In this chapter we will learn

Exploring the cloud computing stack

Connecting to cloud

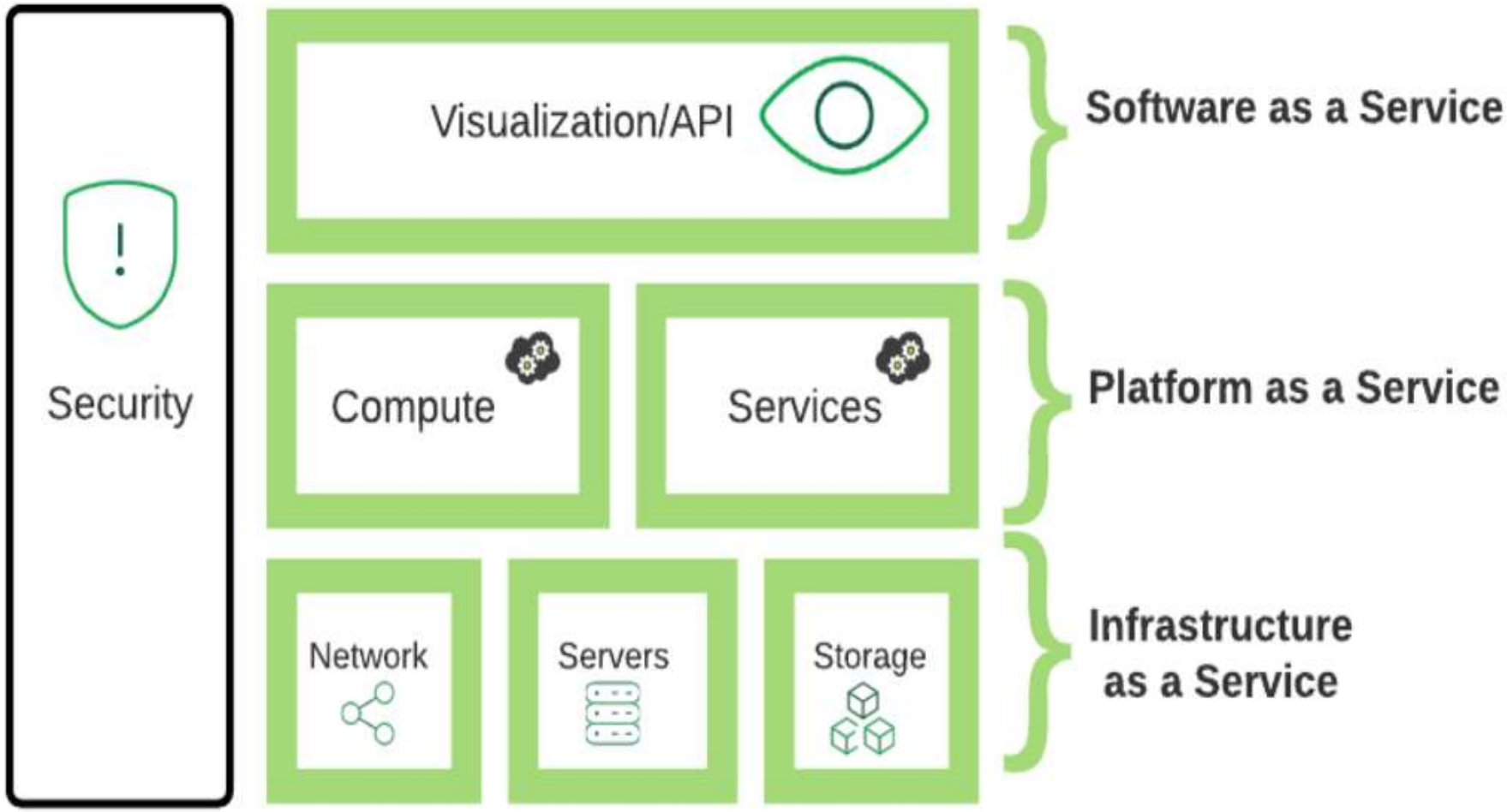
Understanding cloud architecture

- Workload distribution architecture,
- Cloud bursting architecture,
- Disk provisioning architecture,
- Dynamic failure detection and recovery architecture

Capacity planning

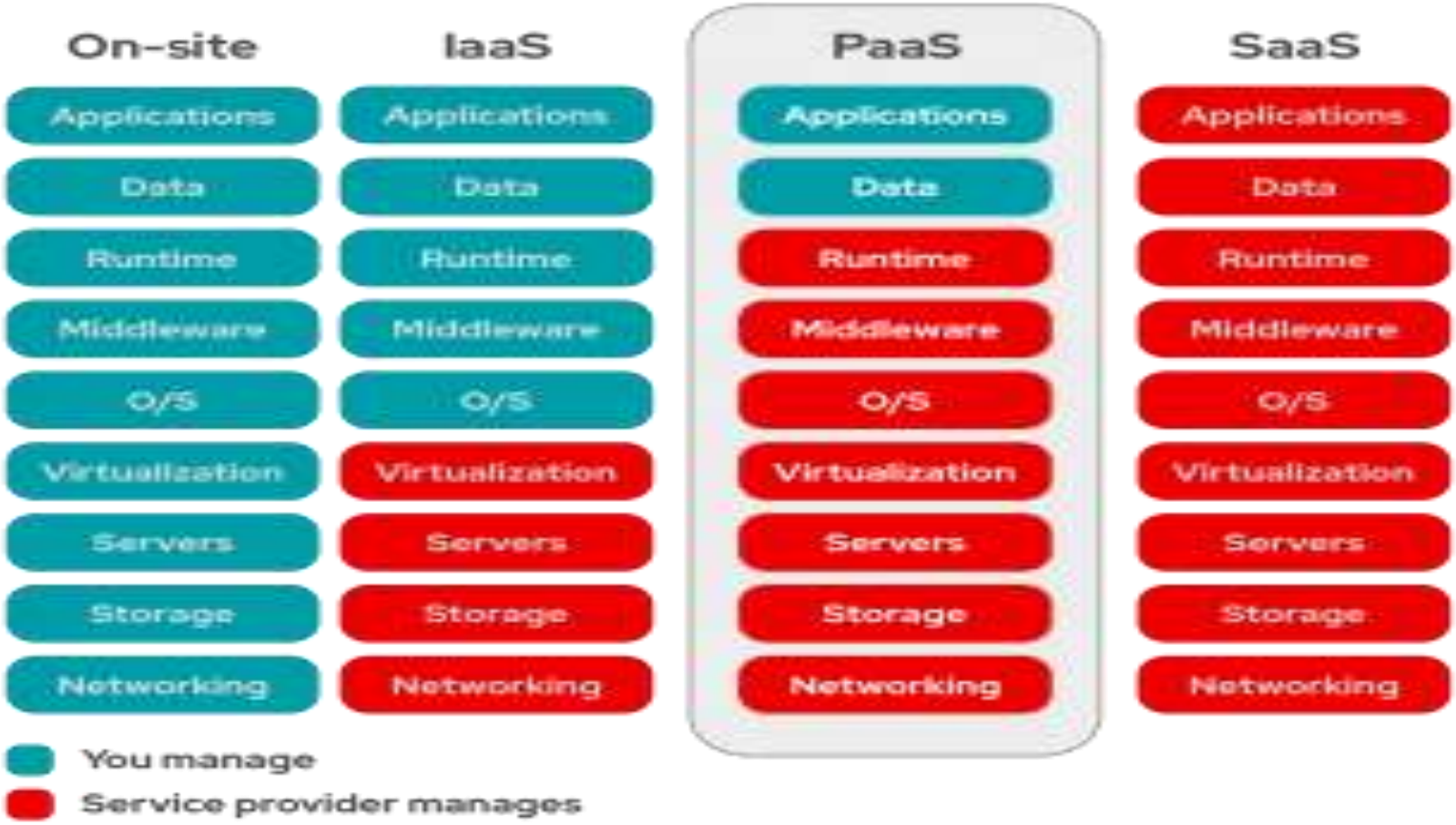
Cloud mechanisms :

- Automated scaling listerner,
- Load balancer,
- Pay-per-use monitor, Audit monitor, SLA monitor,
- Fail-over Systems, Resource Cluster



Cloud Computing Stack

Cloud computing stack



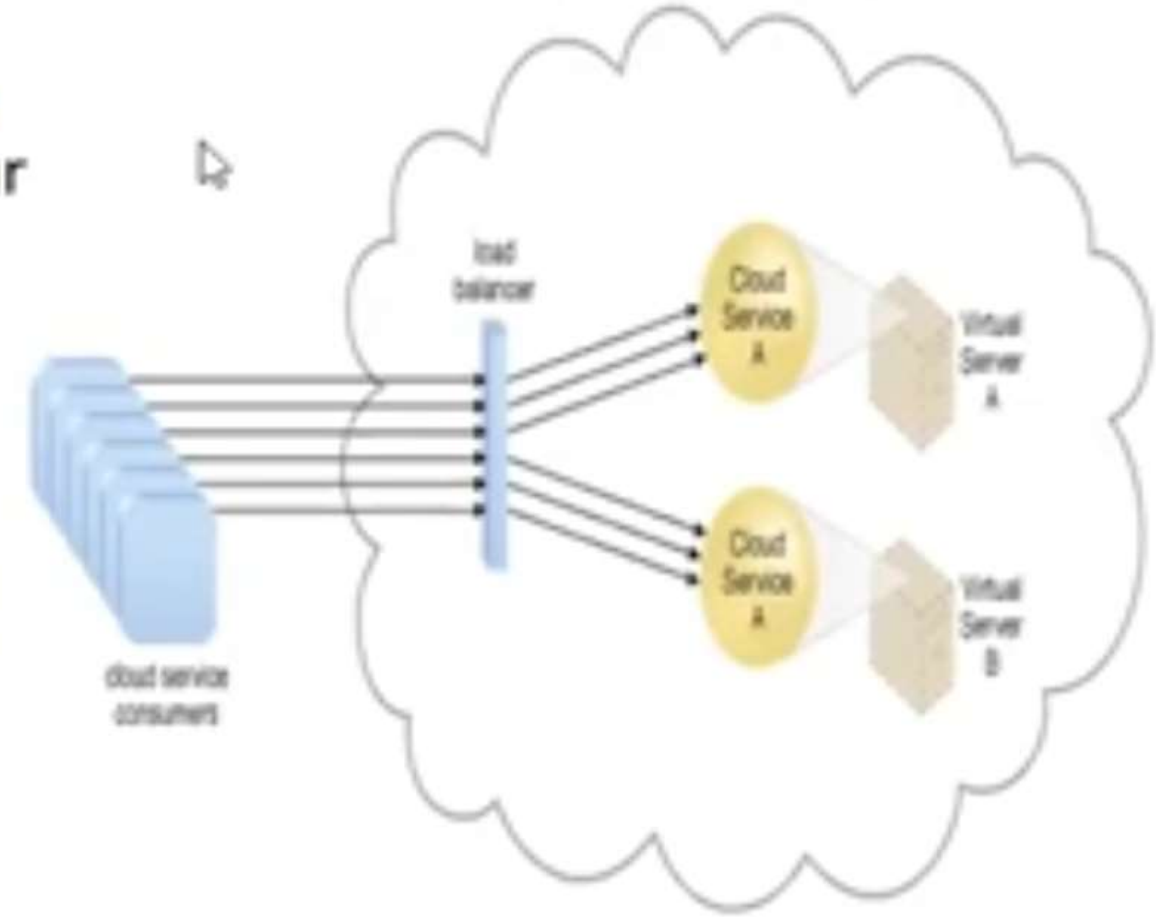
Composability	• **Composability** is a system design principle that deals with the inter-relationships of components. A highly **composable** system provides components that can be selected and assembled in various combinations to satisfy specific user requirements.
Infrastructure	• Virtual servers described in terms of a machine image or instance have characteristics that often can be described in terms of real servers delivering a certain number of microprocessor (CPU) cycles, memory access, and network bandwidth to customers.
Platforms	• Provisioning various platforms to users to customize and develop applications. Development, testing, deployments are made easier through this medium.
Virtual Appliances	• The machines that are installed in order to run services in cloud. These are platform instances in particular that can be provisioned to cloud users.
Communication Protocols	• Common XML based set of protocols used as the messaging format are the Simple Object Access Protocol (SOAP) protocol as the object model, and a set of discovery and description protocols based on the Web Services Description Language (WSDL) to manage transactions in cloud.
Applications	• Services running in the cloud

Understanding cloud architecture

What Kind of Architecture?

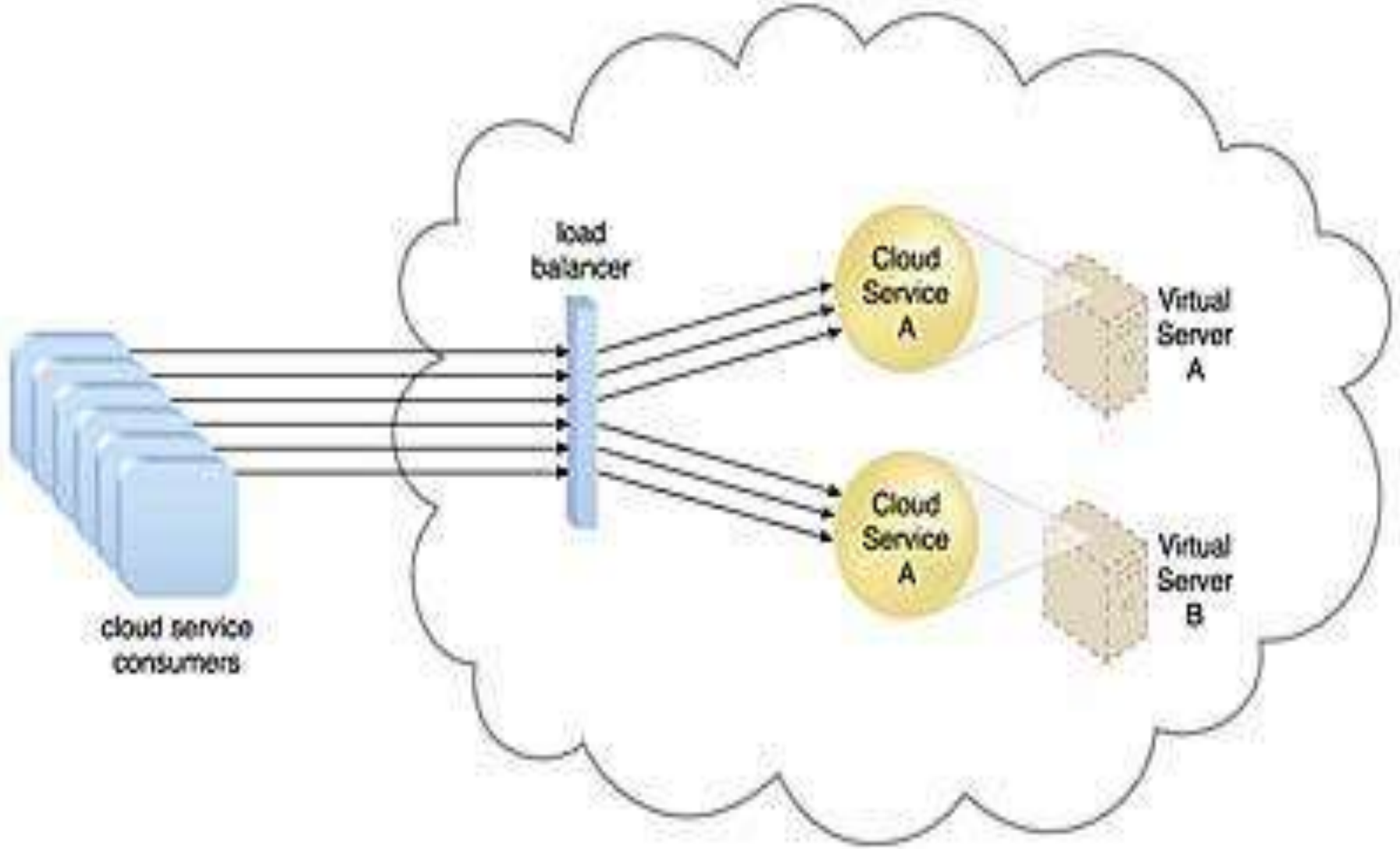
Workload distribution Architecture

- Workload is an abstraction of the actual work that your instance or a set of instances are going to perform.
- a load balancer that provides runtime logic capable of distributing the workload among the available IT resources.



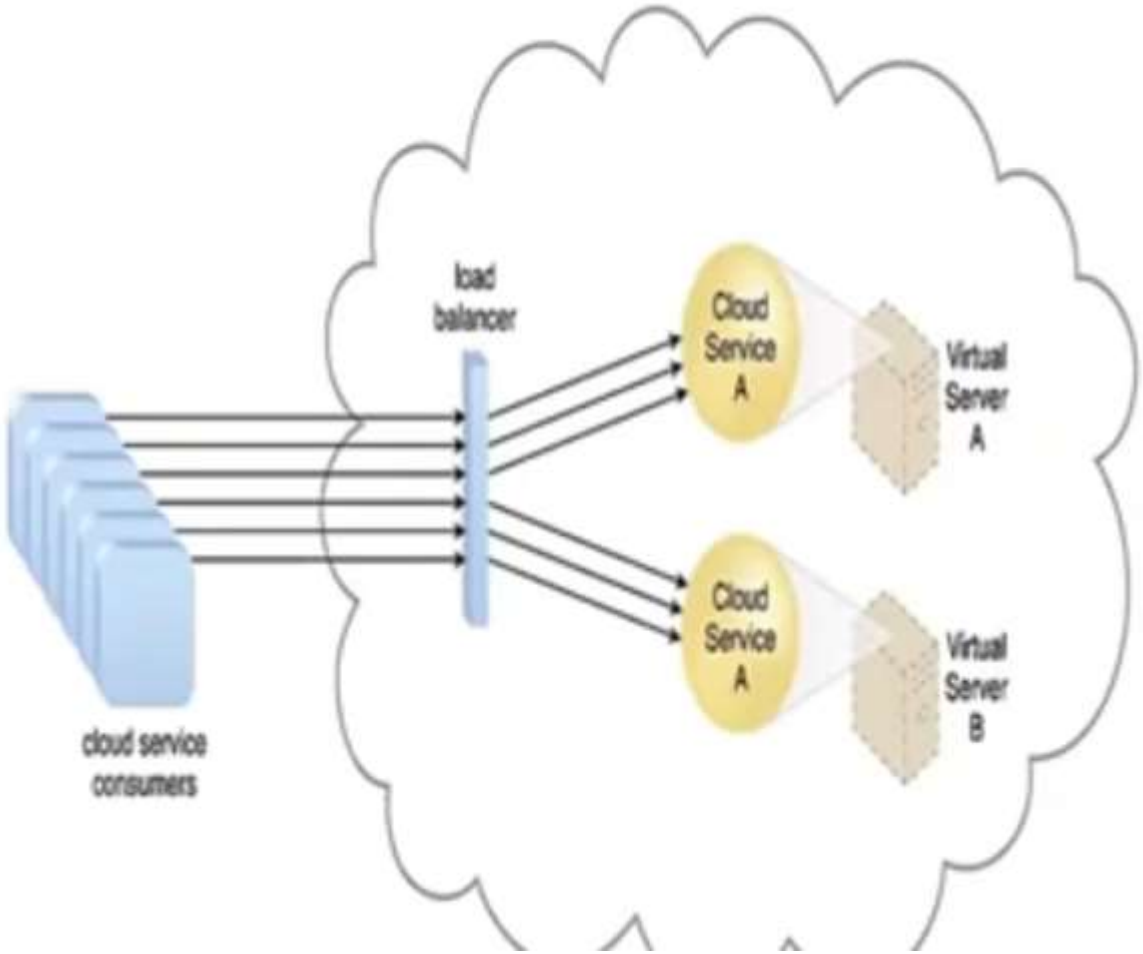
Workload Distribution Architecture

- Workload distribution architecture uses IT resources that can be horizontally scaled with the use of one or more identical IT resources.
- This is accomplished through the use of a load balancer that provides runtime logic which distributes the workload among the available IT assets evenly.
- This model can be applied to any IT resource and is commonly used with; distributed virtual servers, cloud storage devices, and cloud services.
- In addition to a load balancer and the previously mentioned resources, the following mechanisms can also be a part of this model:
  - Cloud Usage Monitor that can carry out run-time tracking and data processing.
  - Audit Monitor used for monitoring the system as may be required to fulfill legal requirements. Hypervisor which is used to manage workloads and virtual hosts that require distribution. Logical network perimeter which isolates cloud consumer network boundaries. Resource clusters commonly used to support workload balancing between cluster nodes. Resource replication which generates new instances of virtualized resources under increased workloads.


load
balancer
Cloud
Service
A
Cloud
Service
A
Virtual
Server
A
Virtual
Server
B
cloud service
consumers

Workload distribution Architecture

- A redundant copy of Cloud Service A is implemented on Virtual Server B.
- The load balancer intercepts cloud service consumer requests and
- directs them to both Virtual Servers A and B to ensure even workload distribution.



Working of Workload Distribution Architecture:-

The workload architecture model basically functions as follows:

- Resource A and resource B are exact copies of the same resource.
- Inbound requests from consumers are handled by the load balancer which forwards the request to the appropriate resource dependent on workload being handled by each resource.
- In other words, if resource A is busier than resource B, it will forward the resource request to resource B.
- In this manner this model distributes the load among the available IT resources based on workload of each resource.

In addition to the base load balancer mechanism, and the virtual server and cloud storage device mechanisms to which load balancing can be applied, the following mechanisms can also be part of this cloud architecture:

Audit Monitor – When distributing runtime workloads, the type and geographical location of the IT resources that process the data can determine whether monitoring is necessary to fulfill legal and regulatory requirements.

Cloud Usage Monitor – Various monitors can be involved to carry out runtime workload tracking and data processing.

Hypervisor – Workloads between hypervisors and the virtual servers that they host may require distribution.

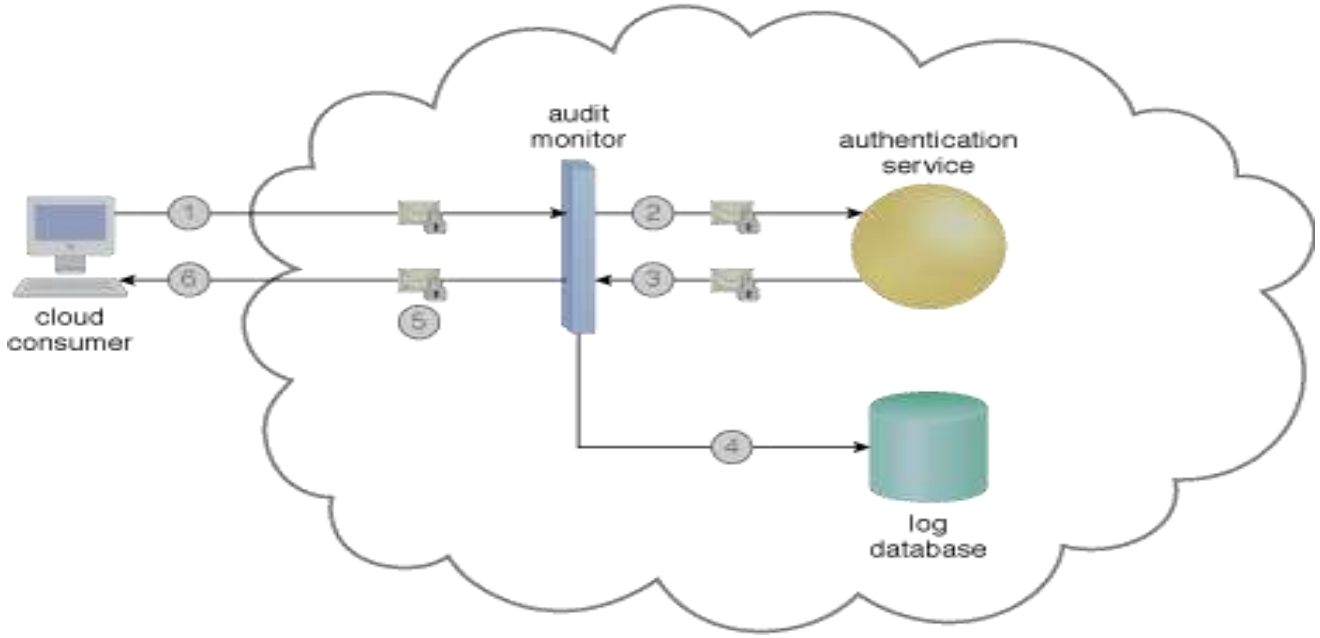
Logical Network Perimeter – The logical network perimeter isolates cloud consumer network boundaries in relation to how and where workloads are distributed.

Resource Cluster – Clustered IT resources in-active/active mode are commonly used to support workload balancing between different cluster nodes.

Resource Replication – This mechanism can generate new instances of virtualized IT resources in response to runtime workload distribution demands

Audit Monitor

The audit monitor mechanism is used to collect audit tracking data for networks and IT resources in support of, or dictated by, regulatory and contractual obligations. The figure depicts an audit monitor implemented as a monitoring agent that intercepts "login" requests and stores the requestor's security credentials, as well as both failed and successful login attempts, in a log database for future audit reporting purposes.



1) A cloud service consumer requests access to a cloud service by sending a login request message with security credentials.
2) The audit monitor intercepts the message
3) And forwards it to the authentication service.
4) The authentication service processes the security credentials. A response message is generated for the cloud service consumer, in addition to the results from the login attempt.
5) The audit monitor intercepts the response message and stores the entire collected login event details in the log database, as per the organization's audit policy requirements.
6) Access has been granted, and a response is sent back to the cloud service consumer.

cloud usage monitor

Cloud monitoring or cloud usage monitor is managing and reviewing operational processes in the cloud infrastructure. This is implemented via automatic monitoring software, giving control and central access over cloud infrastructure.

How Cloud Monitoring Works

Cloud has plenty of moving parts, so it is very important to make sure that everything works fine in order to enhance performance. Primarily, cloud monitoring contains functions like –

- Website Monitoring: Tracks the traffic, processes, resource utilization, and availability of hosted websites

- Virtual Machine Monitoring: It monitors the particular virtual machines and virtualization infrastructure

.

- Database Monitoring: It monitors the queries, consumption, availability, and processes of database resources

- Virtual Network Monitoring: In this method, devices, performance, connections and virtual network is monitored

- Cloud Storage Monitoring: In this monitoring method, storage resources and processes related to services, databases, virtual machines, and applications are monitored

Logical network perimeters

The logical network perimeter establishes a virtual network boundary that can encompass and isolate a group of related cloud-based IT resources that may be physically distributed. It is defined as the isolation of a network environment from the rest of a communications network. The logical network perimeter can be implemented to:

- isolate IT resources in a cloud from non-authorized users
- control the bandwidth that is available to isolated IT resources

Logical network perimeters are typically established via network devices that supply and control the connectivity of a data center and are commonly deployed as virtualized IT environments that include:

Virtual Firewall – An IT resource that actively filters network traffic to and from the isolated network while controlling its interactions with the Internet.
Virtual Network – Usually acquired through VLANs, this IT resource isolates the network environment within the data center infrastructure.

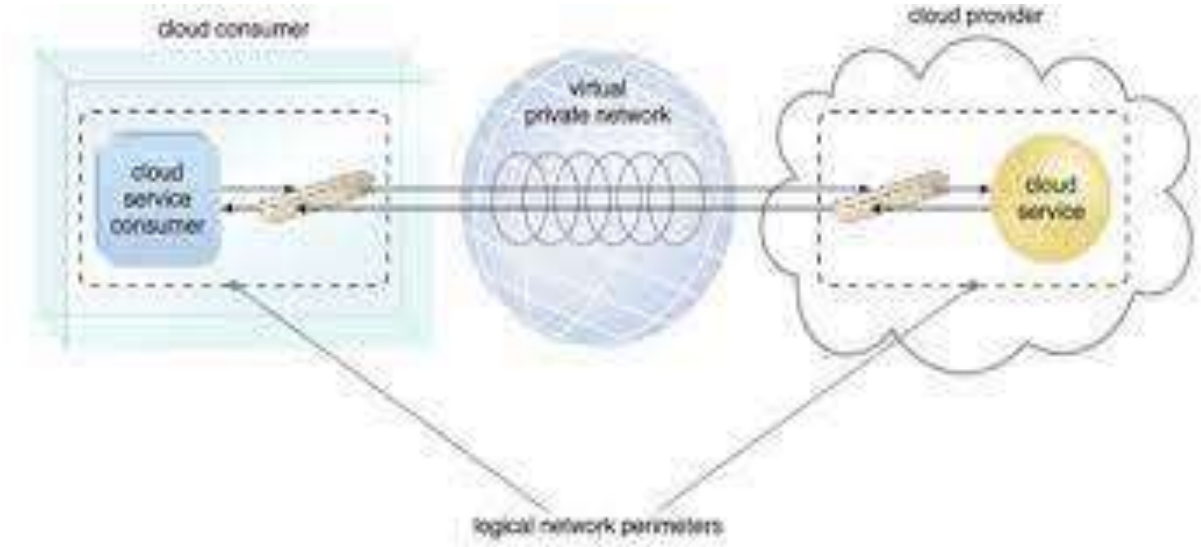


Figure depicts a scenario in which one logical network perimeter contains a cloud consumer's on-premise environment, while another contains a cloud provider's cloud-based environment. These perimeters are connected through a VPN (virtual private network) that protects communications, since the VPN is typically implemented by point-to-point encryption of the data packets sent between the communicating endpoints.

Capacity planning

- For available resources, capacity planning seeks a heavy demand. It determines whether the systems are working properly, used to measure their performance, determine the usage of patterns and predict future demand of cloud-capacity. This also adds an expertise planning for improvement and optimizes performance. The goal of capacity planning is to maintain the workload without improving the efficiency. Tuning of performance and work optimization is not the major target of capacity planners.
- It measures the maximum amount of task that it can perform. The capacity planning for cloud technology offers the systems with more enhanced capabilities including some new challenges over a purely physical system.

Capacity Planning

- Expected Demand
- Determining your expected demand
- Analyzing the unexpected
- The Impact of Load

Capacity Planning

- Capacity planning is basically developing a **strategy** that **guarantees** your infrastructure *can support* the **resource demands** placed on it.

Expected Demand

Well-quantified expectation that will **enable** to:

1. **Plan out** an infrastructure to support expected loads.
2. **Recognize** when actual load is diverging in a meaningful way from expected load.
3. **Understand** the impact of changing application requirements on your infrastructure.

Consider, for **example**, the scenario in which you have an infrastructure that supports **10 million transactions/second** and you have a **growth** from your **average load of 1 million transactions/second to 5 million transactions/second.**

If you had **properly estimated the load**, you would **recognize** whether the sudden surge was **expected** (and thus nothing to be concerned about) or **unexpected** and thus something to watch for potential capacity problems.

Determining your expected demand

FIGURE: The projected daily load on an e-commerce site (Days vs Traffic)

Figures provide charts illustrating the **expected traffic** for a e-commerce site over the course of a typical **day** as well a projected **peak volumes** over the next **12 months**.

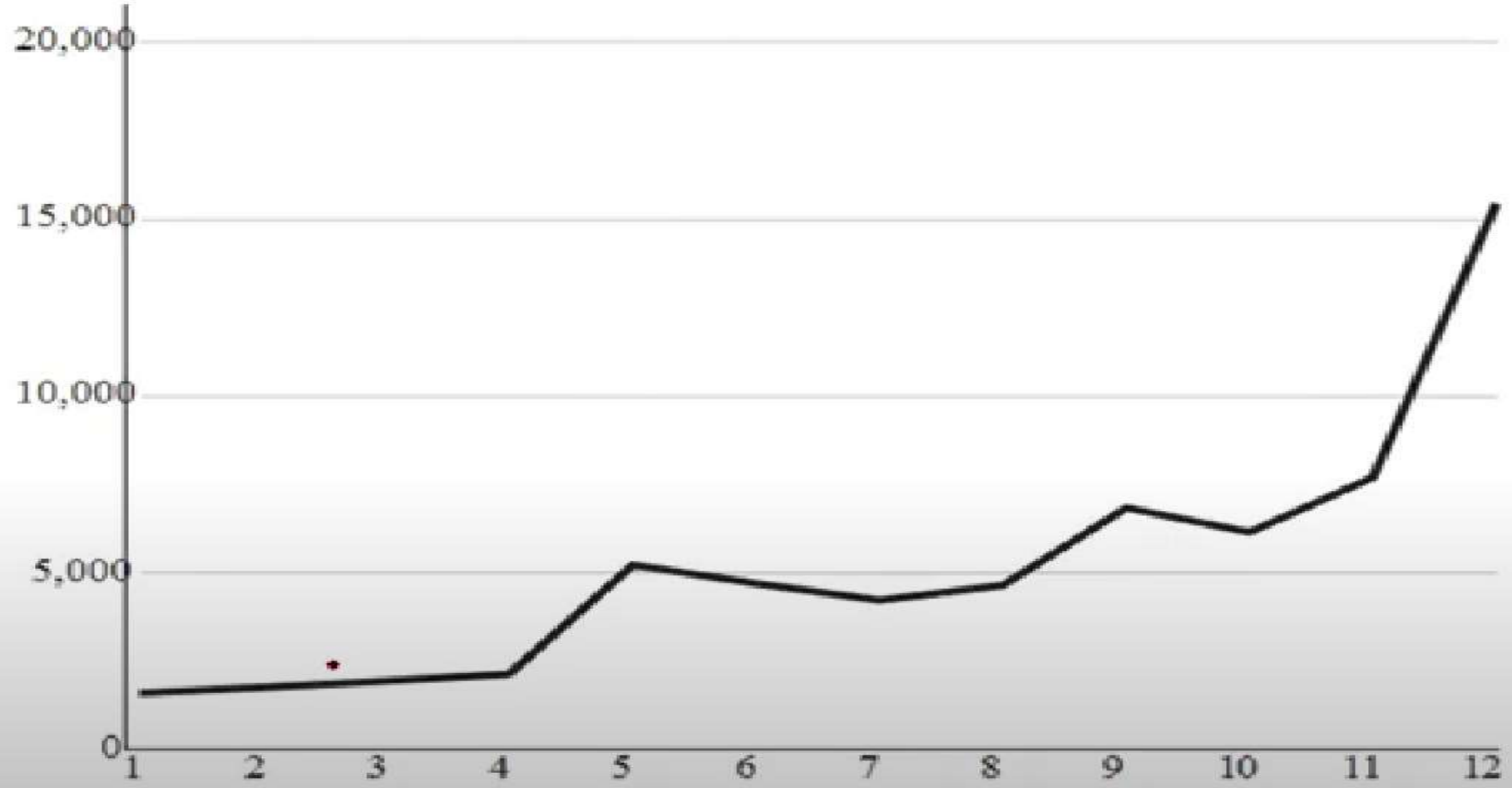


FIGURE: The expected load on the e-commerce site over the next 12 month

Analyzing the unexpected

- ✓ Sometimes the company launches a **product** that exceeds expectations.
- ✓ Sometimes you receive unplanned **media** coverage.
- ✓ Sometimes you are simply going to be **wrong**.

- **Any time** significant unexpected traffic hits your web systems, it's important to **understand why traffic is varying** from the unexpected.

The Impact of Load

- Assuming a perfectly efficient web application and database engine, t typical web application deployed into the cloud has all of these poten capacity constraints:
  1. The **bandwidth** into the load balancer
  2. The **CPU** and **RAM** of the **load balancer**
  3. The ability of the load balancer to properly spread load across the application servers
  4. The **bandwidth** between the load balancer and the **application servers**
  5. The **CPU** and **RAM** of the application server
  6. The **disk I/O** for read operations on the application server
  7. The **write I/O** for disk operations on the application server, a secondary disk constraint if the application is caching on the disk
  8. The **bandwidth** between the application server and **any network storage** devices (such as Amazon elastic block storage devices)
  9. The bandwidth between the application server and the database server
  10. The disk I/O for **read** operations on the read database
  11. The disk I/O for **write** operations on the write database

.

Cloud capacity planning aims to match demand with available resources.

- It analyzes what systems are already in place,
- measuring their performance and predicting demand.
- Your organization can then provision and
- allocate cloud resources based on that demand

Steps in Capacity planning

- Determine the distinctiveness of the present system.
- Determine the working load for different resources in the system such as CPU, RAM, network, etc.
- Load the system until it gets overloaded; & state what's requiring to uphold acceptable performance.
- Predict the future based on older statistical reports & other factors.
- Deploy resources to meet the predictions & calculations.
- Repeat step (i) through (v) as a loop.

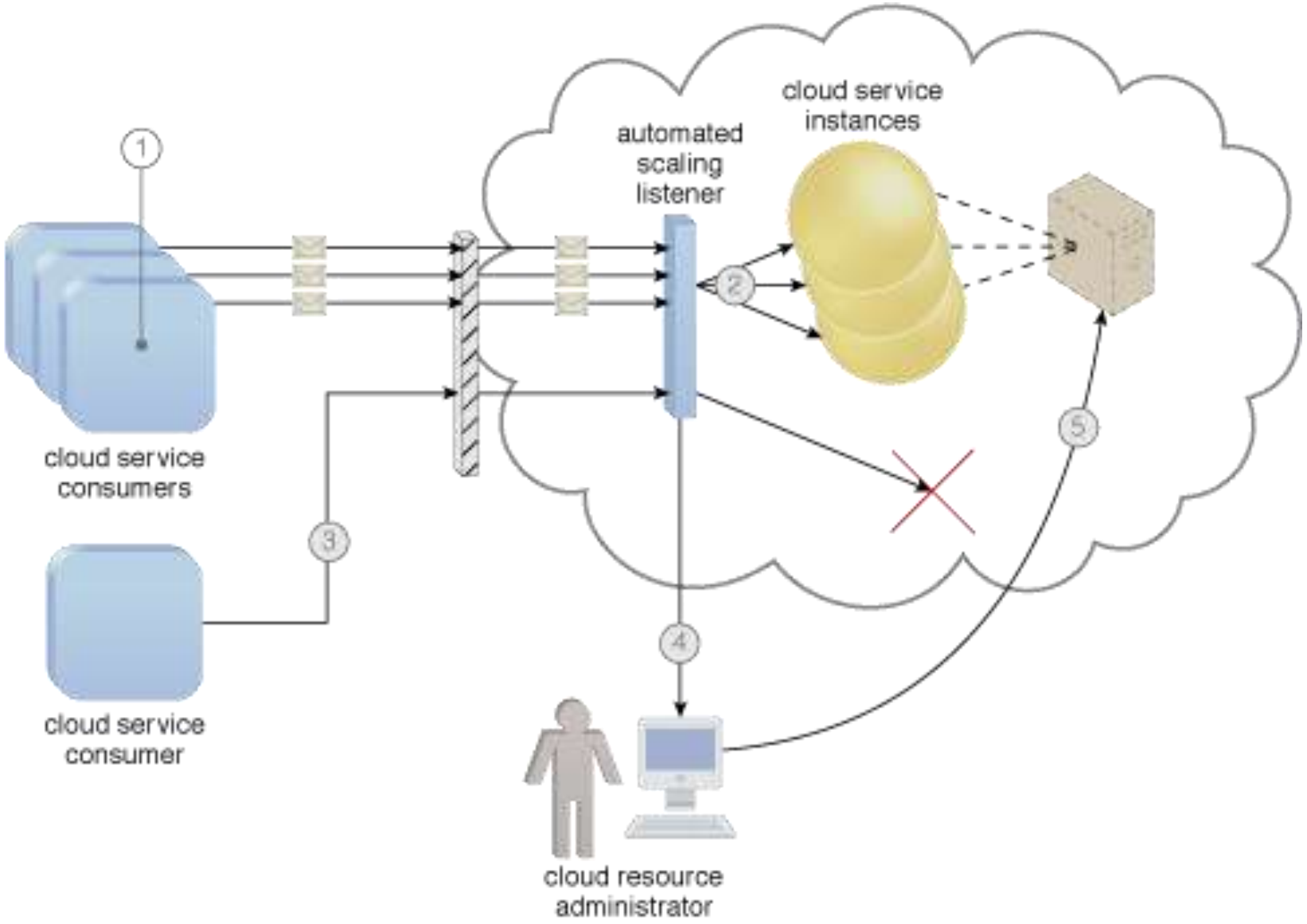
Cloud Bursting architecture:-

Cloud Bursting

Cloud bursting is an application deployment model in which an application runs in an internal cloud or a proprietary computing architecture which provides hosted services to a limited number of people (also known as a private cloud).

This kind of architecture is behind a company's firewall. The application could also be at a data centre.

When the demand of the computing processes reaches its capacity and begins to spike, it **"bursts"** into a public cloud (available to the general public) when the demand for computing capacity sharply increases.


1
cloud service
consumers
3
cloud service
consumer
automated
scaling
listener
cloud service
instances
2
4
5
cloud resource
administrator

Cloud bursting Architecture

- The *cloud bursting architecture* establishes a form of dynamic scaling
- that scales IT resources into a cloud whenever predefined capacity thresholds have been reached.

Cloud bursting Architecture

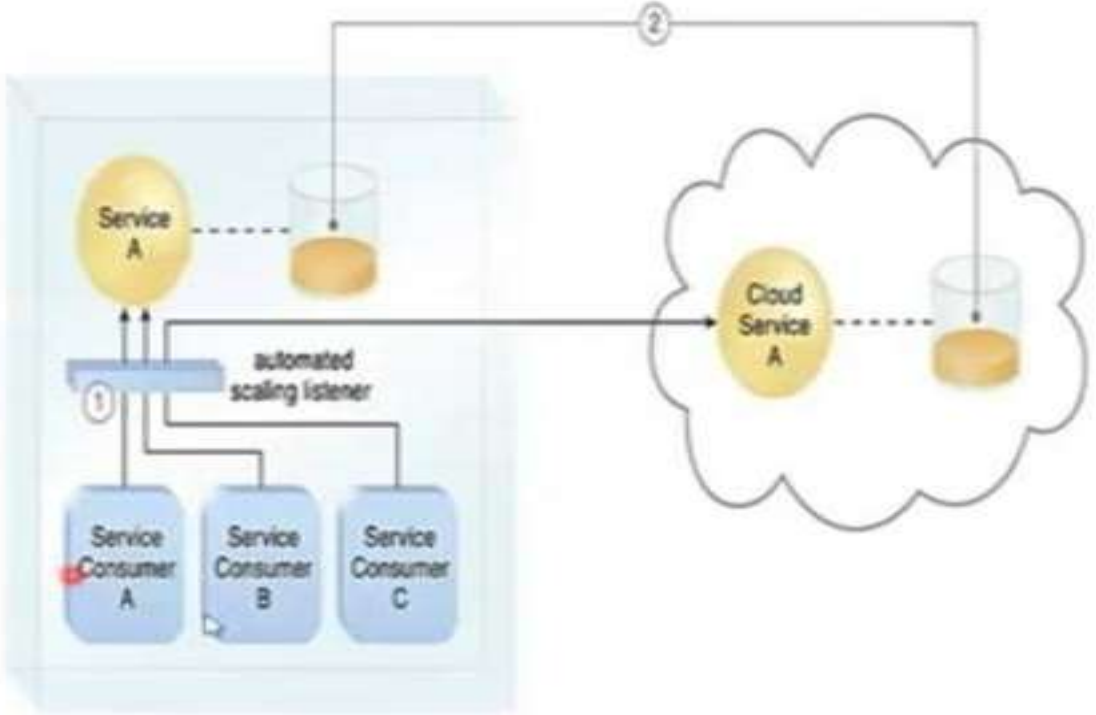
- The corresponding cloud-based IT resources are redundantly pre-deployed but remain inactive until cloud bursting occurs.
- After they are no longer required, the cloud-based IT resources are released and the architecture "bursts in" back to the on-premise environment.

Cloud bursting Architecture

- Cloud bursting is a flexible scaling architecture
- that provides cloud consumers with the option of using cloud-based IT resources only to meet higher usage demands.
- The architectural model is based on the automated scaling listener and resource replication mechanisms.

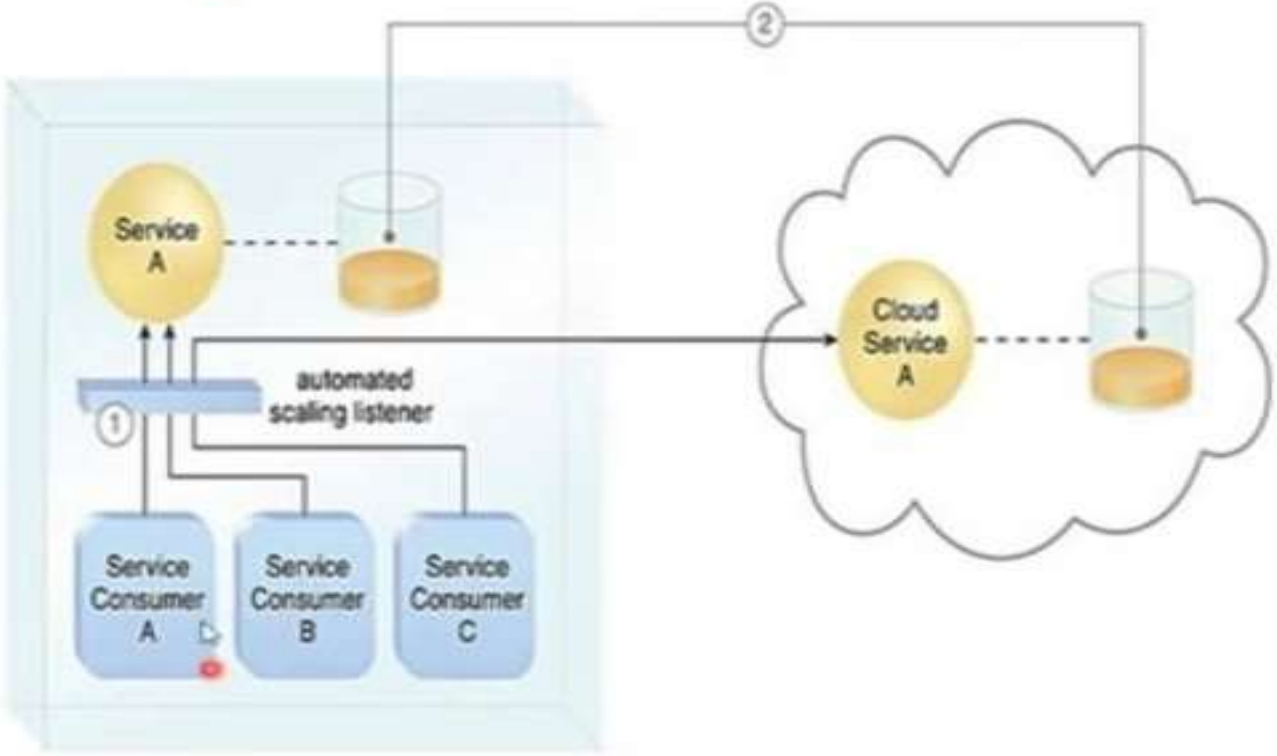
Cloud bursting Architecture

- The automated scaling listener determines when to redirect requests to cloud-based IT resources,
- and resource replication is used to maintain synchronicity between on-premise and cloud-based IT resources in relation to state information



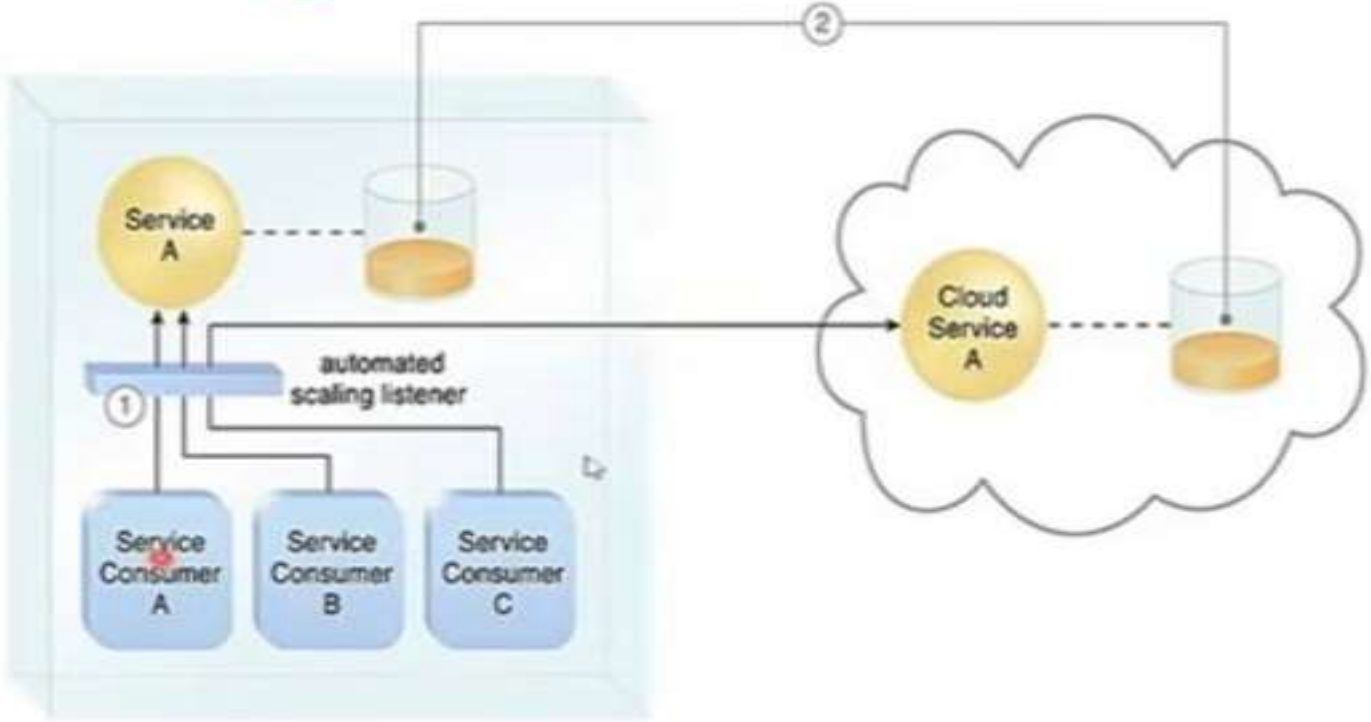
Cloud bursting Architecture

- An automated scaling listener monitors the usage of on-premise Service A, and redirects Service Consumer C's request to Service A's redundant implementation in the cloud (Cloud Service A) once Service A's usage threshold has been exceeded (1).



Cloud bursting Architecture

- A resource replication system is used to keep state management databases synchronized(2)
.



Cloud bursting Architecture

- In addition to the automated scaling listener and resource replication, numerous other mechanisms can be used to automate the burst in and out dynamics for this architecture, depending on the type of IT resource being scaled.

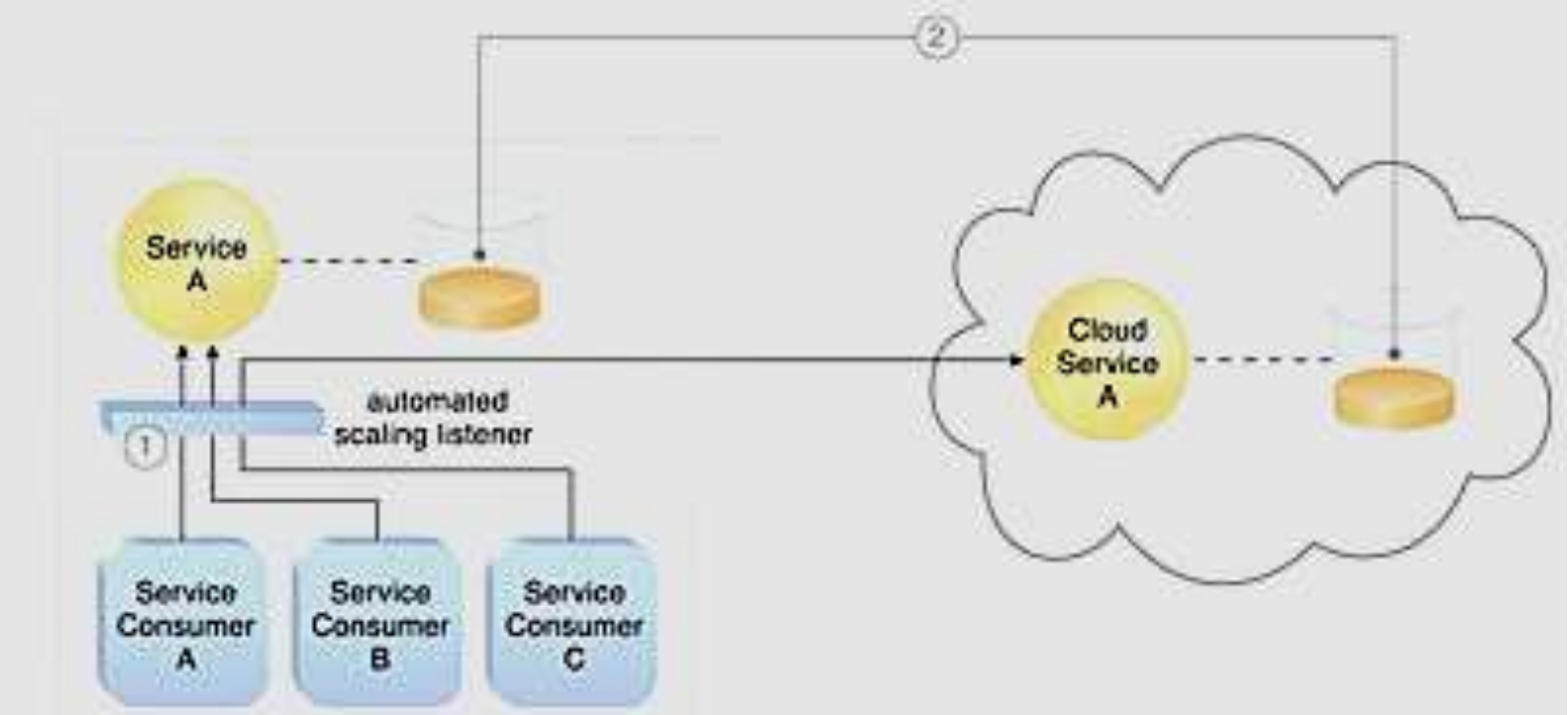
Cloud Bursting Architecture

- **Purpose**
  - To dynamically scale out or "**burst out**" on-premise IT resources into a cloud whenever predefined capacity thresholds have been reached
- **Cloud bursting architecture**
  - The duplicated system of a on-premise system pre-deployed redundantly in a cloud environment while remaining inactive until **cloud bursting** occurs
  - **Resource replication** mechanism and **automated scaling listener** activated to synchronize on-premise and cloud-based systems and to monitor real-time service traffics for possible re-direction
  - Requests to the service **re-directed to** the cloud-based system when the predefined capacity thresholds have been reached by automated scaling listener (**burst out**) and **directed back to** the on-premise system when the service load goes under the predefined capacity thresholds (**burst in**)
  - Flexible scaling architecture for cloud-based IT resources to be used as a **contingency backup** or **standby system** in preparation for workload bursting
- **Consideration**
  - **Replication overhead** – possible performance degradation due to resource replication, especially with low possibility of workload bursting
  - **Synchronization overhead** – drastic performance degradation due to bi-directional resource synchronization when a part of service requests are redirected to the cloud-based IT system – highly recommended for WORK workload only

Cloud bursting architecture

The *cloud bursting architecture* establishes a form of dynamic scaling that scales or "bursts out" on-premise IT resources into a cloud whenever predefined capacity thresholds have been reached. The corresponding cloud-based IT resources are redundantly pre-deployed but remain inactive until cloud bursting occurs. After they are no longer required, the cloud-based IT resources are released and the architecture "bursts in" back to the on-premise environment.

Cloud bursting is a flexible scaling architecture that provides cloud consumers with the option of using cloud-based IT resources only to meet higher usage demands. The foundation of this architectural model is based on the automated scaling listener and resource replication mechanisms.



The automated scaling listener determines when to redirect requests to cloud-based IT resources, and resource replication is used to maintain synchronicity between on-premise and cloud-based IT resources in relation to state information

An automated scaling listener monitors the usage of on-premise Service A, and redirects Service Consumer C's request to Service A's redundant implementation in the cloud (Cloud Service A) once Service A's usage threshold has been exceeded (1). A resource replication system is used to keep state management databases synchronized (2).

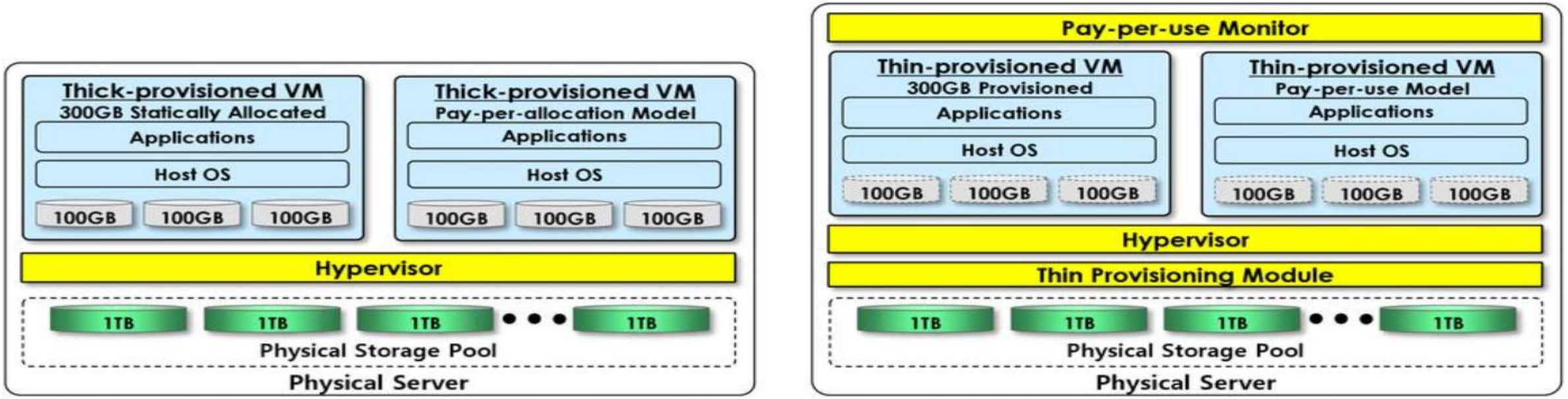
Automated Scaling Listener

- The automated scaling listener mechanism is a service agent that monitors and tracks communications between cloud service consumers and cloud services for dynamic scaling purposes. Automated scaling listeners are deployed within the cloud, typically near the firewall, from where they automatically track workload status information.
- Workloads can be determined by the volume of cloud consumer-generated requests or via back-end processing demands triggered by certain types of requests. For example, a small amount of incoming data can result in a large amount of processing.

- Automated scaling listeners can provide different types of responses to workload fluctuation conditions, such as:
- Automatically scaling IT resources out or in based on parameters previously defined by the cloud consumer (commonly referred to as auto-scaling).
- Automatic notification of the cloud consumer when workloads exceed current thresholds or fall below allocated resources. This way, the cloud consumer can choose to adjust its current IT resource allocation.
- Different cloud provider vendors have different names for service agents that act as automated scaling listeners.

Elastic Disk Provisioning Architecture

- **Purpose**
  - To dynamically provision storage system in order to ensure that the cloud consumer is granularly billed for the exact amount of storage actually in use
- **Elastic disk provisioning architecture**
  - Thin-provisioning technology for the dynamic allocation of storage space and runtime usage monitoring system for collecting accurate usage data for billing purpose
  - **Thin-provisioning** software module on virtual servers to process dynamic storage allocation via the hypervisor
  - **Pay-per-use monitor** to keep track of and to report granular billing-related disk usage data
  - **Resource replication** for possible conversion of dynamic thin-disk storage into static thick-disk storage



Elastic Disk Provisioning Architecture

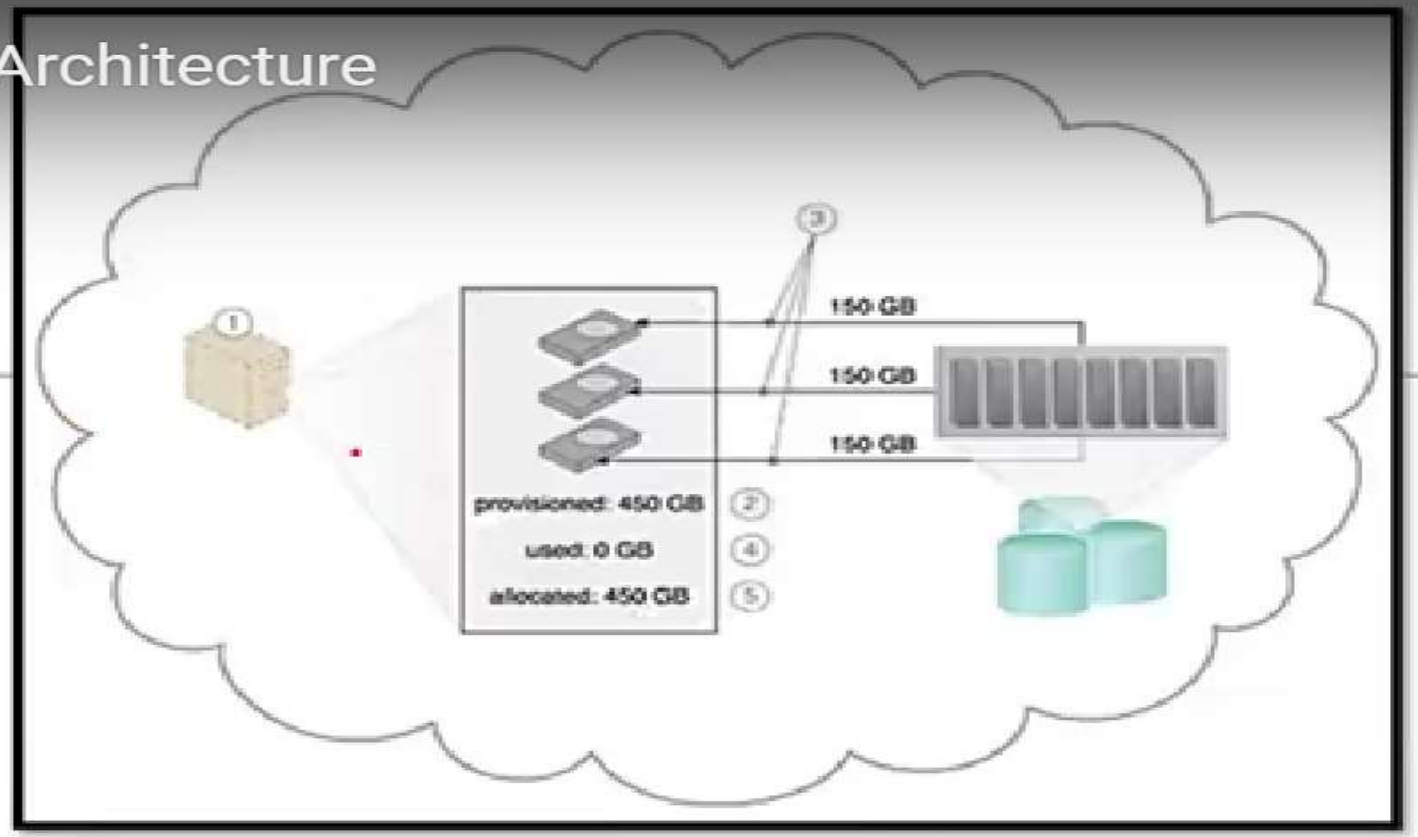


Figure 1. The cloud consumer requests a virtual server with three hard dis
a capacity of 150 GB.

(1)The virtual server is provisioned according to the elastic disk architecture, with a total of 450 GB of disk space.

(2)The 450 GB is allocated to the virtual server by the cloud provider.

(3)The cloud consumer has not installed any software yet, meaning the space is currently 0 GB.

(4)Because the 450 GB are already allocated and reserved for the cloud con will be charged for 450 GB of disk usage as of the point of allocation (5).

Elastic Disk Provisioning Architecture

- The elastic disk provisioning architecture establishes a **dynamic storag provisioning** system that ensures that the cloud consumer is granular billed for the exact amount of storage that it actually uses.
- This system uses **thin-provisioning** technology for the dynamic allocatio of storage space, and is further supported by runtime usage monitoring t collect accurate usage data for billing purposes (Figure 2).
- Thin-provisioning software is **installed** on **virtual servers** that proces dynamic storage allocation via the hypervisor, while the **pay-per-us** monitor tracks and reports granular billing-related disk usage data.

sk Provisioning Architecture

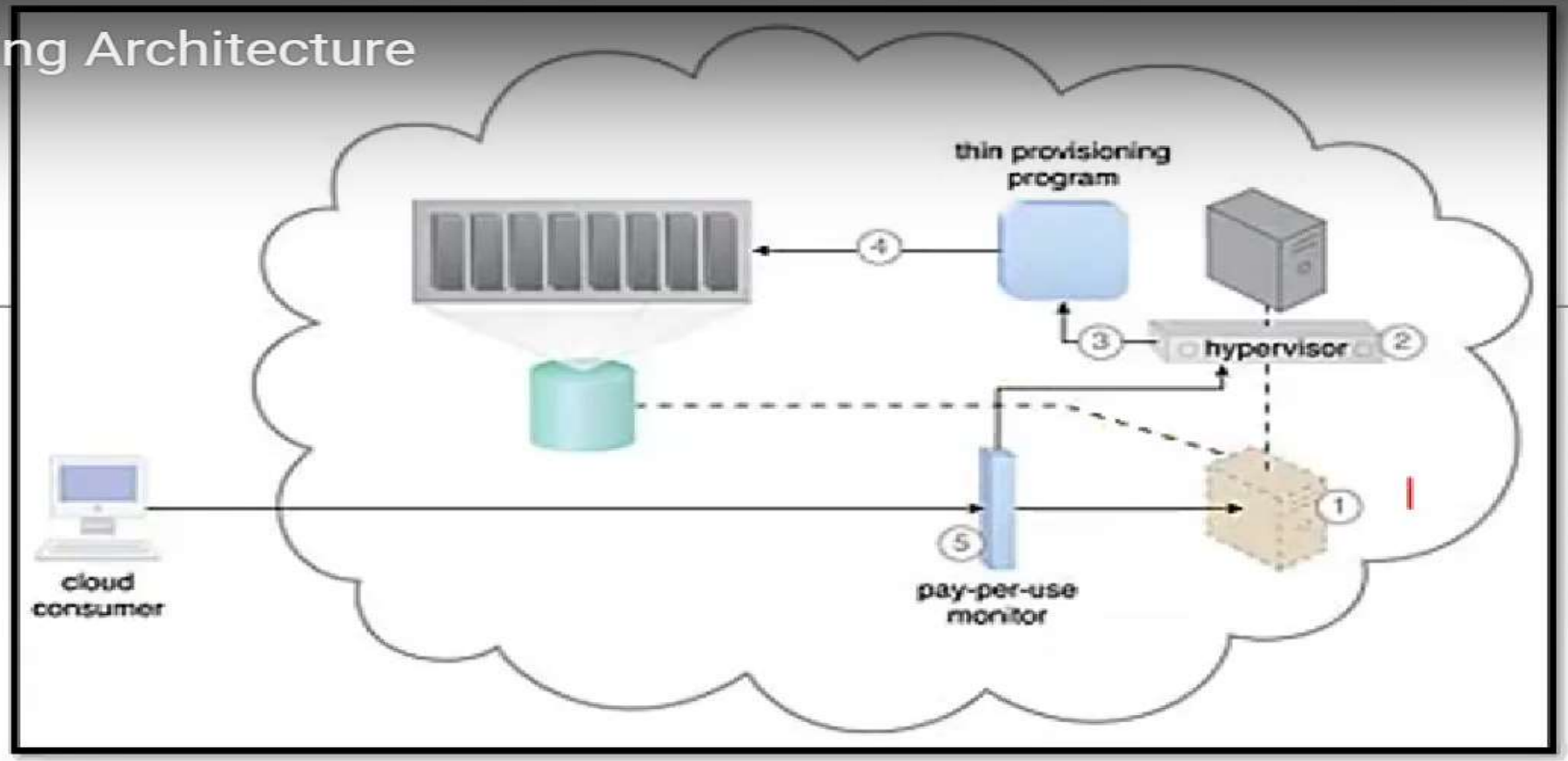


Figure 2. A request is received from a cloud consumer, and the provisioning of a n
virtual server instance begins. (1). As part of the provisioning process, the hard di
are chosen as dynamic or thin-provisioned disks (2). The hypervisor calls a dynar
disk allocation component to create thin disks for the virtual server (3). Virtual ser
disks are created via the thin-provisioning program and saved in a folder of near-z
size. The size of this folder and its files grow as operating applications are installed a
additional files are copied onto the virtual server (4). The pay-per-use monitor tra
the actual dynamically allocated storage for billing purposes (5).

Elastic Disk Provisioning Architecture

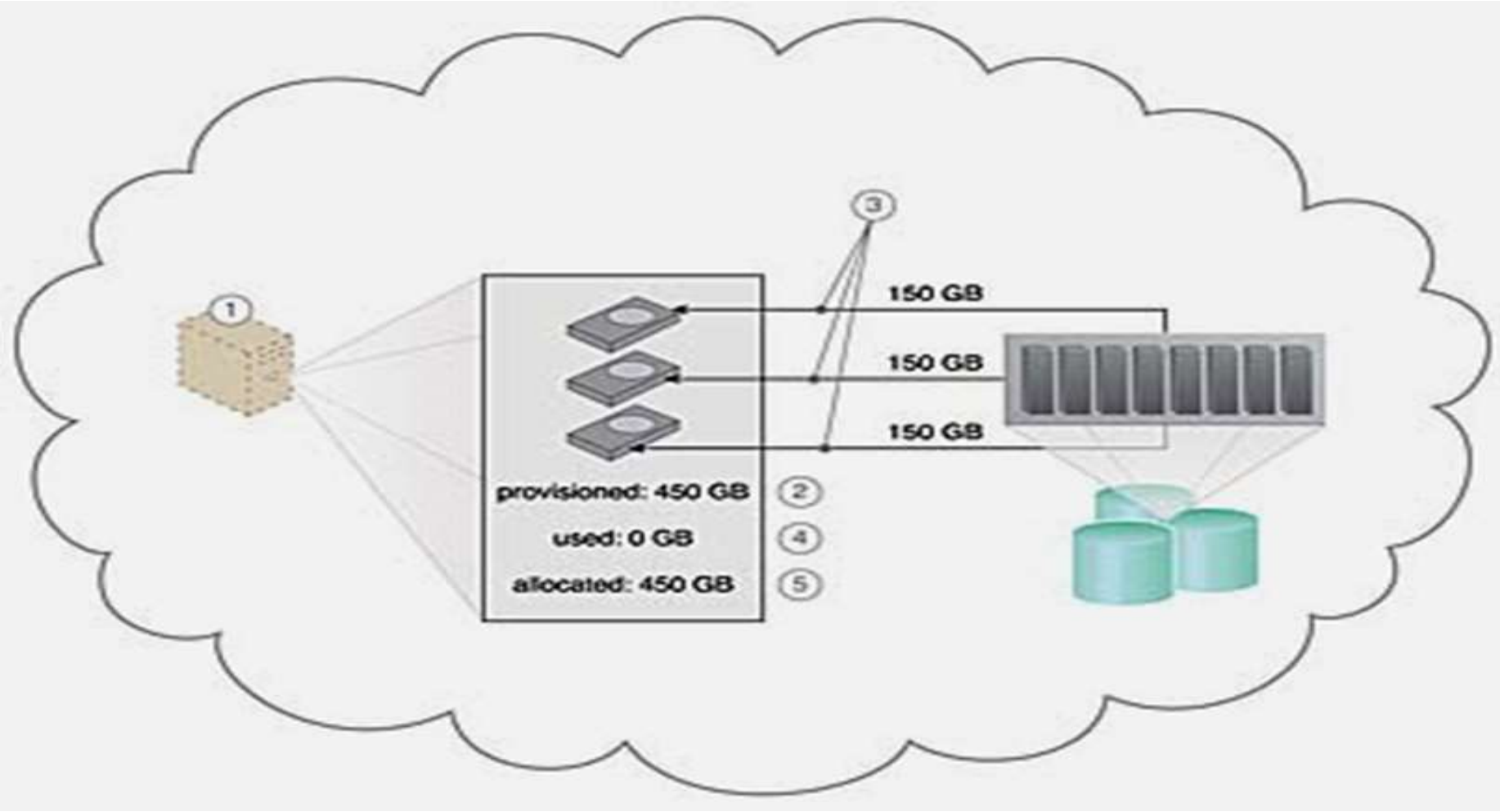
The following mechanisms can be included in this architecture in addition to the cloud storage device, virtual server, hypervisor, and pay-per-use monitor:

1. **Cloud Usage Monitor**: Specialized cloud usage monitors can be us to track and log storage usage fluctuations.
2. **Resource Replication**: Resource replication is part of an elastic d provisioning system when conversion of dynamic thin-disk storage in static thick-disk storage is required.

Elastic Disk Provisioning Architecture

Cloud consumers are commonly charged for cloud-based storage space based on fixed-disk storage allocation, meaning the charges are predetermined by disk capacity and not aligned with actual data storage consumption. Figure demonstrates this by illustrating a scenario in which a cloud consumer provisions a virtual server with the Windows Server operating system and three 150 GB hard drives. The cloud consumer is billed for using 450 GB of storage space after installing the operating system, even though the operating system only requires 15 GB of storage space.

Elastic Disk Provisioning Architecture

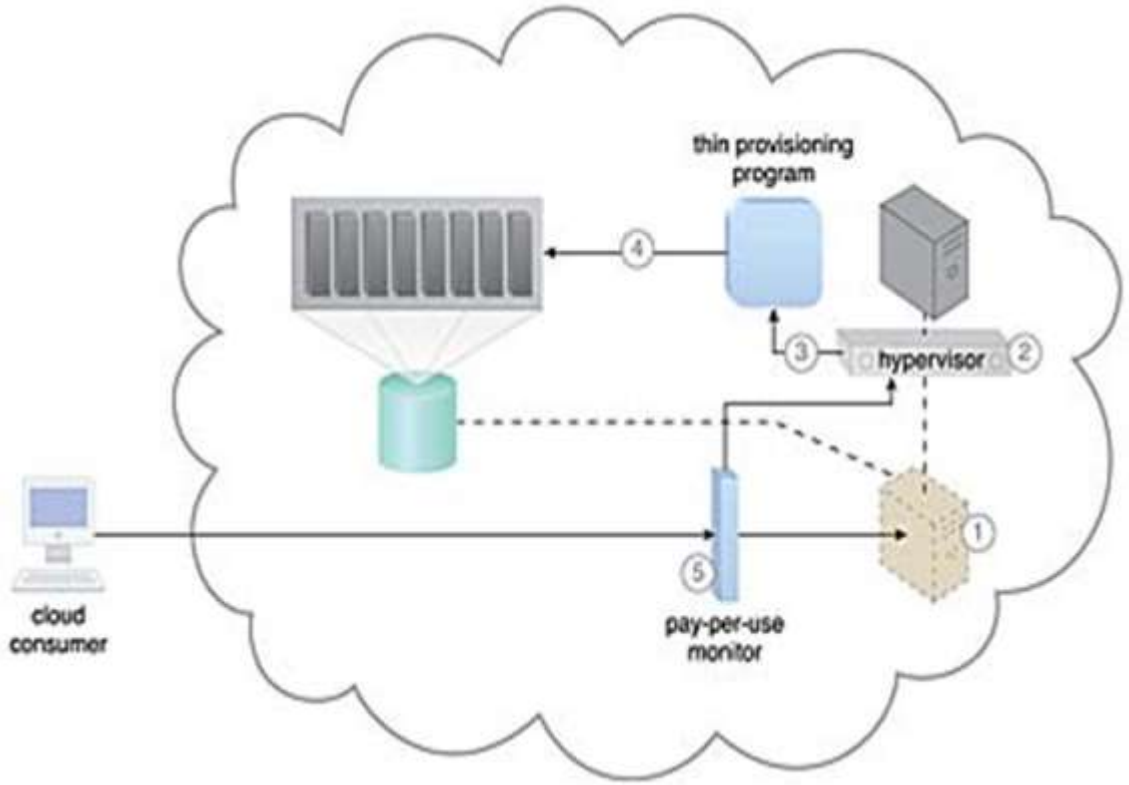


Elastic Disk Provisioning Architecture

The elastic disk provisioning architecture establishes a dynamic storage provisioning system that ensures that the cloud consumer is granularly billed for the exact amount of storage that it actually uses. This system uses thin-provisioning technology for the dynamic allocation of storage space, and is further supported by runtime usage monitoring to collect accurate usage data for billing purposes.

Thin-provisioning software is installed on virtual servers that process dynamic storage allocation via the hypervisor, while the pay-per-use monitor tracks and reports granular billing-related disk usage data.

Elastic Disk Provisioning Architecture



A request is received from a cloud consumer, and the provisioning of a new virtual server instance begins

(1) As part of the provisioning process, the hard disks are chosen as dynamic or thin-provisioned disks.

(2) The hypervisor calls a dynamic disk allocation component to create thin disks for the virtual server.

(3) Virtual server disks are created via the thin-provisioning program and saved in a folder of near-zero size.

(4)The size of this folder and its files grow as operating applications are installed and additional files are copied onto the virtual server.

(5) The pay-per-use monitor tracks the actual dynamically allocated storage for billing purposes.

The following mechanisms can be included in this architecture in addition to the cloud storage device, virtual server, hypervisor, and pay-per-use monitor:

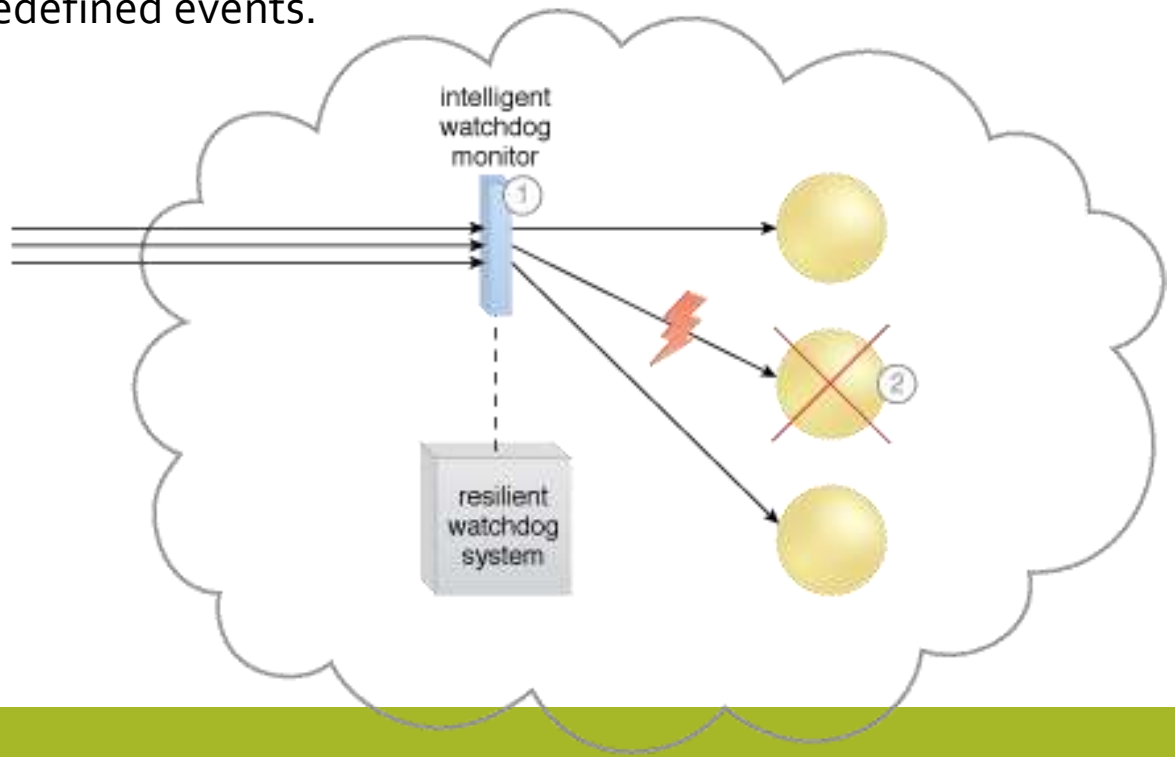
Cloud Usage Monitor – Specialized cloud usage monitors can be used to track and log storage usage fluctuations.

Resource Replication – Resource replication is part of an elastic disk provisioning system when conversion of dynamic thin-disk storage into static thick-disk storage is required.

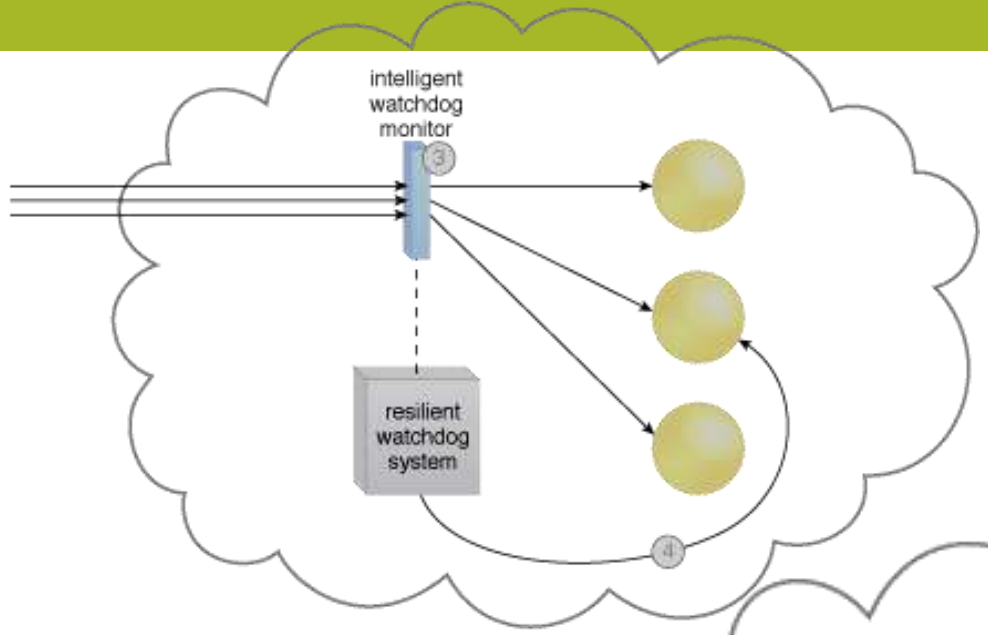
Dynamic Failure Detection and Recovery Architecture

Cloud-based environments can be comprised of vast quantities of IT resources that are simultaneously accessed by numerous cloud consumers. Any of those IT resources can experience failure conditions that require more than manual intervention to resolve. Manually administering and solving IT resource failures is generally inefficient.

The dynamic failure detection and recovery architecture establishes a resilient watchdog system to monitor and respond to a wide range of pre-defined failure scenarios. This system notifies and escalates the failure conditions that it cannot automatically resolve itself. It relies on specialized cloud storage usage monitor called the intelligent watchdog monitor to actively track IT resources and take pre-defined tasks and actions to predefined events.

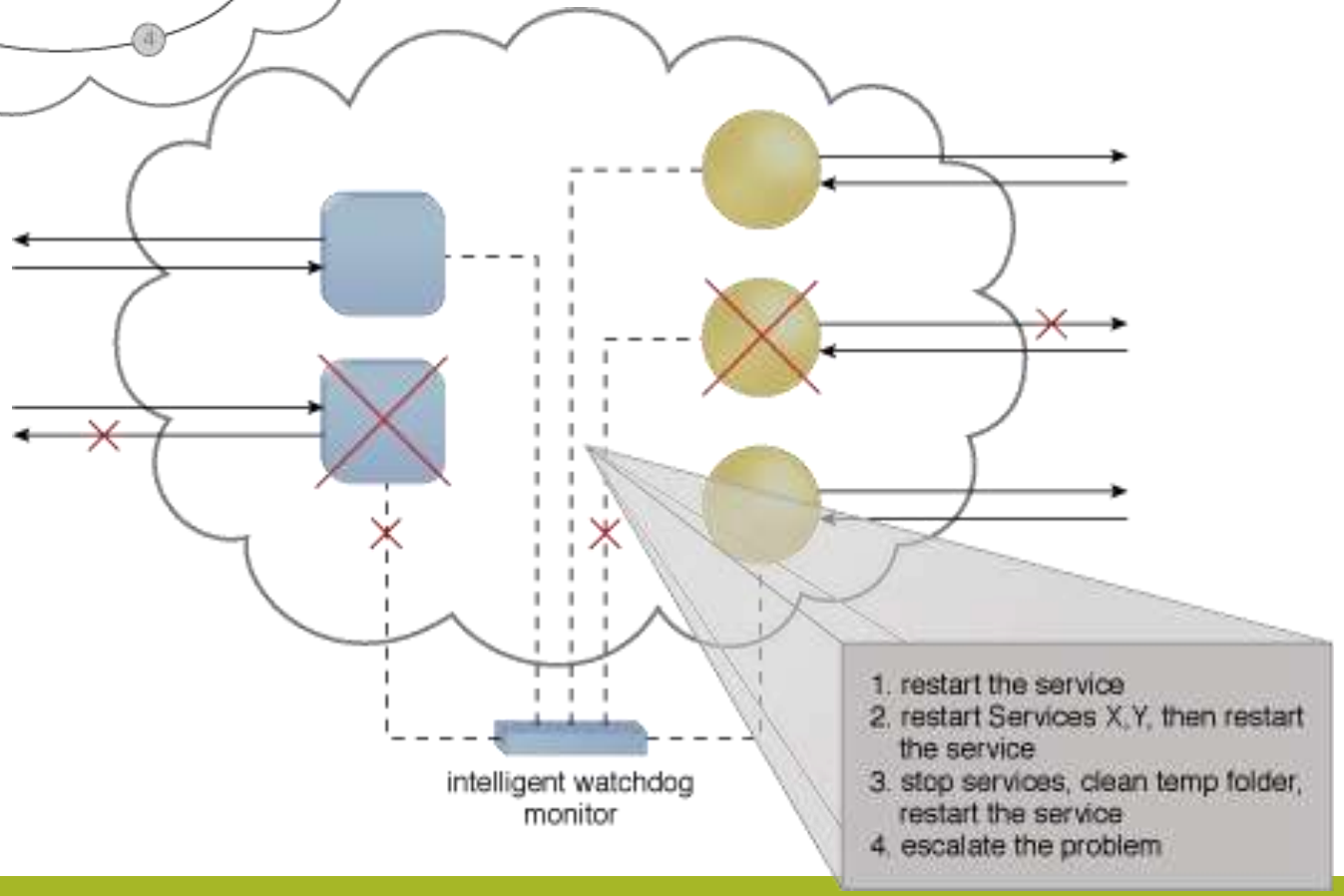


(1)The intelligent watchdog monitor keeps track of cloud consumer requests,
(2) and detects that a cloud service has failed.



The intelligent watchdog monitor notifies the resilient watchdog system (3), which restores the cloud service based on predefined policies (4).

In the event of any failures, the active monitor refers to its predefined policies to recover the service step by step, escalating the processes as the problem proves to be deeper than expected.



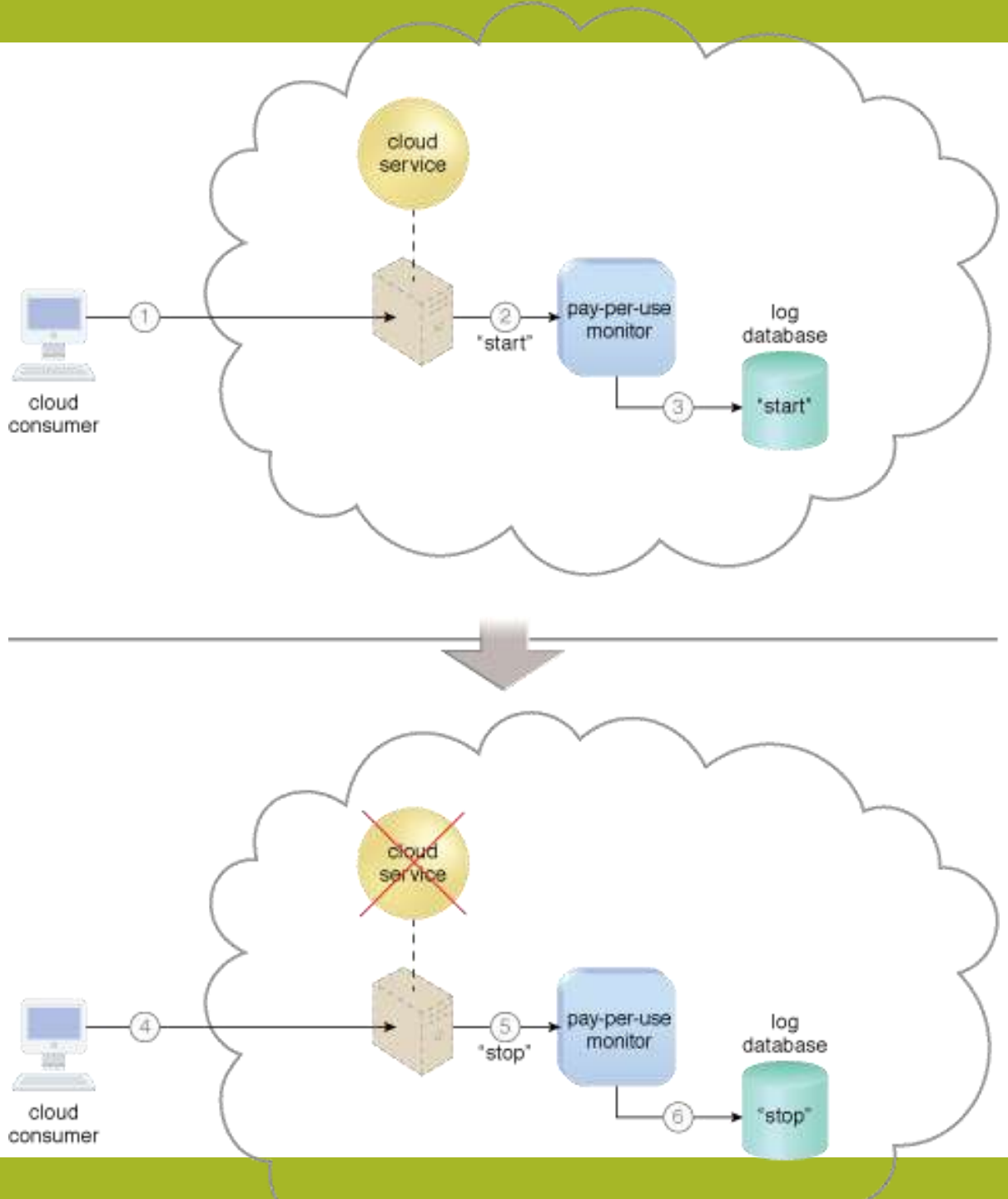
Pay-Per-Use Monitor

The pay-per-use monitor mechanism measures cloud-based IT resource usage in accordance with predefined pricing parameters and generates usage logs for fee calculations and billing purposes.

Some typical monitoring variables are:-

- request/response message quantity
- transmitted data volume
- bandwidth consumption

The data collected by the pay-per-use monitor is processed by a billing management system that calculates the payment fees.



1) A cloud consumer requests the creation of a new instance of a cloud service
2) The IT resource is instantiated and they pay-per-use monitor mechanism receives a "start" event notification from the resource software
3) The pay-per-use monitor stores the value timestamp in the log database
4) The cloud consumer later requests that the cloud service instance be stopped
5) The pay-per-use monitor receives a "stop" event notification from the resource software and stores the value timestamp in the log database

There are varieties of programs and products that can act as an intelligent watchdog monitor. Most can be integrated with standard ticketing and event management systems.

Mechanisms

Audit Monitor – This mechanism may be required to ensure that the manner in which this pattern is carried out at runtime is in compliance with any related legal or policy requirements.

Cloud Usage Monitor – Various specialized cloud usage monitors may be involved with monitoring and collecting IT resource usage data as part of failure conditions and recovery, notification, and escalation activity.

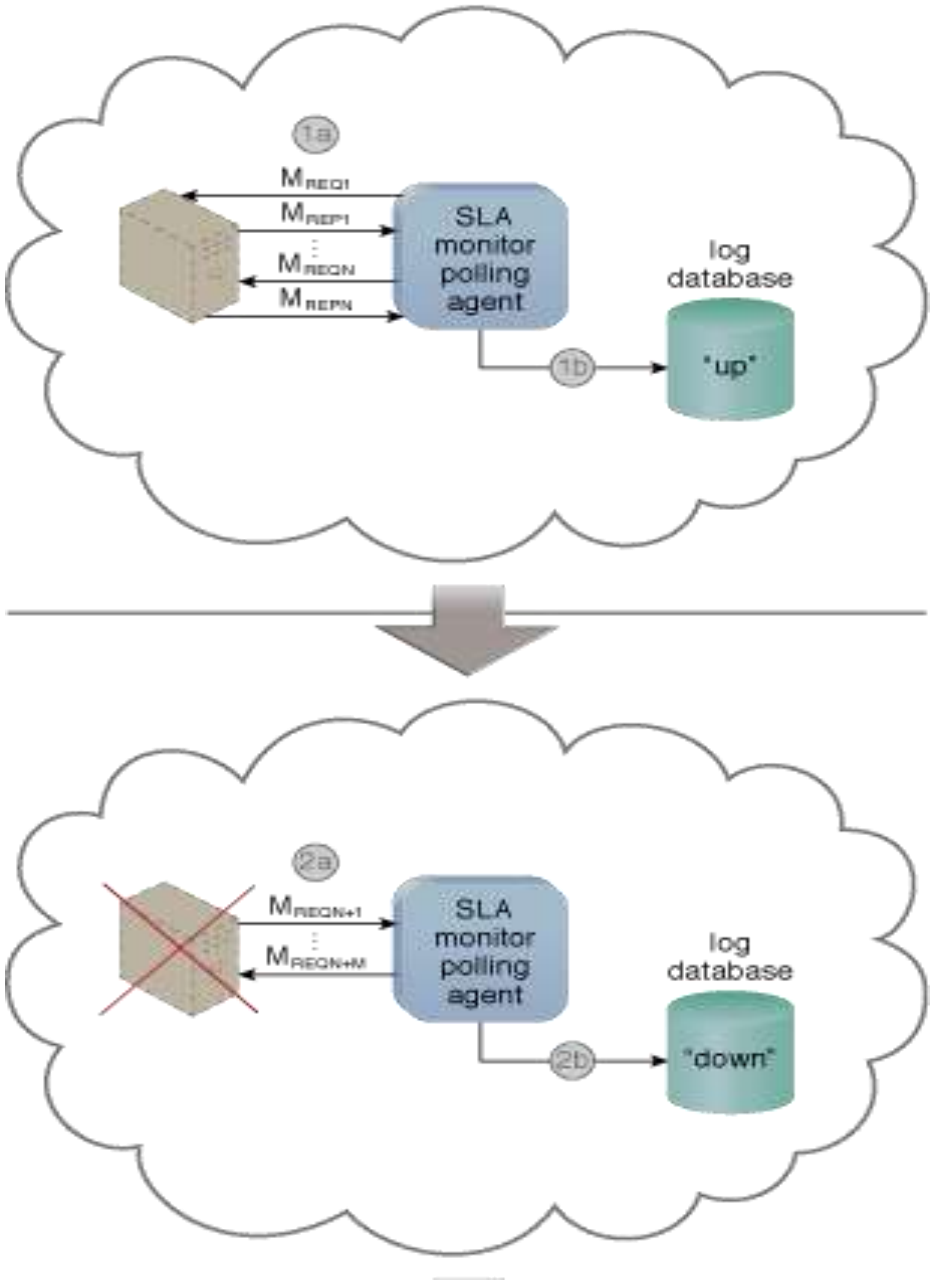
Failover System – Failover is fundamental to the application of this pattern, as the failover system mechanism is generally utilized during the initial attempts to recover failed IT resources.

SLA Management System and *SLA Monitor* – The functionality introduced by the application of the Dynamic Failure Detection and Recovery pattern is closely associated with SLA guarantees and therefore commonly relies on the information managed and processed by these mechanisms.

SLA Management System

The SLA management system mechanism represents a range of commercially available cloud management products that provide features pertaining to the administration, collection, storage, reporting, and runtime notification of SLA data.

An SLA management system deployment will generally include a repository used to store and retrieve collected SLA data based on pre-defined metrics and reporting parameters. It will further rely on one or more SLA monitor mechanisms to collect the SLA data that can then be made available in near-realtime to usage and administration portals to provide ongoing feedback regarding active cloud services (Figure 1). The metrics monitored for individual cloud services are aligned with the SLA guarantees in corresponding cloud provisioning contracts.



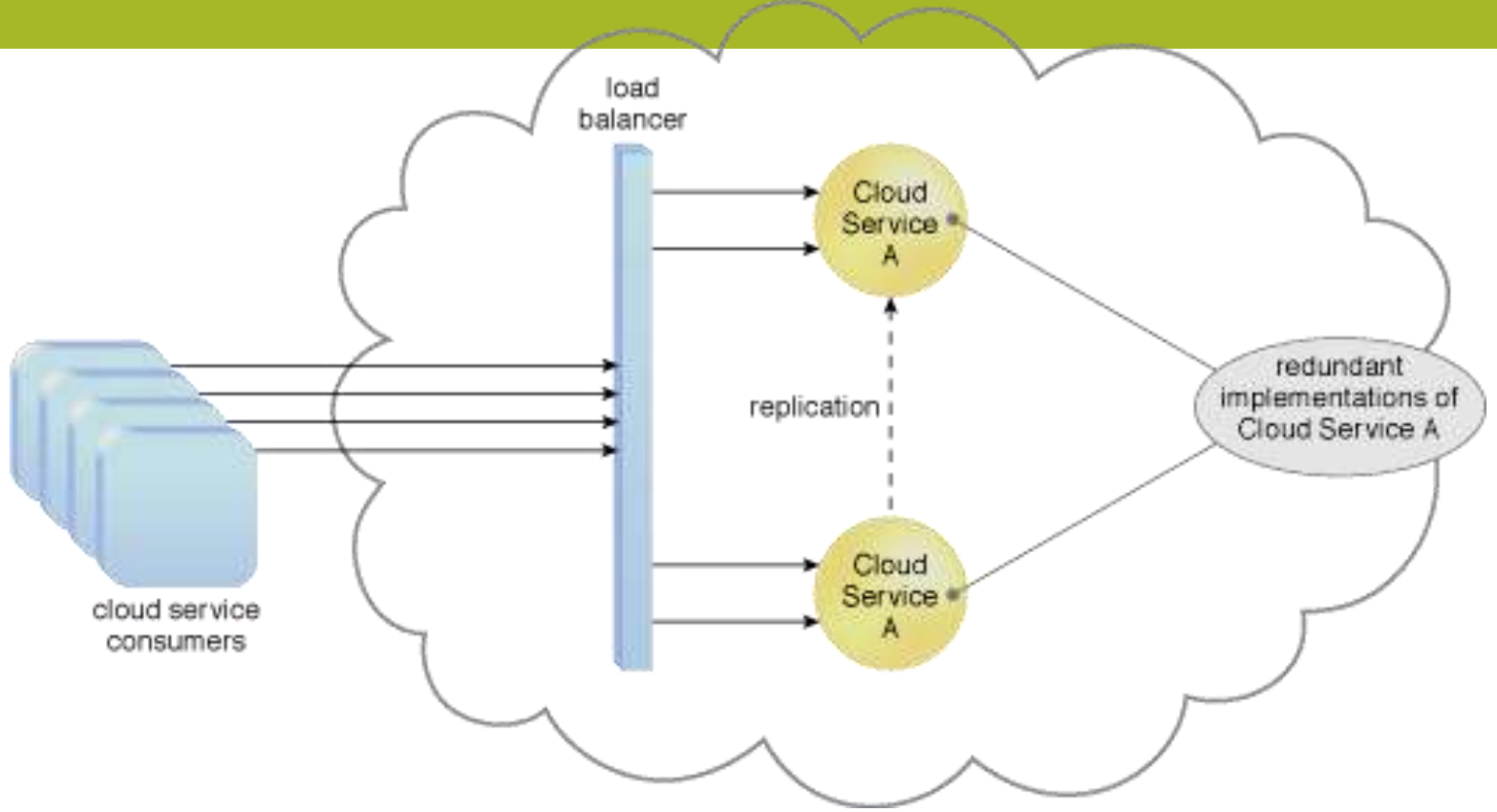
A cloud service consumer interacts with a cloud service (1).
An SLA monitor intercepts the exchanged messages, evaluates the interaction, and collects relevant runtime data in relation to quality-of-service guarantees defined in the cloud service's SLA (2A).
The data collected is stored in a repository (2B) that is part of the SLA management system (3).
Queries can be issued and reports can be generated for an external cloud resource administrator via a usage and administration portal (4) or for an internal cloud resource administrator via the SLA management system's native user-interface (5).

The SLA monitor mechanism is used to specifically observe the runtime performance of cloud services to ensure that they are fulfilling the contractual QoS requirements published in SLAs (Figure 1). The data collected by the SLA monitor is processed by an SLA management system to be aggregated into SLA reporting metrics. This system can proactively repair or failover cloud services when exception conditions occur, such as when the SLA monitor reports a cloud service as “down.”

Load Balancer

The load balancer mechanism is a runtime agent with logic fundamentally based on the premise of employing horizontal scaling to balance a workload across two or more IT resources to increase performance and capacity beyond what a single IT resource can provide. Beyond simple division of labor algorithms, load balancers can perform a range of specialized runtime workload distribution functions that include:

- *Asymmetric Distribution* – larger workloads are issued to IT resources with higher processing capacities
- *Workload Prioritization* – workloads are scheduled, queued, discarded, and distributed workloads according to their priority levels
- *Content-Aware Distribution* – requests are distributed to different IT resources as dictated by the request content



- A load balancer implemented as a service agent transparently distributes incoming workload request messages across two redundant cloud service implementations, which in turn maximizes performance for the clouds service consumers.
- A load balancer is programmed or configured with a set of performance and QoS rules and parameters with the general objectives of optimizing IT resource usage, avoiding overloads, and maximizing throughput.

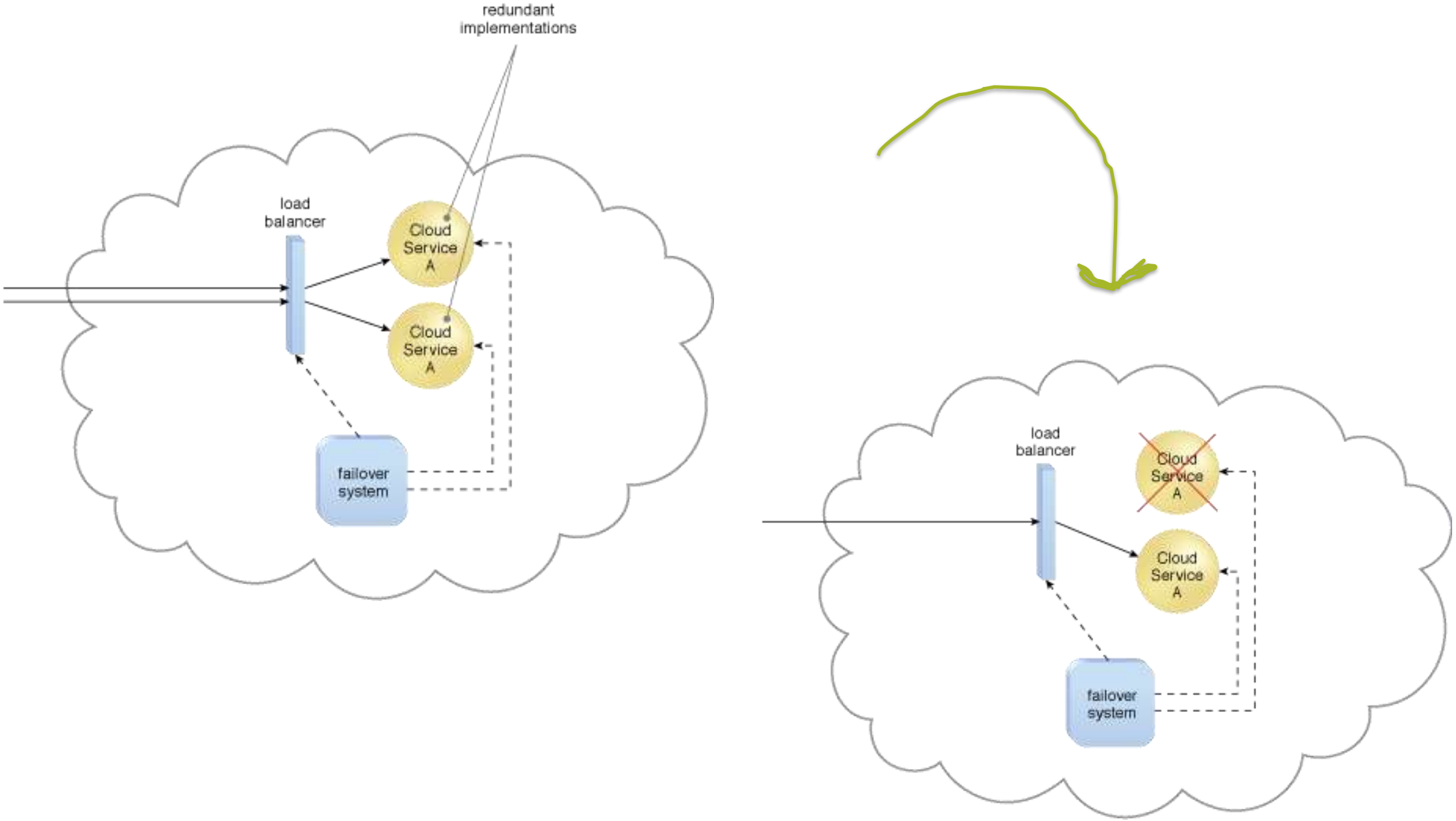
The load balancer mechanisms can exist as a:

- multi-layer network switch
- dedicated hardware appliance
- dedicated software-based system (common in server operating systems)
- service agent (usually controlled by cloud management software)

The load balancer is typically located on the communication path between the IT resources generating the workload and the IT resources performing the workload processing. This mechanism can be designed as a transparent agent that remains hidden from the cloud service consumers, or as a proxy component that abstracts the IT resources performing their workload.

Fail over systems

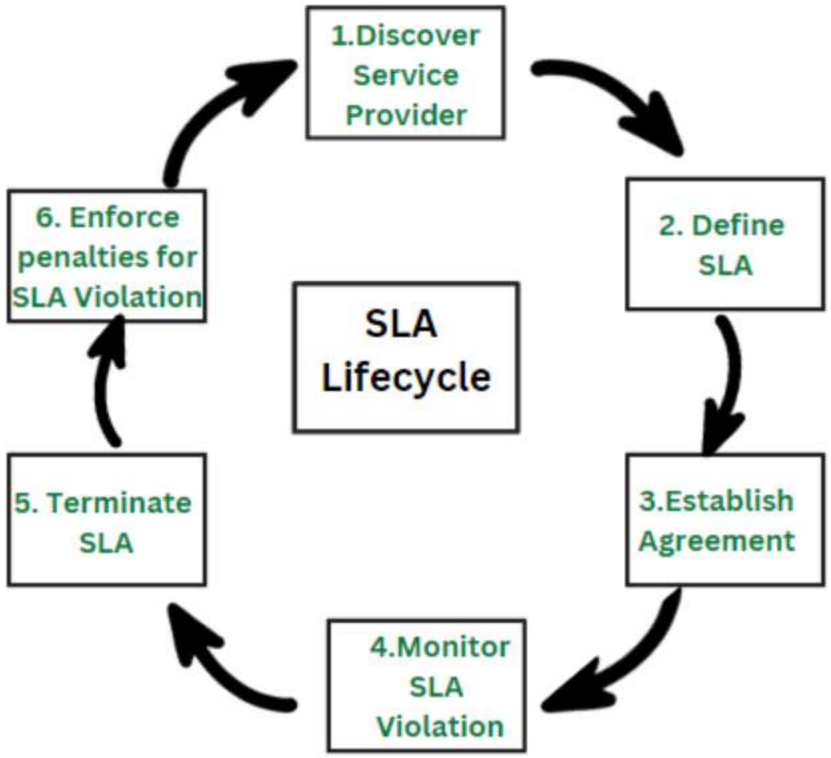
- The failover system mechanism is used to increase the reliability and availability of IT resources by using established clustering technology to provide redundant implementations.
- A failover system is configured to automatically switch over to a redundant or standby IT resource instance whenever the currently active IT resource becomes unavailable.
- A failover system can span more than one geographical region so that each location hosts one or more redundant implementations of the same IT resource.


redundant
implementations
load
balancer
Cloud
Service
A
Cloud
Service
A
failover
system
load
balancer
Cloud
Service
A
Cloud
Service
A
failover
system

Service level agreements

A Service Level Agreement (SLA) is the bond for performance negotiated between the cloud services provider and the client. Earlier, in cloud computing all Service Level Agreements were negotiated between a client and the service consumer. Nowadays, with the initiation of large utility-like cloud computing providers, most Service Level Agreements are standardized until a client becomes a large consumer of cloud services. Service level agreements are also defined at different levels which are mentioned below:

- Customer-based SLA
- Service-based SLA
- Multi-level SLA


1.Discover Service Provider
2. Define SLA
3.Establish Agreement
4.Monitor SLA Violation
5. Terminate SLA
6. Enforce penalties for SLA Violation
SLA Lifecycle

Few Service Level Agreements are enforceable as contracts, but mostly are agreements or contracts which are more along the lines of an Operating Level Agreement (OLA) and may not have the restriction of law. It is fine to have an attorney review the documents before making a major agreement to the cloud service provider. Service Level Agreements usually specify some parameters which are mentioned below:

- Availability of the Service (uptime)
- Latency or the response time
- Service components reliability
- Each party accountability
- Warranties

In any case, if a cloud service provider fails to meet the stated targets of minimums then the provider has to pay the penalty to the cloud service consumer as per the agreement. So, Service Level Agreements are like insurance policies in which the corporation has to pay as per the agreements if any casualty occurs.

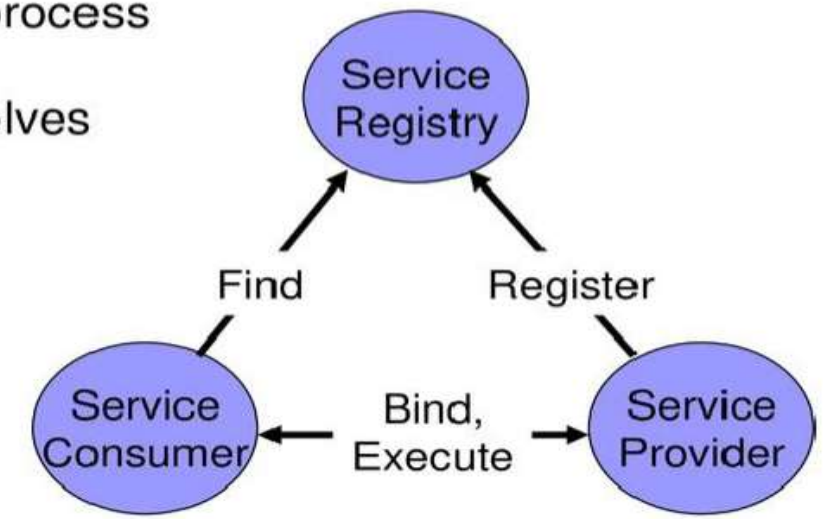
Service Oriented Architecture

- SOA, or service-oriented architecture, defines a way to make software components reusable via service interfaces. These interfaces utilize common communication standards in such a way that they can be rapidly incorporated into new applications without having to perform deep integration each time.

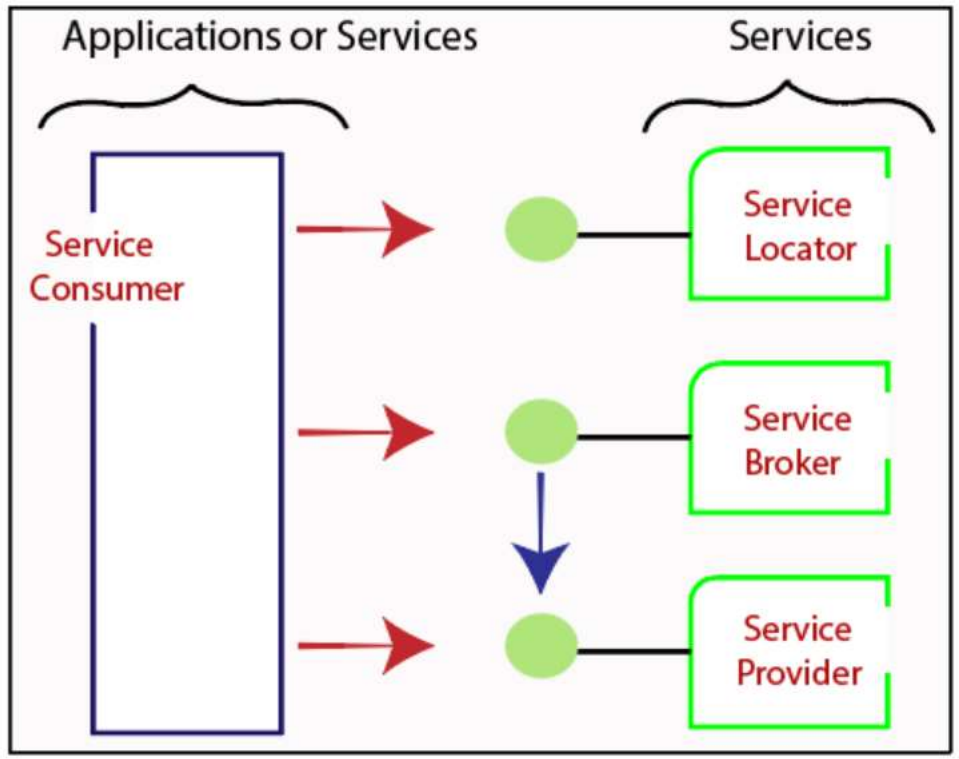
- Each service in an SOA embodies the code and data integrations required to execute a complete, discrete business function (e.g., checking a customer's credit, calculating a monthly loan payment, or processing a mortgage application). The service interfaces provide loose coupling, meaning they can be called with little or no knowledge of how the integration is implemented underneath. The services are exposed using standard network protocols—such as SOAP (simple object access protocol)/HTTP or JSON/HTTP—to send requests to read or change data. The services are published in a way that enables developers to quickly find them and reuse them to assemble new applications.

What is Service Oriented Architecture (SOA)?

- An SOA application is a composition of services
- A "service" is the atomic unit of an SOA
- Services encapsulate a business process
- Service Providers Register themselves
- Service use involves: Find, Bind, Execute
- Most well-known instance is Web Services



Service-oriented architecture (SOA) is a method of software development that uses software components called services to create business applications. Each service provides a business capability, and services can also communicate with each other across platforms and languages. Developers use SOA to reuse services in different systems or combine several independent services to perform complex tasks.



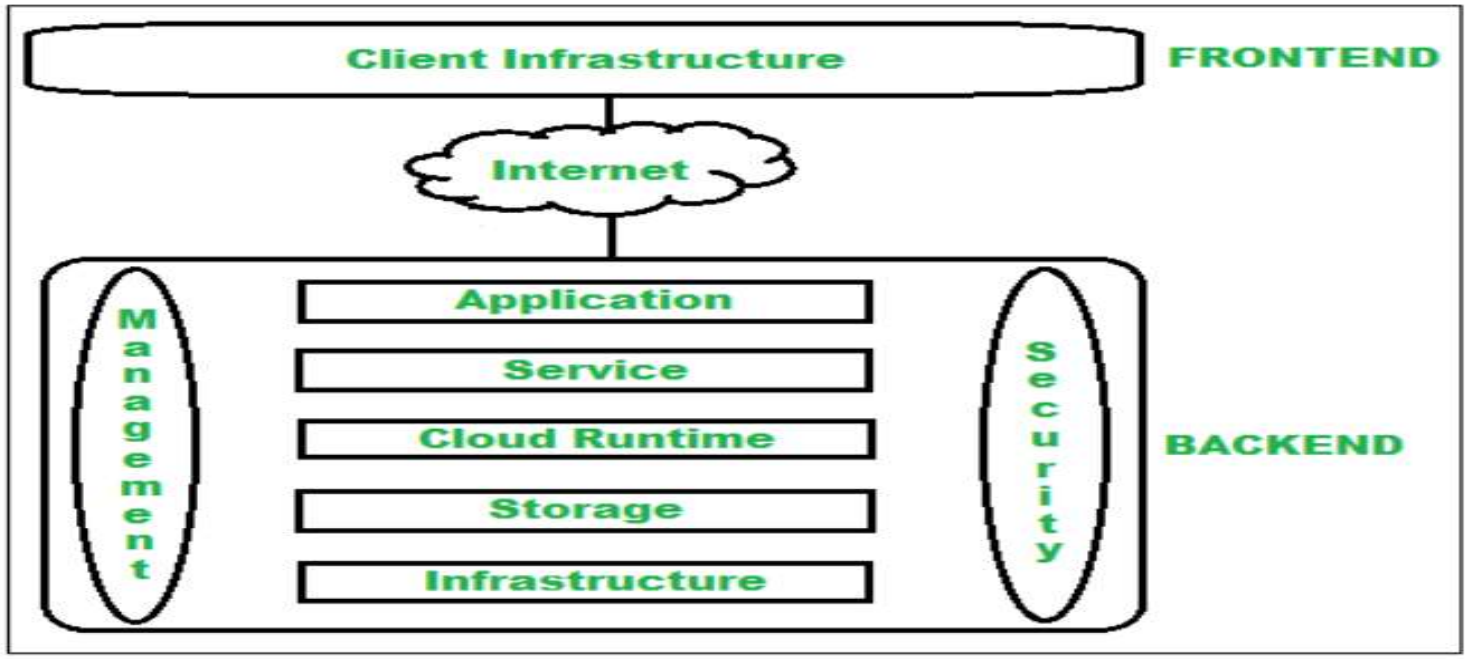
Benefits of SOA

- Greater business agility; faster time to market: The efficiency of assembling applications from reusable service interfaces, rather than rewriting and reintegrating with every new development project, enables developers to build applications much more quickly in response to new business opportunities.
- Ability to leverage legacy functionality in new markets: A well-crafted SOA enables developers to easily take functionality ‘locked’ in one computing platform or environment and extend it to new environments and markets. For example, many companies have used SOA to expose functionality from mainframe-based financial systems to the web, enabling their customers to serve themselves to processes and information previously accessible only through direct interaction with the company’s employees or business partners.
- Improved collaboration between business and IT: In an SOA, services can be defined in business terms (e.g., ‘generate insurance quote’ or ‘calculate capital equipment ROI’). This enables business analysts to work more effectively with developers on important insights—such as the scope of a business process defined by a service or the business implications of changing a process—that can lead to a better result.

Cloud Computing Architecture

The cloud architecture is divided into 2 parts i.e.

1.Frontend 2. Backend



Frontend

- Frontend of the cloud architecture refers to the client side of cloud computing system. Means it contains all the user interfaces and applications which are used by the client to access the cloud computing services/resources.
- **Client Infrastructure –** Client Infrastructure is a part of the frontend component. It contains the applications and user interfaces which are required to access the cloud platform.
- In other words, it provides a GUI(Graphical User Interface) to interact with the cloud.

Backend

- Backend refers to the cloud itself which is used by the service provider. It contains the resources as well as manages the resources and provides security mechanisms.

1. **Application –**
   Application in backend refers to a software or platform to which client accesses. Means it provides the service in backend as per the client requirement.
2. **Service –**
   Service in backend refers to the major three types of cloud based services like SaaS, PaaS and IaaS. Also manages which type of service the user accesses.
3. **Runtime Cloud-**
   Runtime cloud in backend provides the execution and Runtime platform/environment to the Virtual machine.

Backend contd..

4.Storage –
Storage in backend provides flexible and scalable storage service and management of stored data.

5.Infrastructure –
Cloud Infrastructure in backend refers to the hardware and software components of cloud like it includes servers, storage, network devices, virtualization software etc.

6.Management –
Management in backend refers to management of backend components like application, service, runtime cloud, storage, infrastructure, and other security mechanisms etc.

7.Security –
Security in backend refers to implementation of different security mechanisms in the backend for secure cloud resources, systems, files, and infrastructure to end-users.

8.**Internet –**
Internet connection acts as the medium or a bridge between frontend and backend and establishes the interaction and communication between frontend and backend.

Benefits of Cloud Computing Arch.

- Makes overall cloud computing system simpler.
- Improves data processing requirements.
- Helps in providing high security.
- Makes it more modularized.
- Results in better disaster recovery.
- Gives good user accessibility.
- Reduces IT operating costs.

Defining data migration: What is data migration?

In general, data migration means moving digital information. Transferring that information to a different location, file format, environment, storage system, database, datacenter, or application all fit within the definition of data migration.

To define data migration more specifically:

Data migration is the process of selecting, preparing, extracting, and transforming data and permanently transferring it from one computer storage system to another.

Data migration is a common IT activity. However, data assets may exist in many different states and locations, which makes some migration projects more complex and technically challenging than others. Examples of data assets include:

- Unorganized assortments of files stored across many different devices.
- Applications, operating systems, and environments.
- Relational databases like SQL Server, MySQL, PostgreSQL, and MariaDB.
- Unstructured databases such as MongoDB, Azure Cosmos DB, DocumentDB, Cassandra, Couchbase, HBase, Redis, and Neo4j.
- Data lakes, data blobs, and entire datacenters.

As a result, data migration projects require planning, implementation, and validation to ensure their success. Learn more about cloud migration and other types of migration, here.

Planning a data migration

Before even beginning to gather requirements for and scope a cloud data migration, organizations need to start by discovering and assessing what data they actually have. They must map the data—find out how much of it there is, how diverse it is, and what quality or condition it is in.

They'll likewise assess the impact of the migration on the organization, establish who the stakeholders are and who has relevant expertise, assign responsibilities, set budget and timelines, and agree on how everyone will communicate about the data migration project.

After scoping the project, teams then design the migration, which includes selecting data migration software and hardware they'll use when they move the data, creating specifications for the data migration, and determining the rate at which they will migrate the data: all at once, just a little bit at a time, or anywhere in between. Many organizations seek help and guidance right-sizing their migration—especially when moving to the cloud.

Implementing a data migration

When planning is complete and the migration is designed, teams begin implementation. They build the data migration solution according to the requirements and step-by-step migration guidance set forth in the planning phase and begin transferring the data.

As the data migrates, teams monitor and test it to ensure the data is transferring properly and free of conflicts, data quality problems, duplicates, and anomalies. This monitoring and testing take place in an environment that mirrors the production environment and enables teams to quickly identify and remediate any issues with the data migration.

Validating a data migration

After all the data has been migrated and implementation is complete, teams will audit the data in its new configuration and validate that the data has been transferred accurately. Teams take the old data configuration out of service only after the data migration is validated by technical and business stakeholders as well as anyone else—including customers—who might use the data.

An organization may need to or choose to migrate data for many different reasons. At a high level, these reasons can include reducing costs, enabling innovation, increasing performance, creating higher availability, and strengthening security. As organizations make the decision to migrate data, they'll need to consider the integrity of the data, the cost of migrating, and the impact to the business and its customers.

Some specific scenarios and business cases that may require data migration include:

- Upgrading or replacing legacy hardware or software so the organization can meet its performance requirements or be more competitive.
- Lessening environmental impact—and decreasing operational costs—by moving to a system that has a smaller footprint and uses less energy.
- Reducing or eliminating the expense of hosting the data in on-premises datacenters by migrating to the cloud.
- Centralizing data to enable and facilitate interoperability or relocating to a more secure datacenter.
- Backing up data to allow the organization to better prepare for and execute disaster recovery.

Streaming data is data that is generated continuously by thousands of data sources, which typically send in the data records simultaneously, and in small sizes (order of Kilobytes). Streaming data includes a wide variety of data such as log files generated by customers using your mobile or web applications, ecommerce purchases, in-game player activity, information from social networks, financial trading floors, or geospatial services, and telemetry from connected devices or instrumentation in data centers.

This data needs to be processed sequentially and incrementally on a record-by-record basis or over sliding time windows, and used for a wide variety of analytics including correlations, aggregations, filtering, and sampling. Information derived from such analysis gives companies visibility into many aspects of their business and customer activity such as – service usage (for metering/billing), server activity, website clicks, and geo-location of devices, people, and physical goods –and enables them to respond promptly to emerging situations. For example, businesses can track changes in public sentiment on their brands and products by continuously analyzing social media streams, and respond in a timely fashion as the necessity arises.

Benefits of streaming data

Streaming data processing is beneficial in most scenarios where new, dynamic data is generated on a continual basis. It applies to most of the industry segments and big data use cases. Companies generally begin with simple applications such as collecting system logs and rudimentary processing like rolling min-max computations. Then, these applications evolve to more sophisticated near-real-time processing. Initially, applications may process data streams to produce simple reports, and perform simple actions in response, such as emitting alarms when key measures exceed certain thresholds. Eventually, those applications perform more sophisticated forms of data analysis, like applying machine learning algorithms, and extract deeper insights from the data. Over time, complex, stream and event processing algorithms, like decaying time windows to find the most recent popular movies, are applied, further enriching the insights.

Examples of streaming data

- Sensors in transportation vehicles, industrial equipment, and farm machinery send data to a streaming application. The application monitors performance, detects any potential defects in advance, and places a spare part order automatically preventing equipment down time.
- A financial institution tracks changes in the stock market in real time, computes value-at-risk, and automatically rebalances portfolios based on stock price movements.
- A real-estate website tracks a subset of data from consumers' mobile devices and makes real-time property recommendations of properties to visit based on their geo-location.
- A solar power company has to maintain power throughput for its customers, or pay penalties. It implemented a streaming data application that monitors of all of panels in the field, and schedules service in real time, thereby minimizing the periods of low throughput from each panel and the associated penalty payouts.

- A media publisher streams billions of clickstream records from its online properties, aggregates and enriches the data with demographic information about users, and optimizes content placement on its site, delivering relevancy and better experience to its audience.
- An online gaming company collects streaming data about player-game interactions, and feeds the data into its gaming platform. It then analyzes the data in real-time, offers incentives and dynamic experiences to engage its players.

Comparison between batch processing and stream processing

Before dealing with streaming data, it is worth comparing and contrasting stream processing and batch processing. Batch processing can be used to compute arbitrary queries over different sets of data. It usually computes results that are derived from all the data it encompasses, and enables deep analysis of big data sets. MapReduce-based systems, like Amazon EMR, are examples of platforms that support batch jobs. In contrast, stream processing requires ingesting a sequence of data, and incrementally updating metrics, reports, and summary statistics in response to each arriving data record. It is better suited for real-time monitoring and response