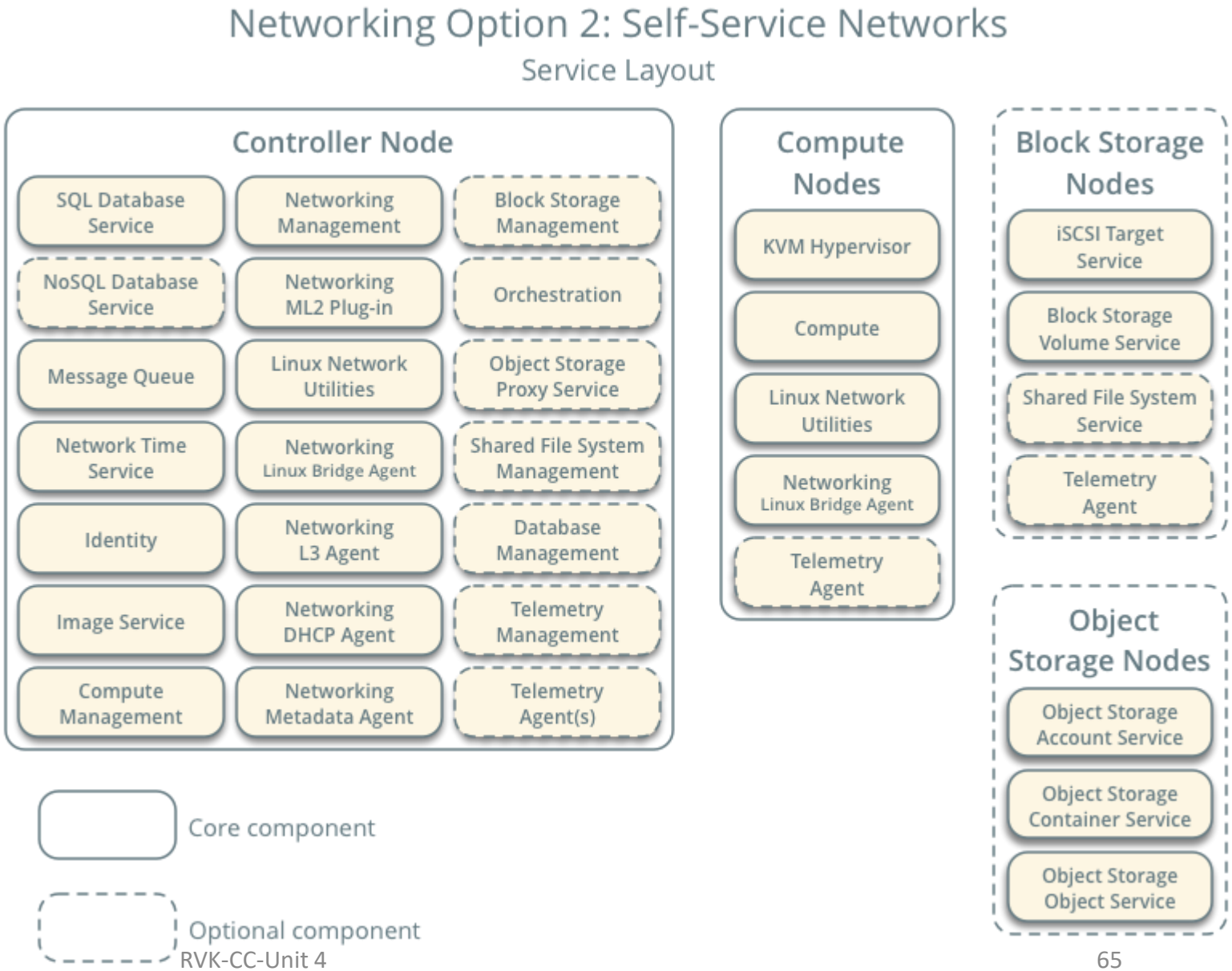
Networking Option 1: Provider networks

- The provider networks option deploys the OpenStack Networking service in the simplest way possible with primarily layer-2 (bridging/switching) services and VLAN segmentation of networks.
- Essentially, it bridges virtual networks to physical networks and relies on physical network infrastructure for layer-3 (routing) services.
- Additionally, a Dynamic Host Configuration Protocol (DHCP) service provides IP address information to instances.



Networking Option 2: Self-service networks

- The self-service networks option augments the provider networks option with layer-3 (routing) services that enable self-service networks using overlay segmentation methods such as Virtual Extensible LAN (VXLAN).
- Essentially, it routes virtual networks to physical networks using Network Address Translation (NAT).
- Additionally, this option provides the foundation for advanced services such as LBaaS and FWaaS.
- The OpenStack user can create virtual networks without the knowledge of underlying infrastructure on the data network.
- This can also include VLAN networks if the layer-2 plug-in is configured accordingly.



Keystone Cinder

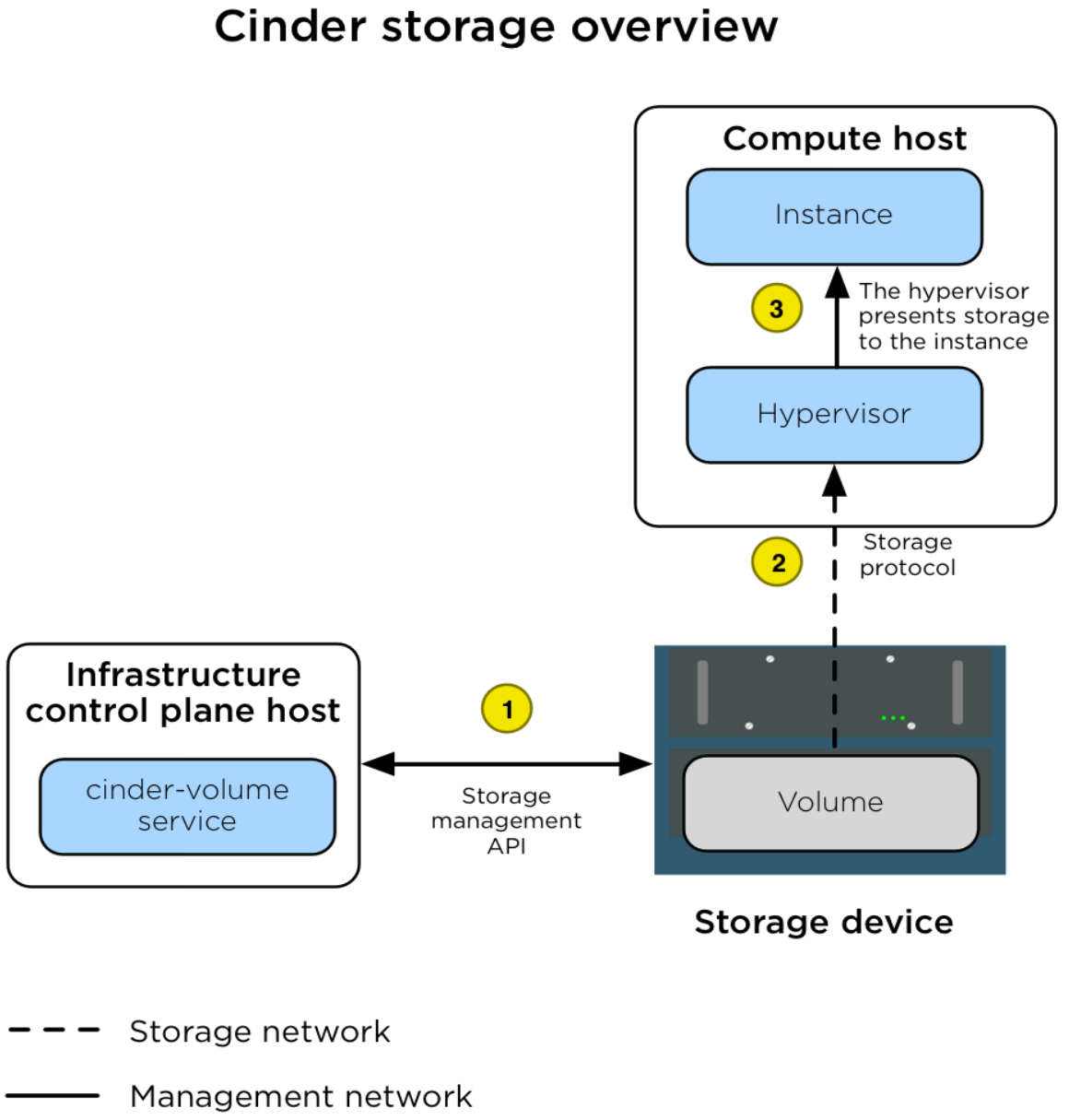
For more details refer to: Cinder

Cinder

- Cinder is the OpenStack Block Storage service for providing volumes to Nova virtual machines, Ironic bare metal hosts, containers and more. The Block Storage (cinder) service manages volumes on storage devices in an environment.
- Some of the goals of Cinder are to be/have:
  - Component based architecture: Quickly add new behaviors
  - Highly available: Scale to very serious workloads
  - Fault-Tolerant: Isolated processes avoid cascading failures
  - Recoverable: Failures should be easy to diagnose, debug, and rectify
  - Open Standards: Be a reference implementation for a community-driven api
- As an end user of Cinder, you'll use Cinder to create and manage volumes using the Horizon user interface, command line tools such as the python-cinderclient, or by directly using the REST API.

Cinder (cont..)

- In a production environment, the device presents storage via a storage protocol (for example, NFS, iSCSI, or Ceph RBD) to a storage network (br-storage) and a storage management API to the management network (br-mgmt).
- Instances are connected to the volumes via the storage network by the hypervisor on the Compute host.
- The Cinder Storage Overview diagram shows the following steps:
  1. A volume is created by the assigned cinder-volume service using the appropriate cinder driver. The volume is created by using an API that is presented to the management network.
  2. A volume is created by the assigned cinder-volume service using the appropriate cinder driver. The volume is created by using an API that is presented to the management network.
  3. After the hypervisor is connected to the volume, it presents the volume as a local hardware device to the instance.



Tools for using Cinder

- Horizon: The official web UI for the OpenStack Project.

- OpenStack Client: The official CLI for OpenStack Projects. You should use this as your CLI for most things, it includes not just nova commands but also commands for most of the projects in OpenStack.

- Cinder Client: The openstack CLI is recommended, but there are some advanced features and administrative commands that are not yet available there. For CLI access to these commands, the cinder CLI can be used instead.

- Using the Cinder API:
  - All features of Cinder are exposed via a REST API that can be used to build more complicated logic or automation with Cinder. This can be consumed directly or via various SDKs.

Swift

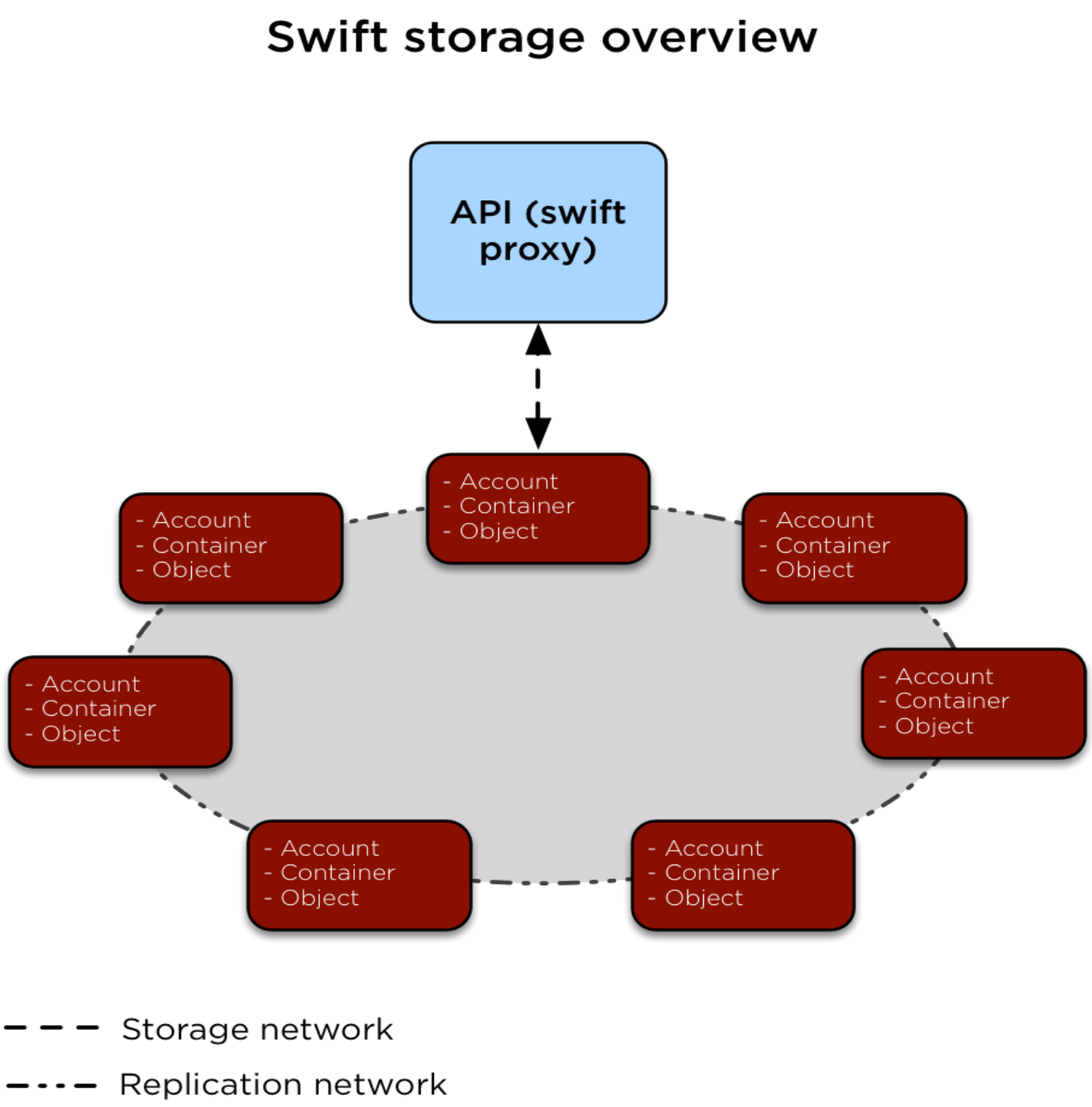
For more details refer to: Swift

Swift

- Swift is the OpenStack Object Storage service used for redundant, scalable data storage using clusters of standardized servers to store petabytes of accessible data.
- It implements a highly available, distributed, eventually consistent object/blob store that is accessible via HTTP/HTTPS.
- It is a long-term storage system for large amounts of static data which can be retrieved and updated. Object Storage uses a distributed architecture with no central point of control, providing greater scalability, redundancy, and permanence.
- Objects are written to multiple hardware devices, with the OpenStack software responsible for ensuring data replication and integrity across the cluster.
- Storage clusters scale horizontally by adding new nodes. Should a node fail, OpenStack works to replicate its content from other active nodes.
- Because OpenStack uses software logic to ensure data replication and distribution across different devices, inexpensive commodity hard drives and servers can be used in lieu of more expensive equipment.
- Object Storage is ideal for cost effective, scale-out storage. It provides a fully distributed, API-accessible storage platform that can be integrated directly into applications or used for backup, archiving, and data retention.

Swift

- The Swift Storage Overview diagram illustrates how data is accessed and replicated.
- The swift-proxy service is accessed by clients via the load balancer on the management network (br-mgmt).
- The swift-proxy service communicates with the Account, Container, and Object services on the Object Storage hosts via the storage network(br-storage).
- Replication between the Object Storage hosts is done via the replication network (br-repl).



Glance

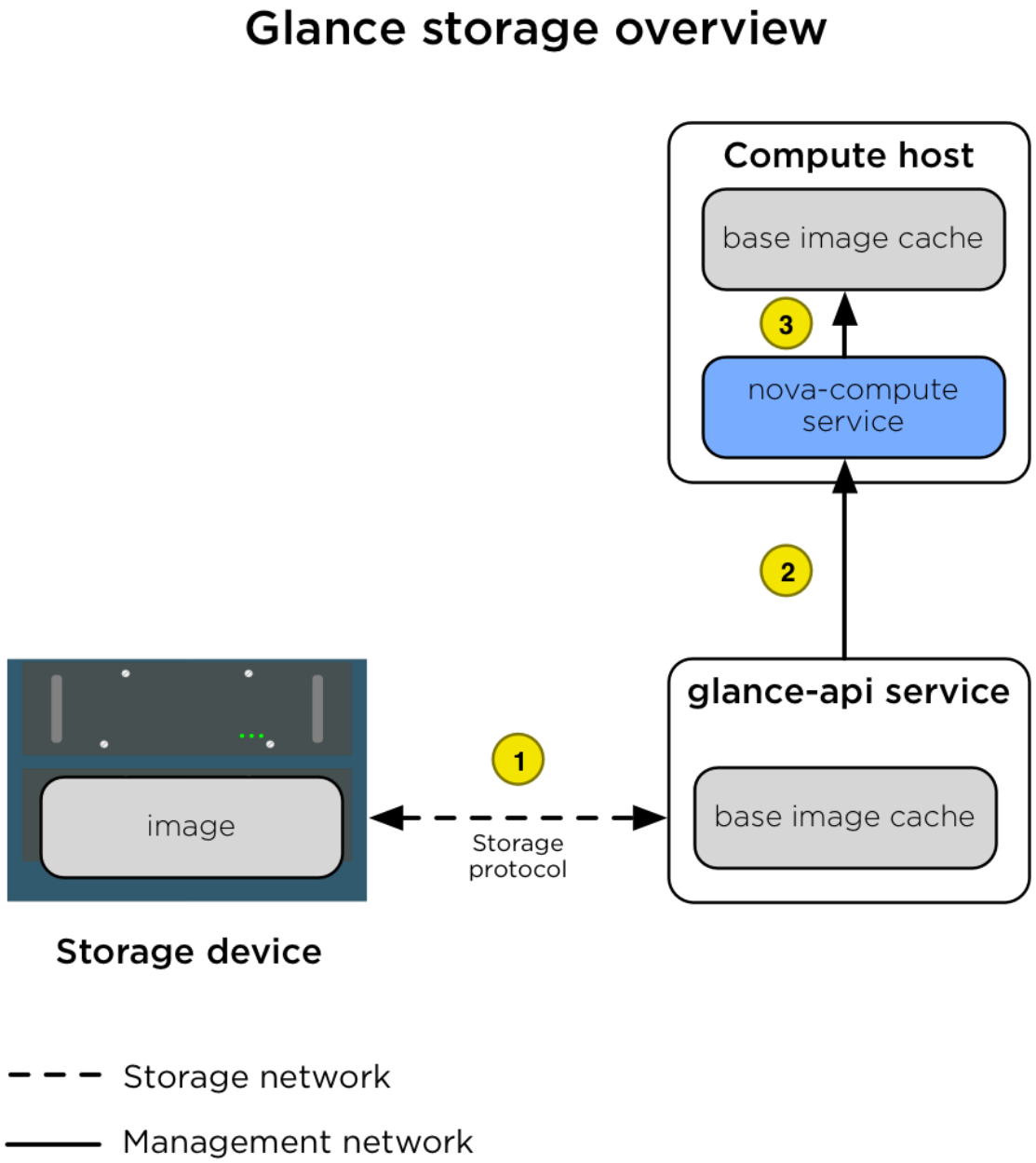
For more details refer to: Glance

Glance

- Glance is the OpenStack Image Storage service to store images on a variety of storage back ends supported by the glance_store drivers.
- It enables users to discover, register, and retrieve virtual machine images. It offers a REST API that enables you to query virtual machine image metadata and retrieve an actual image.
- You can store virtual machine images made available through the Image service in a variety of locations, from simple file systems to object-storage systems like OpenStack Object Storage.
- It is central to Infrastructure-as-a-Service (IaaS). It accepts API requests for disk or server images, and metadata definitions from end users or OpenStack Compute components.
- It also supports the storage of disk or server images on various repository types, including OpenStack Object Storage.
- A number of periodic processes run on the OpenStack Image service to support caching.
- Replication services ensure consistency and availability through the cluster.
- Other periodic processes include auditors, updaters, and reapers.

Glance (cont..)

- The Glance Storage Overview diagram illustrates the interactions between the Image service, the storage device, and the nova-compute service when an instance is created.
- The diagram shows the following steps.
    1. When a client requests an image, the glance-api service accesses the appropriate store on the storage device over the storage network (br-storage) and pulls it into its cache. When the same image is requested again, it is given to the client directly from the cache.
    2. When an instance is scheduled for creation on a Compute host, the nova-compute service requests the image from the glance-api service over the management network (br-mgmt).
    3. After the image is retrieved, the nova-compute service stores the image in its own image cache. When another instance is created with the same image, the image is retrieved from the local base image cache.



Vmware vSphere

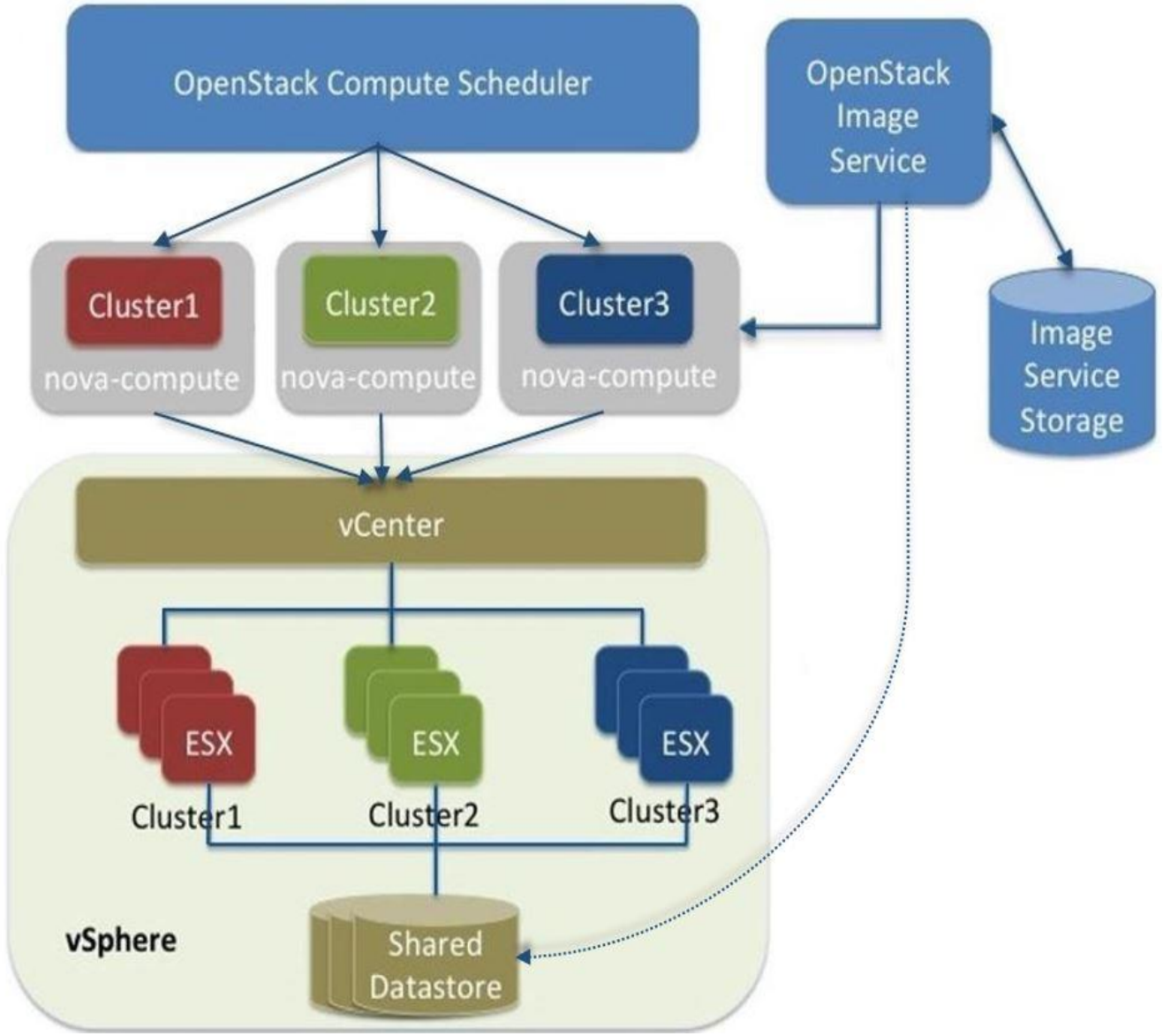
For more details refer to: VMware vSphere

VMware vSphere

- OpenStack Compute supports the VMware vSphere product family and enables access to advanced features such as vMotion, High Availability, and Dynamic Resource Scheduling (DRS).
- The VMware vCenter driver enables the nova-compute service to communicate with a VMware vCenter server that manages one or more ESX (Elastic Sky X : an enterprise-class, type-1 hypervisor) clusters.
- The driver aggregates the ESX hosts in each cluster to present one large hypervisor entity for each cluster to the Compute scheduler.
- Because individual ESX hosts are not exposed to the scheduler, Compute schedules to the granularity of clusters and vCenter uses DRS to select the actual ESX host within the cluster. When a virtual machine makes its way into a vCenter cluster, it can use all vSphere features.

VMware Driver Architecture

- In this figure, the OpenStack Compute Scheduler sees three hypervisors that each correspond to a cluster in vCenter.
- nova-compute contains the VMware driver. You can run with multiple nova-compute services.
- It is recommended to run with one nova-compute service per ESX cluster thus ensuring that while Compute schedules at the granularity of the nova-compute service it is also in effect able to schedule at the cluster level.
- In turn the VMware driver inside nova-compute interacts with the vCenter APIs to select an appropriate ESX host within the cluster. Internally, vCenter uses DRS for placement.



VMware Driver Architecture (cont..)

- The VMware vCenter driver also interacts with the Image service to copy VMDK images from the Image service back-end store.
- The dotted line in the figure represents VMDK images being copied from the OpenStack Image service to the vSphere data store.
- VMDK images are cached in the data store so the copy operation is only required the first time that the VMDK image is used.
- After OpenStack boots a VM into a vSphere cluster, the VM becomes visible in vCenter and can access vSphere advanced features.
- At the same time, the VM is visible in the OpenStack dashboard and you can manage it as you would any other OpenStack VM.
- You can perform advanced vSphere operations in vCenter while you configure OpenStack resources such as VMs through the OpenStack dashboard.

Apache CloudStack

For more details refer to: Apache CloudStack

Apache CloudStack

- Apache CloudStack is an open source Infrastructure-as-a-Service platform that manages and orchestrates pools of storage, network, and computer resources to build a public or private IaaS compute cloud.
- With CloudStack you can:
  - Set up an on-demand elastic cloud computing service.
  - Allow end-users to provision resources

- Apache CloudStack provides:
  1. Multiple Hypervisor Support
  2. Massively Scalable Infrastructure Management
  3. Automatic Cloud Configuration Management
  4. Graphical User Interface
  5. API
  6. AWS EC2 API Support
  7. High Availability

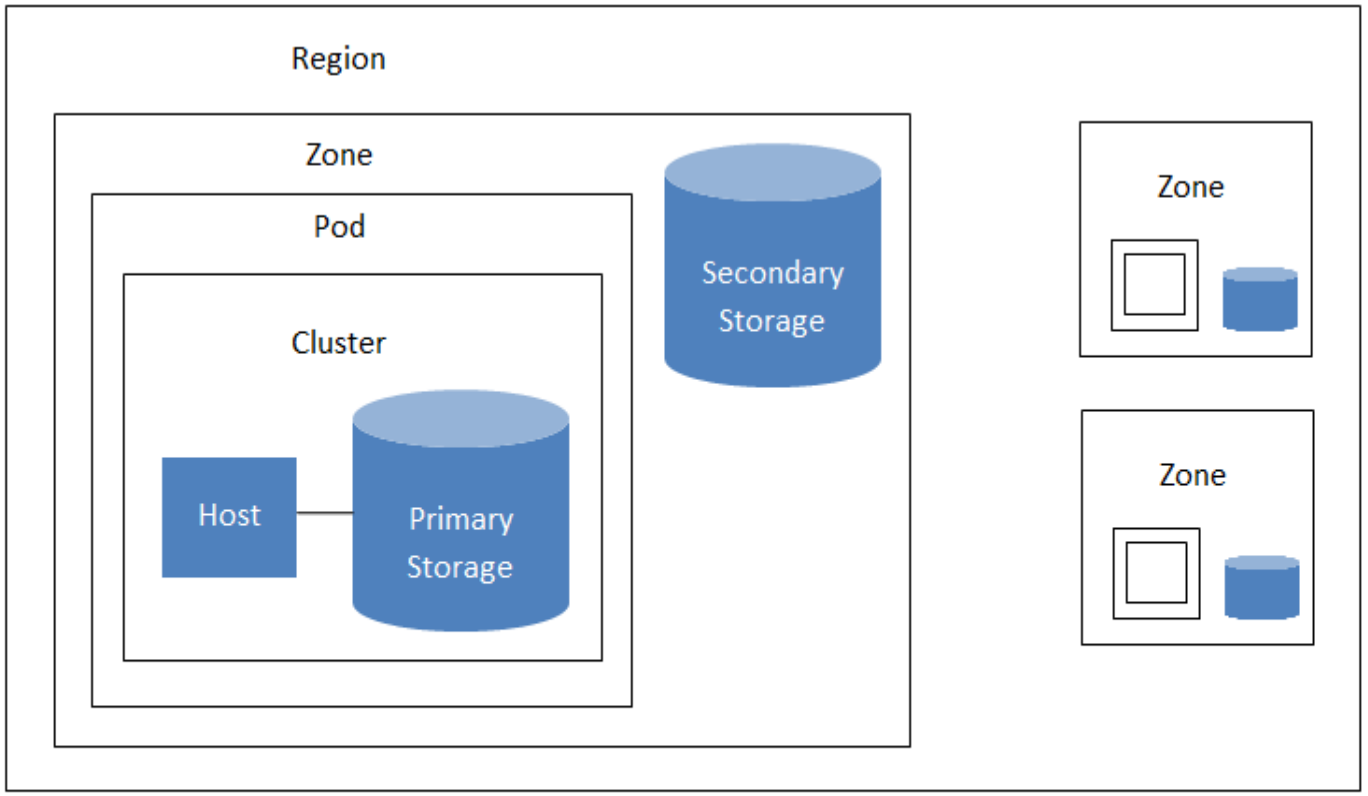
Apache CloudStack: Management Server Overview

- The management server orchestrates and allocates the resources in your cloud deployment.
- The management server typically runs on a dedicated machine or as a virtual machine. It controls allocation of virtual machines to hosts and assigns storage and IP addresses to the virtual machine instances. The Management Server runs in an Apache Tomcat container and requires a MySQL database for persistence.

- The management server:
  - Provides the web interface for both the adminstrator and end user.
  - Provides the API interfaces for both the CloudStack API as well as the EC2 interface.
  - Manages the assignment of guest VMs to a specific compute resource
  - Manages the assignment of public and private IP addresses.
  - Allocates storage during the VM instantiation process.
  - Manages snapshots, disk images (templates), and ISO images.
  - Provides a single point of configuration for your cloud.

Apache CloudStack: Cloud Infrastructure Overview

- Resources within the cloud are managed as follows:
  - Regions: A collection of one or more geographically proximate zones managed by one or more management servers.
  - Zones: Typically, a zone is equivalent to a single datacenter. A zone consists of one or more pods and secondary storage.
  - Pods: A pod is usually a rack, or row of racks that includes a layer-2 switch and one or more clusters.
  - Clusters: A cluster consists of one or more homogenous hosts and primary storage.
  - Host: A single compute node within a cluster; often a hypervisor.
  - Primary Storage: A storage resource typically provided to a single cluster for the actual running of instance disk images. (Zone-wide primary storage is an option, though not typically used.)
  - Secondary Storage: A zone-wide resource which stores disk templates, ISO images, and snapshots.



A region with multiple zones

Apache CloudStack: Networking Overview

- CloudStack offers many types of networking, but they typically fall into one of two scenarios:
  - Basic: Most analogous to AWS-classic style networking. Provides a single flat layer-2 network where guest isolation is provided at layer-3 by the hypervisors bridge device.
  - Advanced: This typically uses layer-2 isolation such as VLANs, though this category also includes SDN technologies such as Nicira NVP.

Apache CloudStack: Primary Storage

- Primary storage is associated with a cluster, and it stores virtual disks for all the VMs running on hosts in that cluster. On KVM and VMware, you can provision primary storage on a per-zone basis.
- You can add multiple primary storage servers to a cluster or zone. At least one is required. It is typically located close to the hosts for increased performance. CloudStack manages the allocation of guest virtual disks to particular primary storage devices.
- It is useful to set up zone-wide primary storage when you want to avoid extra data copy operations. With cluster-based primary storage, data in the primary storage is directly available only to VMs within that cluster. If a VM in a different cluster needs some of the data, it must be copied from one cluster to another, using the zone's secondary storage as an intermediate step. This operation can be unnecessarily time-consuming.
- CloudStack is designed to work with all standards-compliant iSCSI and NFS servers that are supported by the underlying hypervisor, including, for example:
  - SolidFire for iSCSI
  - Dell EqualLogic™ for iSCSI
  - Network Appliances filers for NFS and iSCSI
  - Scale Computing for NFS
  - Dell EMC PowerFlex™ (v3.5)
- If you intend to use only local disk for your installation, you can skip adding separate primary storage.

Apache CloudStack: Secondary Storage

- Secondary storage stores the following:
  - Templates — OS images that can be used to boot VMs and can include additional configuration information, such as installed applications
  - ISO images — disc images containing data or bootable media for operating systems
  - Disk volume snapshots — saved copies of VM data which can be used for data recovery or to create new templates
- The items in secondary storage are available to all hosts in the scope of the secondary storage, which may be defined as per zone or per region.
- To make items in secondary storage available to all hosts throughout the cloud, you can add object storage in addition to the zone-based NFS Secondary Staging Store.
- It is not necessary to copy templates and snapshots from one zone to another, as would be required when using zone NFS alone. Everything is available everywhere.
- CloudStack provides plugins that enable both OpenStack Object Storage (Swift, swift.openstack.org) and Amazon Simple Storage Service (S3) object storage.

Thank you!