Advanced Cloud Architectures

Presented By,

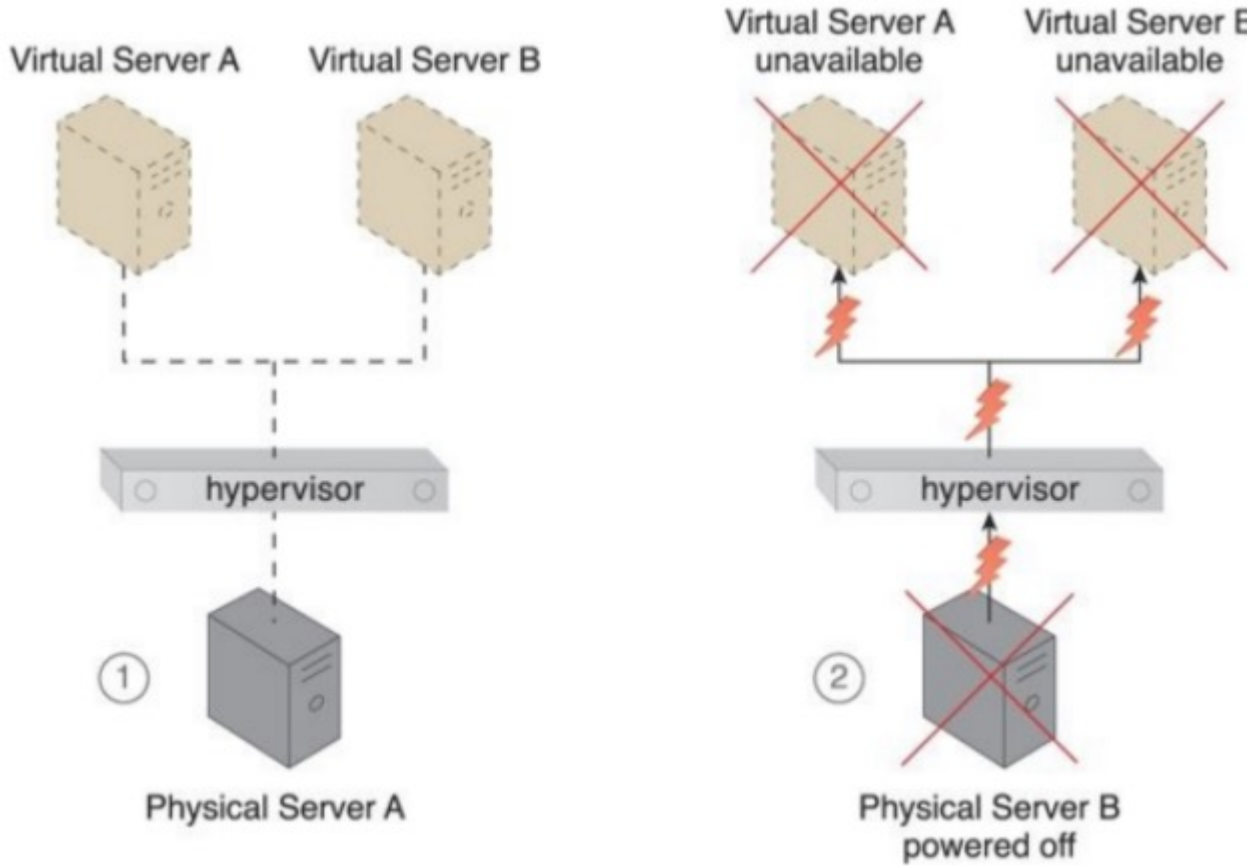
VKP

Contents are from Cloud computing book-Thomas Erl (Chapter:12)

Hypervisor Clustering Architecture

- Hypervisors can be responsible for creating and hosting multiple virtual servers. Because of this dependency, any failure conditions that affect a hypervisor can cascade to its virtual servers.

Physical Server A is hosting a hypervisor that hosts Virtual Servers A and B (1). When Physical Server A fails, the hypervisor and two virtual servers consequently fail as well (2).



Heartbeats

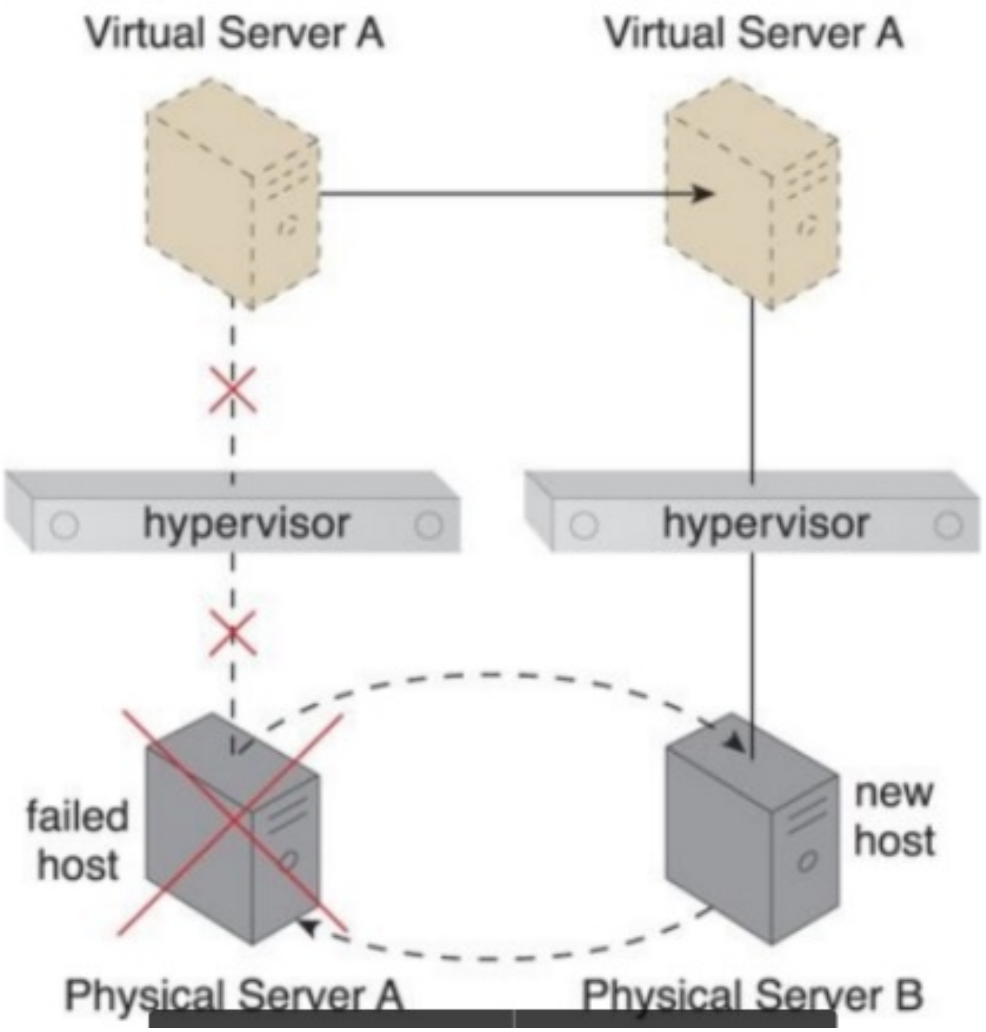
heartbeat

Heartbeats are system-level messages exchanged between hypervisors, hypervisors and virtual servers, and hypervisors and VIMs.

The hypervisor clustering architecture establishes a high-availability cluster of hypervisors across multiple physical servers.

Note:- ***If a given hypervisor or its underlying physical server becomes unavailable, the hosted virtual servers can be moved to another physical server or hypervisor to maintain runtime operations.***

A case study



Physical Server A becomes unavailable and causes its hypervisor to fail. Virtual Server A is migrated to Physical Server B, which has another hypervisor that is part of the cluster to which Physical Server A belongs.

The hypervisor cluster is controlled via a central VIM, which sends regular heartbeat messages to the hypervisors to confirm that they are up and running.

Unacknowledged heartbeat messages cause the VIM to initiate the live VM migration program, in order to dynamically move the affected virtual servers to a new host.

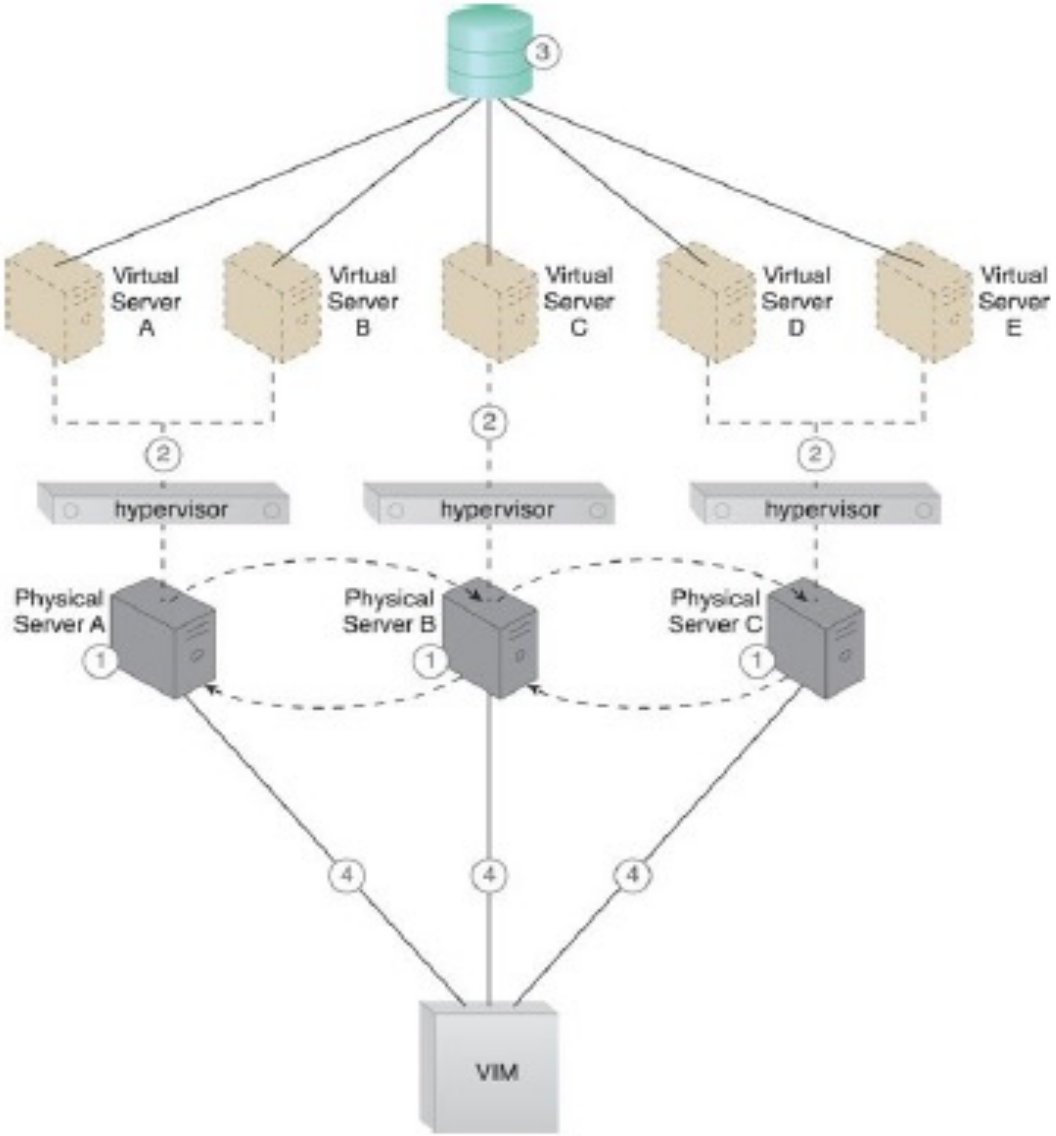


Figure 12.3. Hypervisors are installed on Physical Servers A, B, and C (1). Virtual servers are created by the hypervisors (2). A shared cloud storage device containing virtual server configuration files is positioned in a shared cloud storage device for access by all hypervisors (3). The hypervisor cluster is enabled on the three physical server hosts via a central VIM (4).

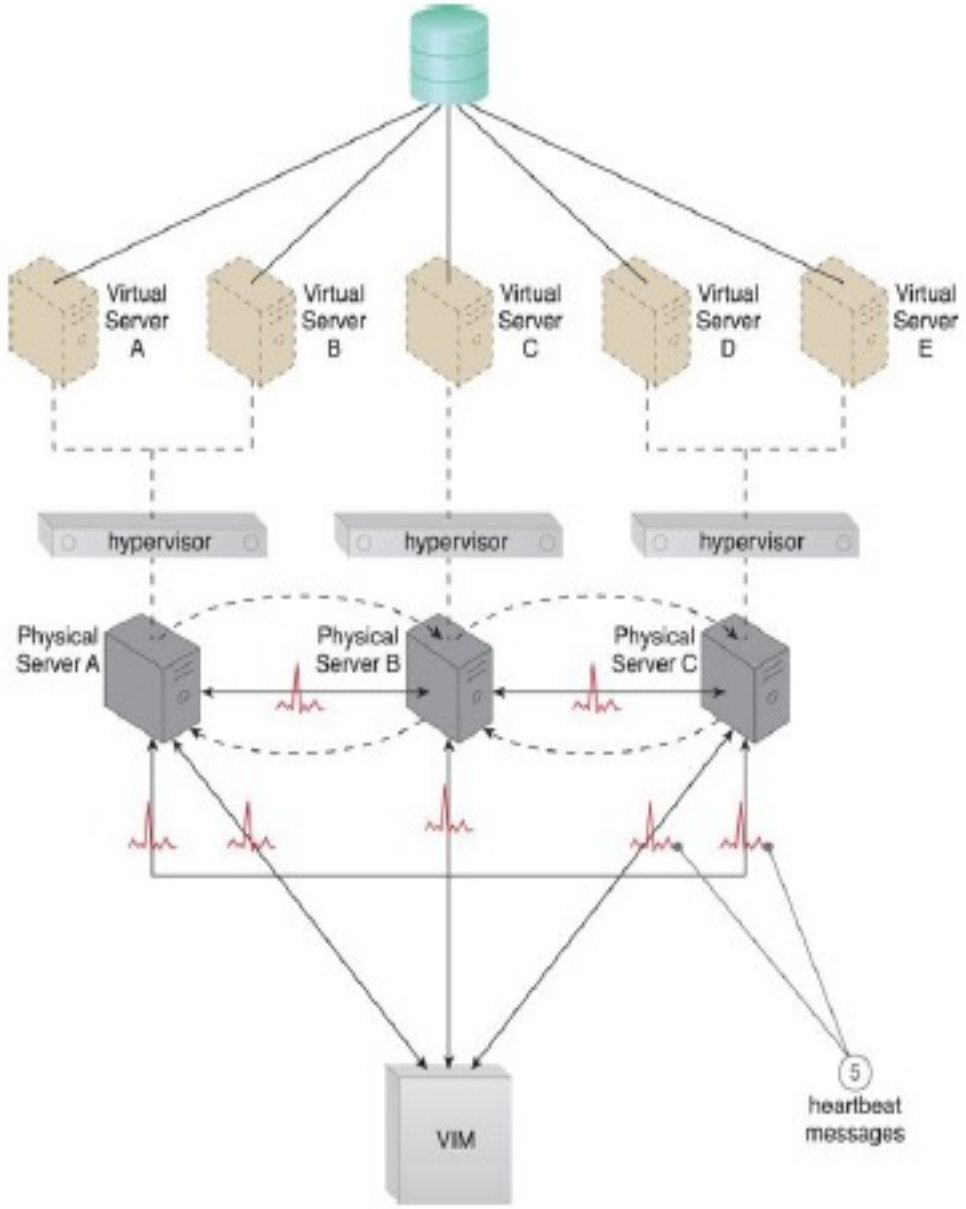


Figure 12.4. The physical servers exchange heartbeat messages with one another and the VIM according to a pre-defined schedule (5).

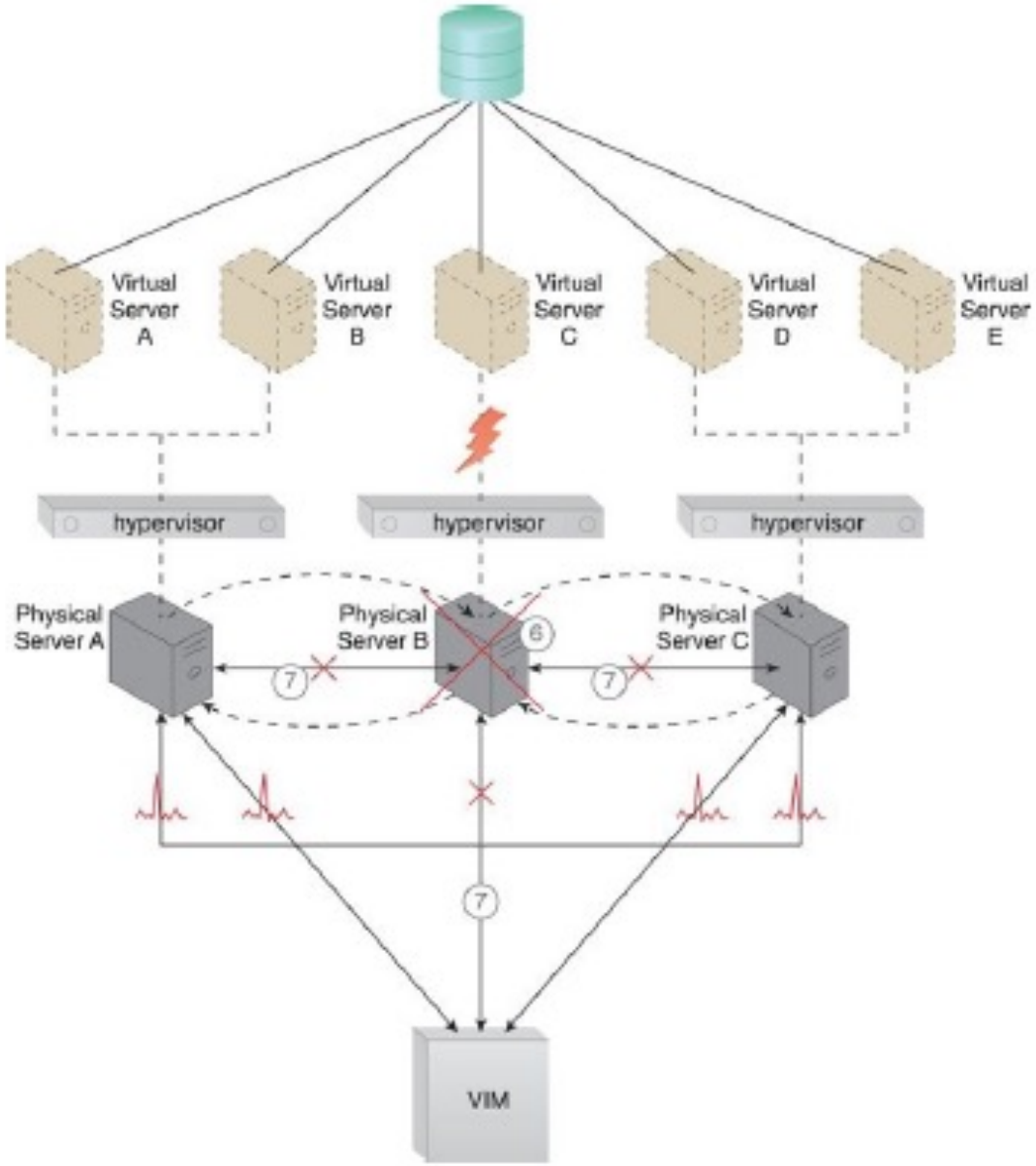


Figure 12.5. Physical Server B fails and becomes unavailable, jeopardizing Virtual Server C (6). The other physical servers and the VIM stop receiving heartbeat messages from Physical Server B (7).

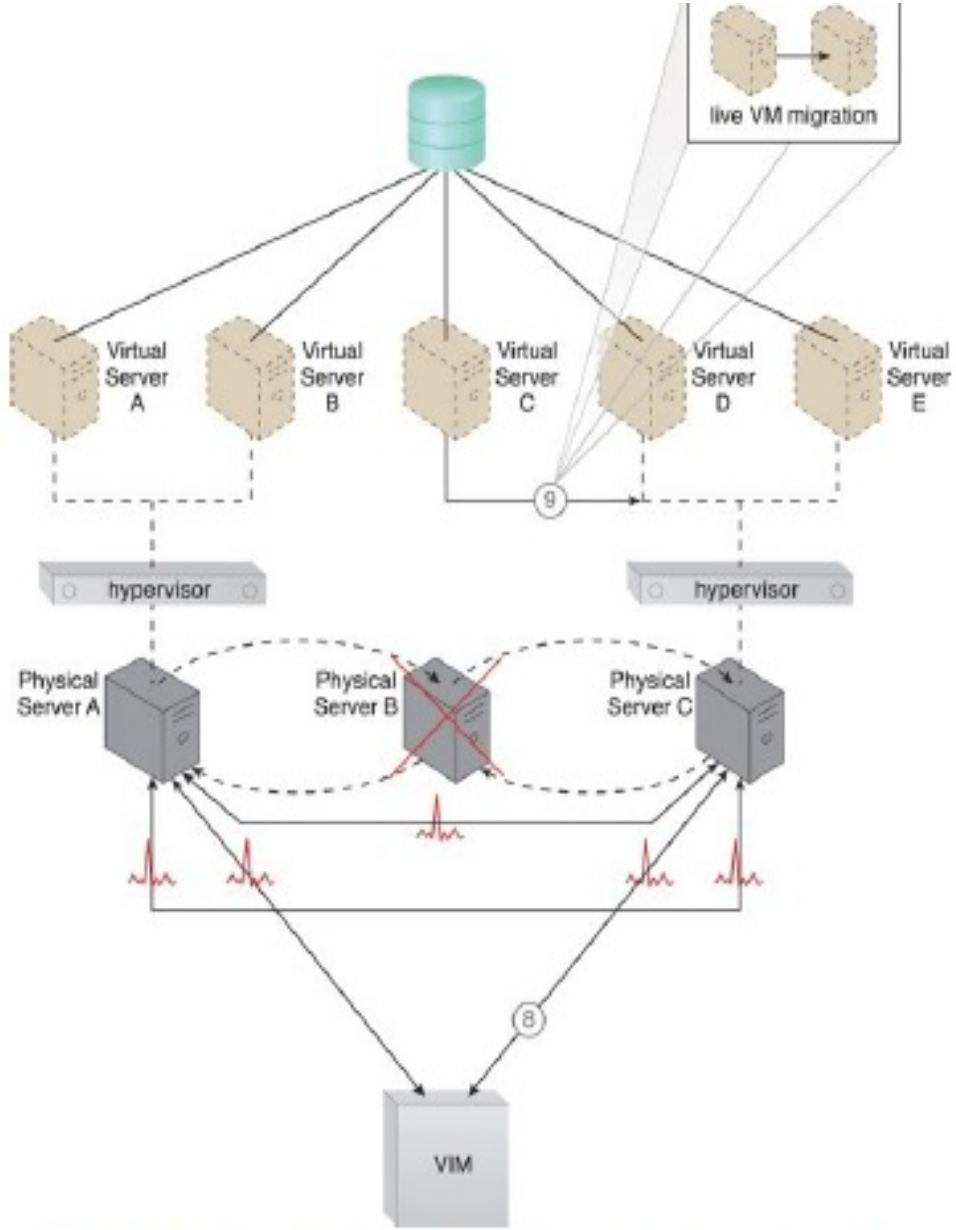


Figure 12.6. The VIM chooses Physical Server C as the new host to take ownership of Virtual Server C after assessing the available capacity of other hypervisors in the cluster (8). Virtual Server C is live-migrated to the hypervisor

In addition to the hypervisor and resource cluster mechanisms that form the core of this architectural model and the virtual servers that are protected by the clustered environment, the following mechanisms can be incorporated:

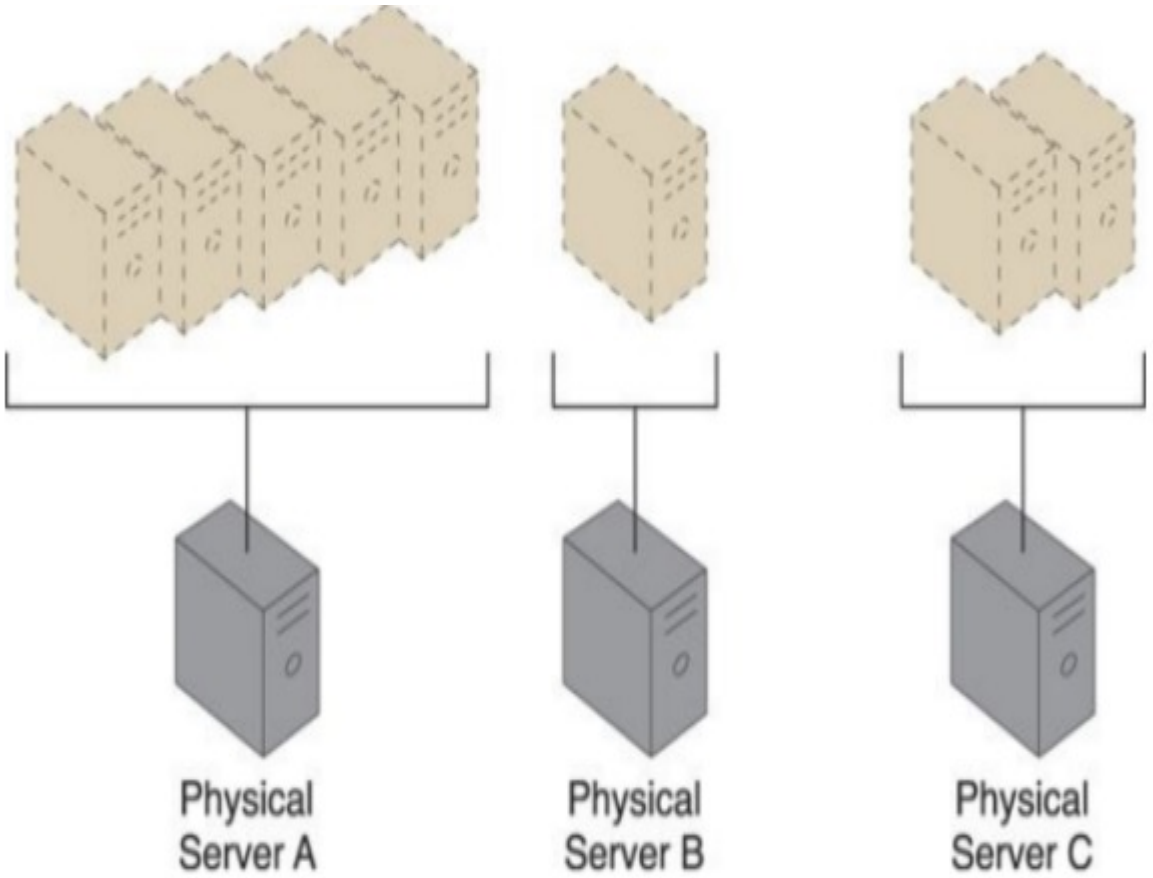
- *Logical Network Perimeter* – The logical boundaries created by this mechanism ensure that none of the hypervisors of other cloud consumers are accidentally included in a given cluster (sandbox→isolation).
- *Resource Replication* – Hypervisors in the same cluster inform one another about their status and availability. Updates on any changes that occur in the cluster, such as the creation or deletion of a virtual switch, need to be replicated to all of the hypervisors via the VIM.

Load Balanced Virtual Server Instances Architecture

Keeping ***cross-server workloads*** evenly balanced between physical servers whose operation and management are isolated can be challenging.

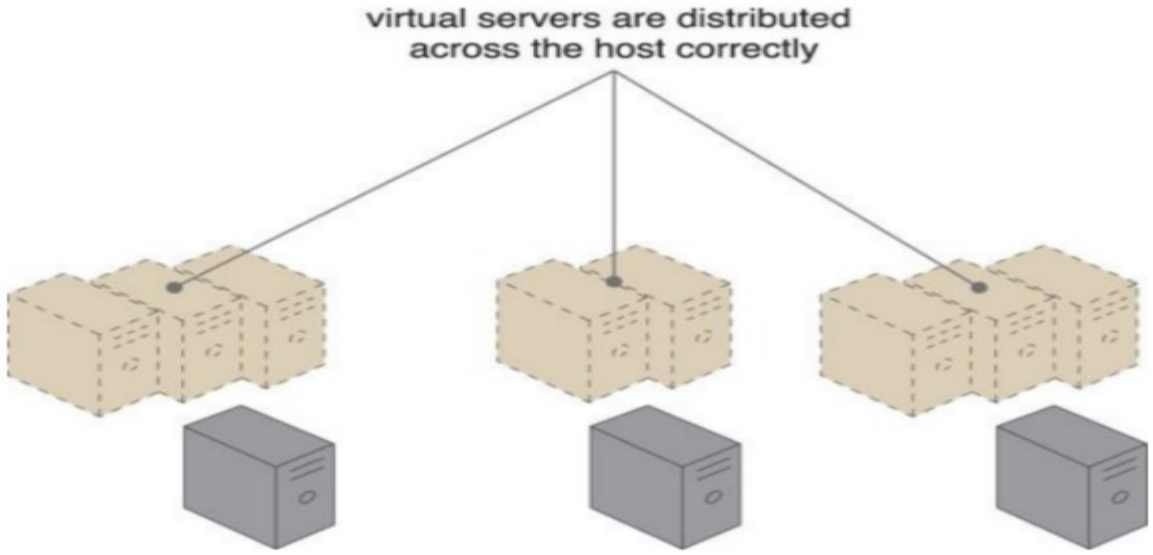
A physical server can easily end up hosting more virtual servers or receive larger workloads than its neighbouring physical servers (Figure in next slide).

Both physical server *over and under-utilization can increase dramatically over time, leading to on-going performance challenges* (for over-utilized servers) and constant waste (for the lost processing potential of under-utilized servers).



Three physical servers have to host different quantities of virtual server instances, leading to both over-utilized and under-utilized servers.

- The load balanced virtual server instances architecture establishes a ***capacity watchdog system*** that dynamically calculates virtual server instances and associated workloads, before distributing the processing across available physical server hosts



The virtual server instances are more evenly distributed across the physical server hosts.

- The capacity watchdog system is comprised of a ***capacity watchdog cloud usage monitor, the live VM migration program, and a capacity planner.***
- The capacity watchdog monitor tracks physical and virtual server usage and reports any significant fluctuations to the capacity planner, which is responsible for ***dynamically calculating physical server computing capacities against virtual server capacity requirements***.
- If the capacity planner decides to move a virtual server to another host to distribute the workload, the live ***VM migration program is signalled to move the virtual server.***

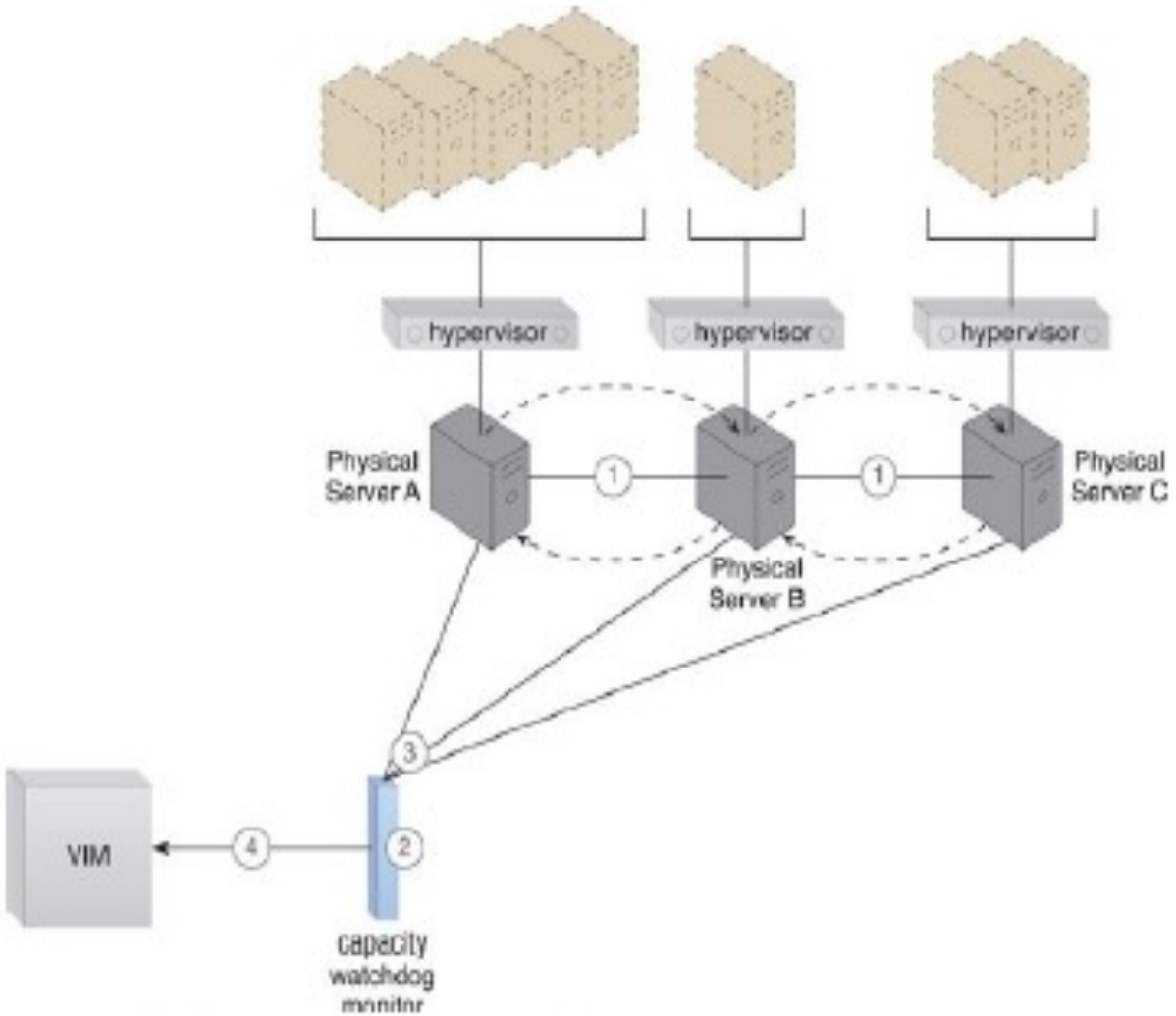


Figure 12.9. The hypervisor cluster architecture provides the foundation upon which the load-balanced virtual server architecture is built (1). Policies and thresholds are defined for the capacity watchdog monitor (2), which compares physical server capacities with virtual server processing (3). The capacity watchdog monitor reports an over-utilization to the VIM (4).

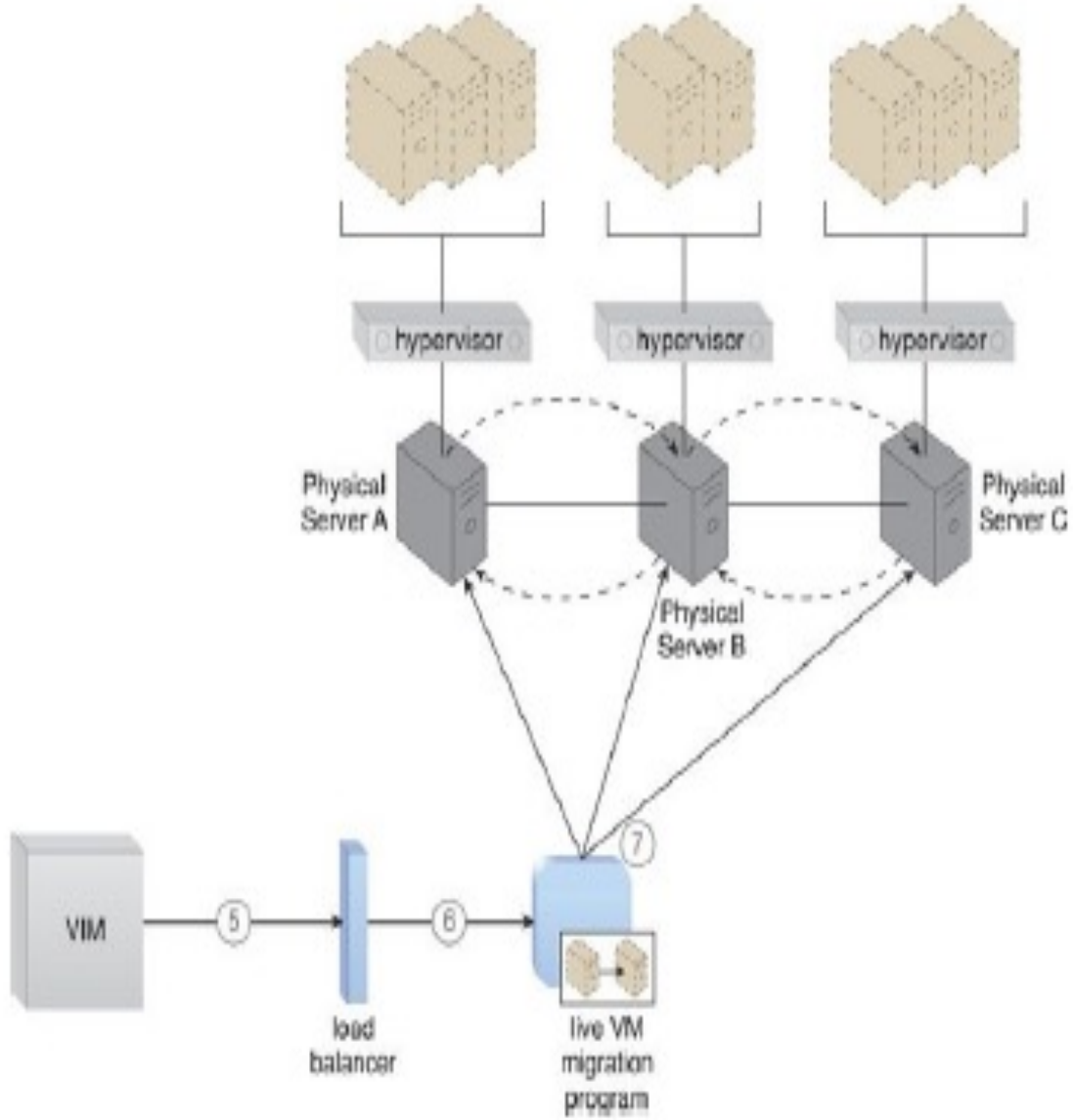


Figure 12.10. The VIM signals the load balancer to redistribute the workload based on pre-defined thresholds (5). The load balancer initiates the live VM migration program to move the virtual servers (6). Live VM migration moves the selected virtual servers from one physical host to another (7).

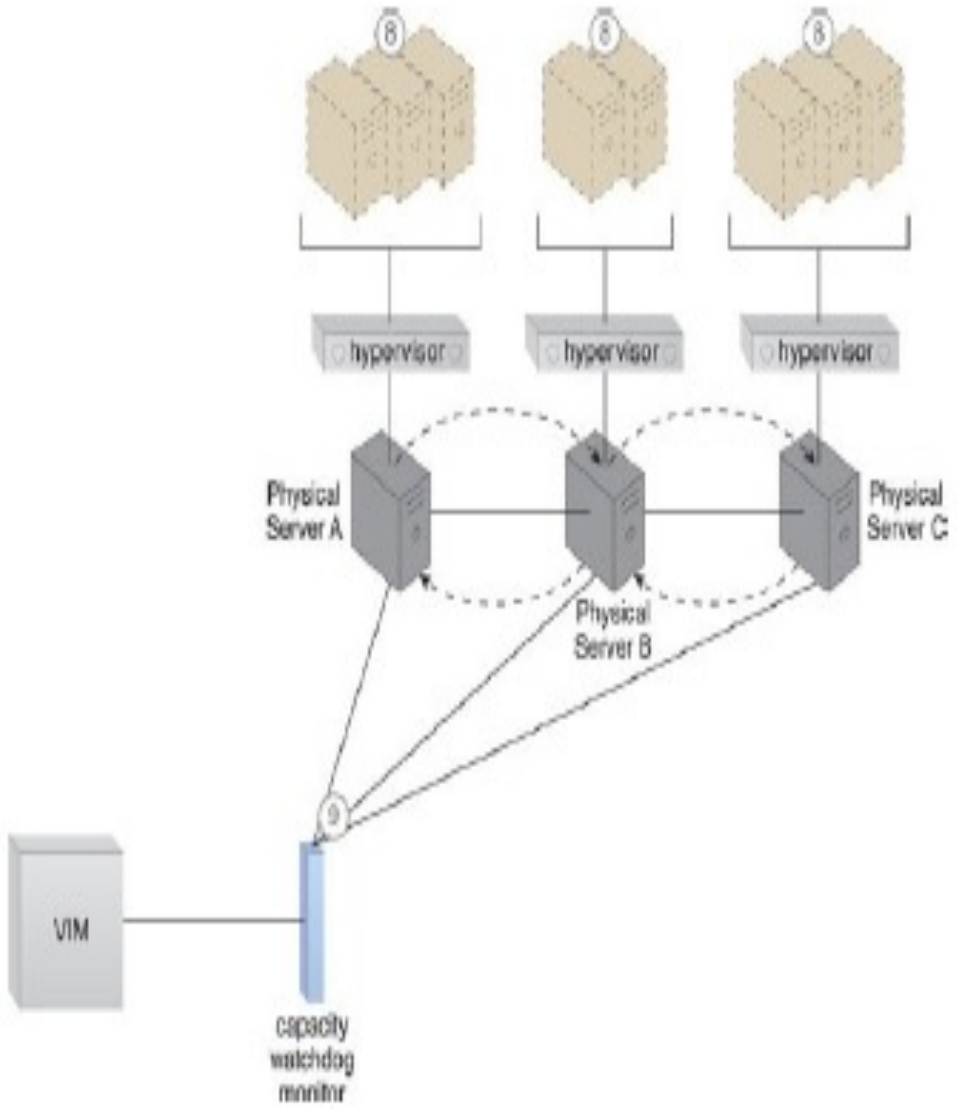


Figure 12.11. The workload is balanced across the physical servers in the cluster (8). The capacity watchdog continues to monitor the workload and resource consumption (9).

The following mechanisms can be included in this architecture, in addition to the hypervisor, resource clustering, virtual server, and (capacity watchdog) cloud usage monitor:

- ***Automated Scaling Listener*** – The automated scaling listener may be used to initiate the process of load balancing and to dynamically monitor workload coming to the virtual servers via the hypervisors.
- ***Load Balancer*** – The load balancer mechanism is responsible for distributing the workload of the virtual servers between the hypervisors.
- ***Logical Network Perimeter*** – A logical network perimeter ensures that the destination of a given relocated virtual server is in compliance with SLA and privacy regulations.
- ***Resource Replication*** – The replication of virtual server instances may be required as part of the load balancing functionality.

Non-Disruptive Service Relocation Architecture

- A cloud service can become ***unavailable for a number of reasons***, such as:
- Runtime usage demands that exceed its processing capacity.
- A maintenance update that mandates a temporary outage.
- Permanent migration to a new physical server host.

Cloud service consumer requests are usually rejected if a cloud service becomes unavailable, which can potentially result in exception conditions.

Rendering the cloud service temporarily unavailable to cloud consumers is not preferred even if the outage is planned.

The non-disruptive service relocation architecture establishes a system by which a predefined event triggers the duplication or migration of a cloud service implementation at runtime, thereby avoiding any disruption.

Instead of scaling cloud services in or out with redundant implementations, cloud service activity can be ***temporarily diverted to another hosting environment at runtime by adding a duplicate implementation*** onto a new host.

Similarly, cloud service consumer requests can be temporarily ***redirected to a duplicate implementation when the original implementation needs to undergo a maintenance outage***.

A key aspect of the underlying architecture is that the new cloud service implementation is guaranteed to be successfully receiving and responding to cloud service consumer requests ***before the original cloud service implementation is deactivated or removed.***

A common approach is for ***live VM migration*** to move the entire virtual server instance that is hosting the cloud service.

The automated ***scaling listener and/or load balancer mechanisms can be used to trigger a temporary redirection of cloud service consumer requests, in response to scaling and workload distribution requirements.***

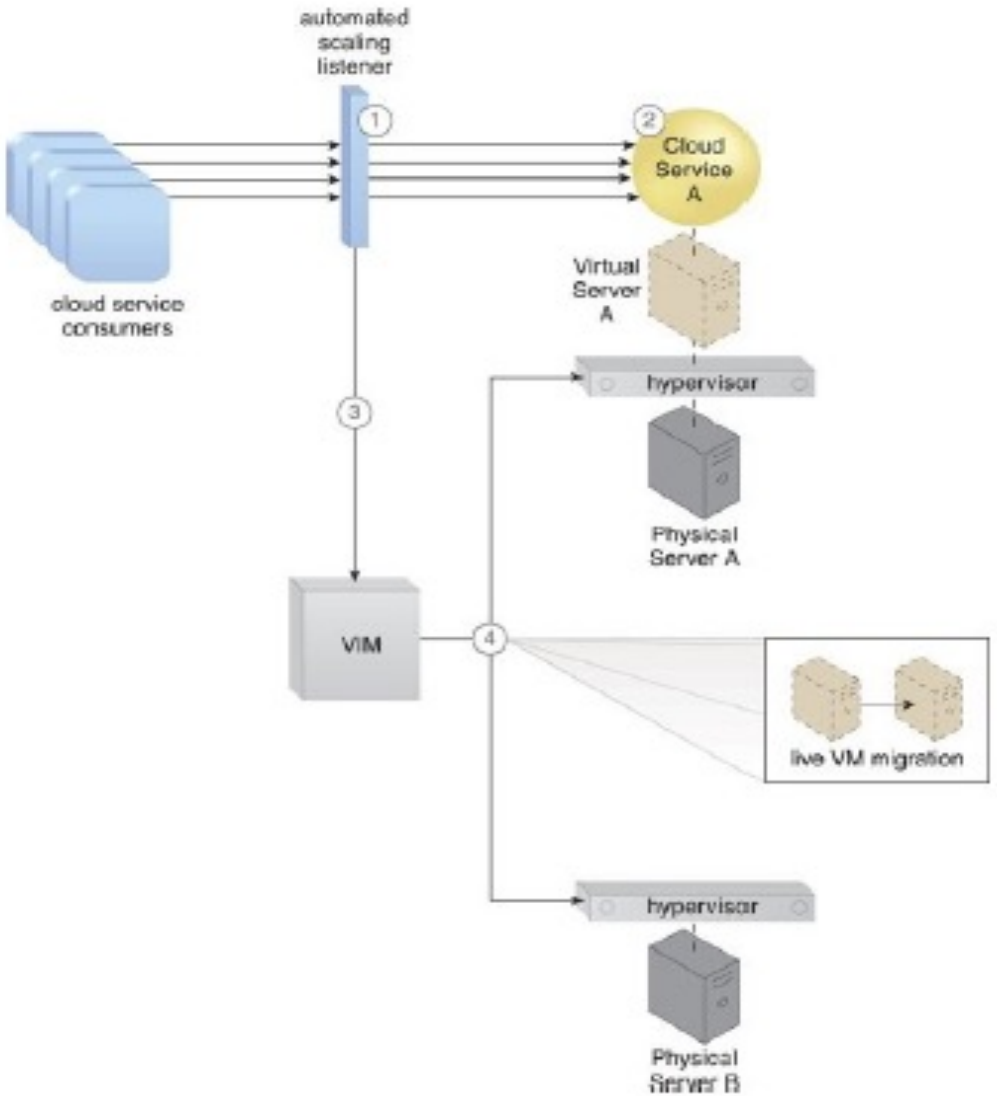


Figure 12.12. The automated scaling listener monitors the workload for a cloud service (1). The cloud service's predefined threshold is reached as the workload increases (2), causing the automated scaling listener to signal the VIM to initiate relocation (3). The VIM uses the live VM migration program to instruct both the origin and destination hypervisors to carry out runtime relocation (4).

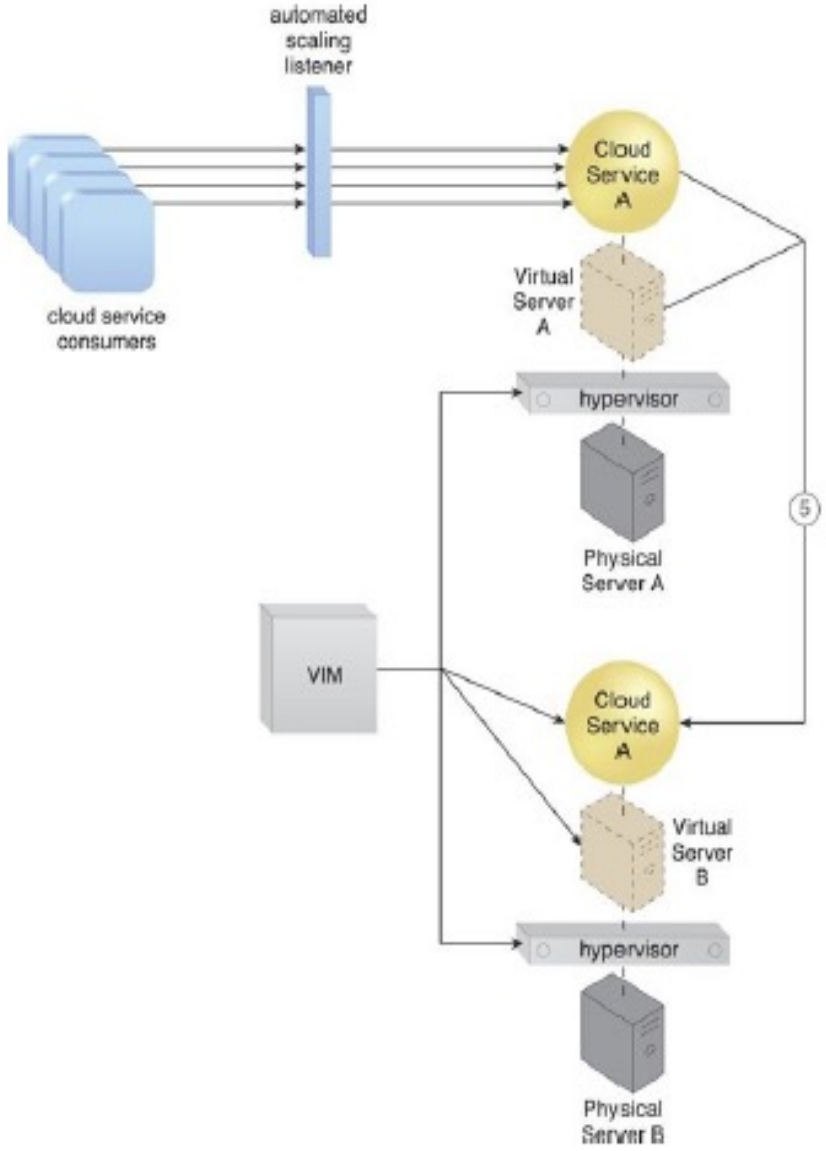


Figure 12.13. A second copy of the virtual server and its hosted cloud service are created via the destination hypervisor on Physical Server B (5).

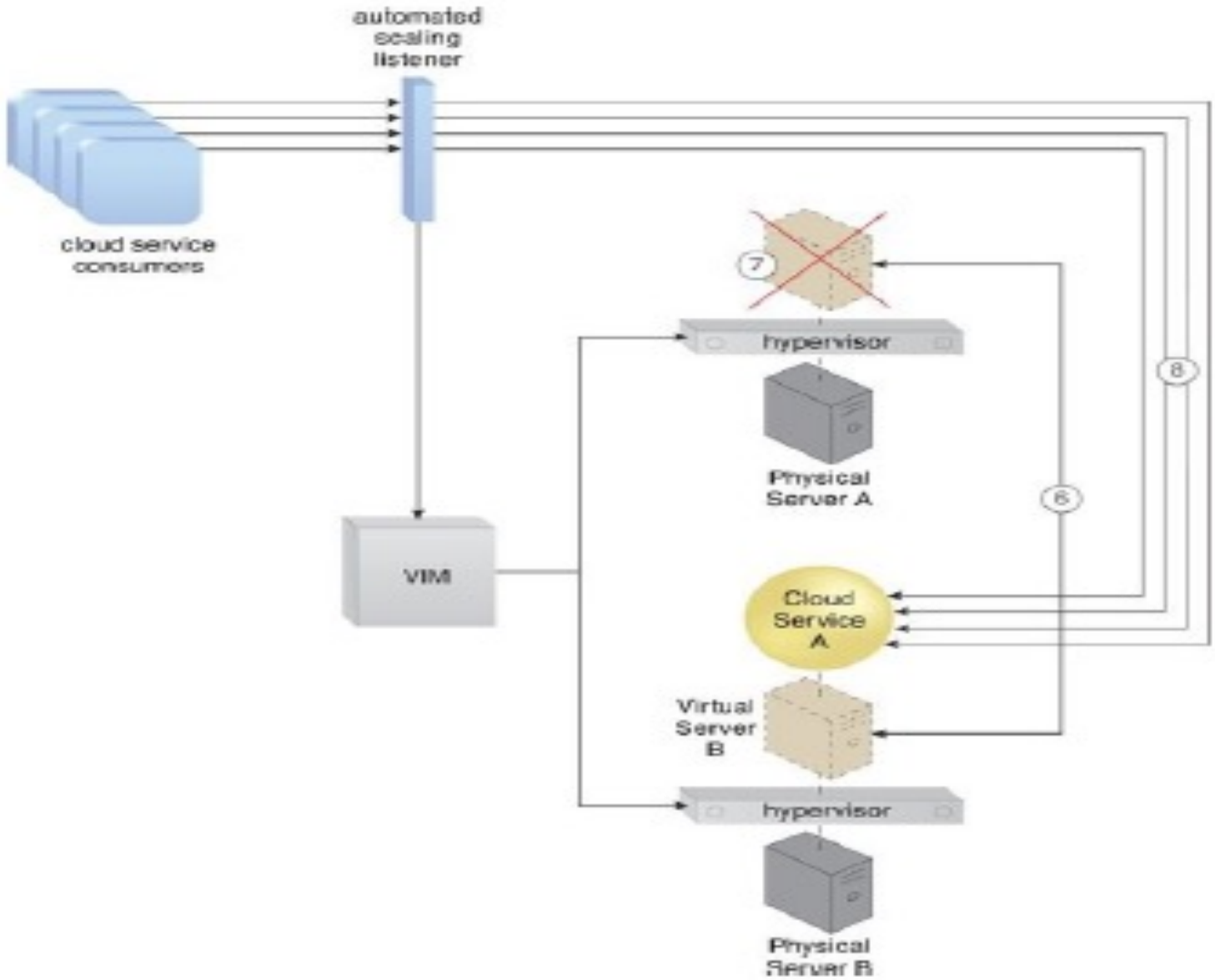


Figure 12.14. The state of both virtual server instances is synchronized (6). The first virtual server instance is removed from Physical Server A after cloud service consumer requests are confirmed to be successfully exchanged with the cloud service on Physical Server B (7). Cloud service consumer requests are now only sent to the cloud service on Physical Server B (8).

- This architecture can be supported by the persistent virtual network configurations architecture, so that ***the defined network configurations of migrated virtual servers are preserved to retain connection with the cloud service*** consumers.
- Besides the automated scaling listener, load balancer, cloud storage device, hypervisor, and virtual server, other mechanisms that can be part of this architecture include the following:-
- ***Cloud Usage Monitor*** – Different types of cloud usage monitors can be used to continuously track IT resource usage and system activity.
- ***Pay-Per-Use Monitor*** – The pay-per-use monitor is used to collect data for service usage cost calculations for IT resources at both source and destination locations.
- ***Resource Replication*** – The resource replication mechanism is used to instantiate the shadow copy of the cloud service at its destination.

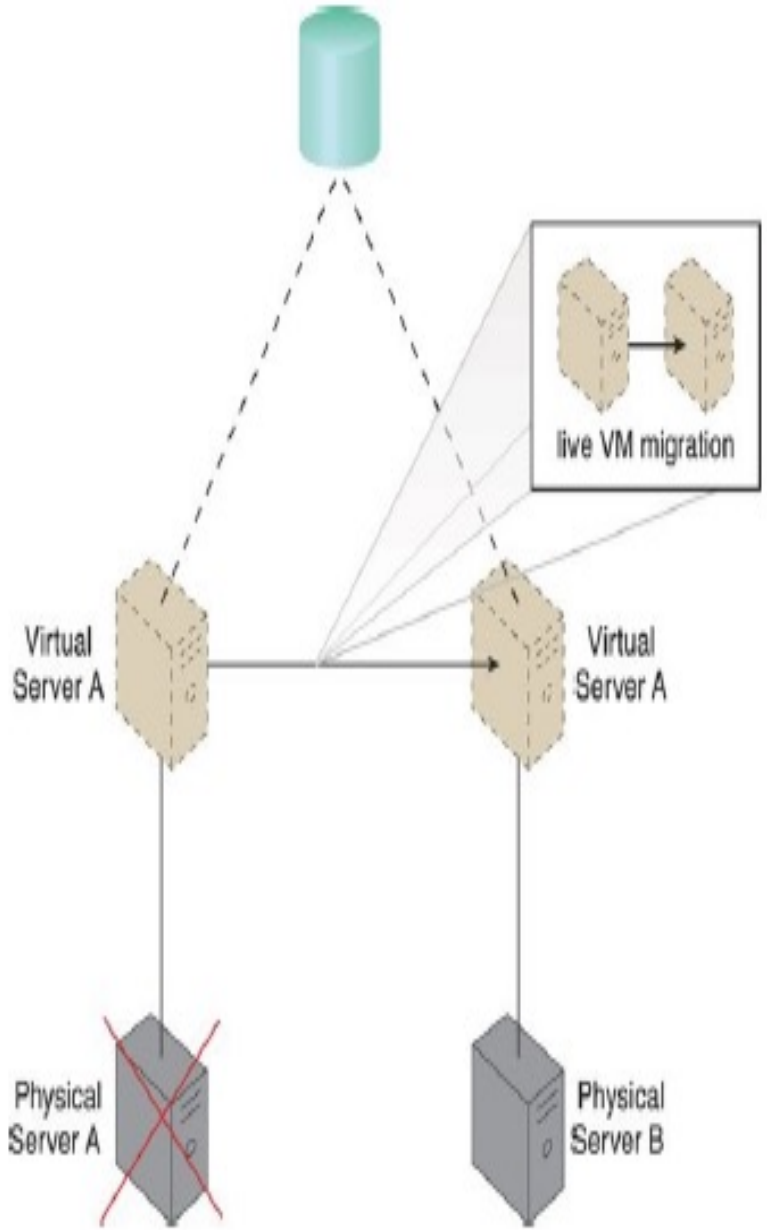
- ***SLA Management System*** – This management system is responsible for processing SLA data provided by the SLA monitor to obtain cloud service availability assurances, both during and after cloud service duplication or relocation.
- ***SLA Monitor*** – This monitoring mechanism collects the SLA information required by the SLA management system, which may be relevant if availability guarantees rely on this architecture.

ZERO DOWNTIME ARCHITECTURE

A physical server naturally acts as a single point of failure for the virtual servers it hosts. As a result, when the physical server fails or is compromised, the availability of any (or all) hosted virtual servers can be affected.

This makes the issuance of zero downtime guarantees by a cloud provider to cloud consumers challenging.

The zero downtime architecture establishes a sophisticated failover system ***that allows virtual servers to be dynamically moved to different physical server hosts, in the event that their original physical server host fails*** (and is shown in the next figure).



Physical Server A fails triggering the live VM migration program to dynamically move Virtual Server A to Physical Server B.

Multiple physical servers are assembled into a group that is controlled by a ***fault tolerance system capable of switching activity from one physical server to another, without interruption***.

The live VM migration component is typically a core part of this form of high availability cloud architecture.

Besides the failover system, cloud storage device, and virtual server mechanisms, the following mechanisms can be part of this architecture:

- ***Audit Monitor*** – This mechanism may be required to check whether the relocation of virtual servers also relocates hosted data to prohibited locations.
- ***Cloud Usage Monitor*** – Incarnations of this mechanism are used to monitor the actual IT resource usage of cloud consumers to help ensure that virtual server capacities are not exceeded.
- ***Hypervisor*** – The hypervisor of each affected physical server hosts the affected virtual servers.
- ***Logical Network Perimeter*** – Logical network perimeters provide and maintain the isolation that is required to ensure that each cloud consumer remains within its own logical boundary subsequent to virtual server relocation.

- *Resource Cluster* – The resource cluster mechanism is applied to create ***different types of active-active cluster groups that collaboratively improve the availability of virtual server-hosted IT resources***.
- *Resource Replication* – This mechanism can create the new virtual server and cloud service instances upon primary virtual server failure.

Cloud Balancing Architecture

The cloud balancing architecture establishes a specialized architectural model in which IT resources can be load-balanced across multiple clouds.

- The cross-cloud balancing of cloud service consumer requests can help:

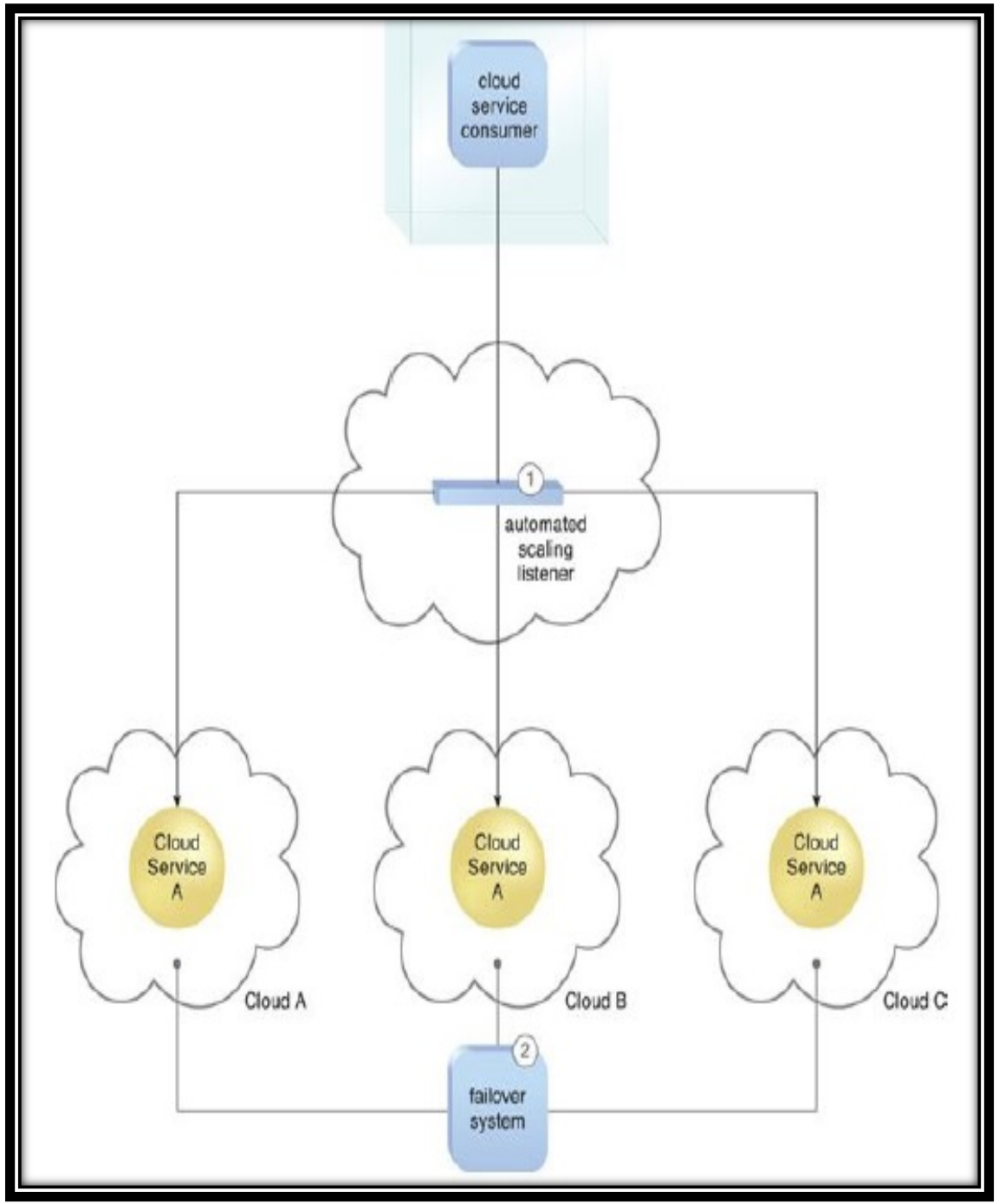
To improve the ***performance and scalability*** of IT resources

To increase the ***availability and reliability*** of IT resources

To improve ***load-balancing and IT resource optimization***

Cloud balancing functionality is primarily based on the combination of the automated scaling listener and failover system mechanisms (Figure in next slide).

Many more components (and possibly other mechanisms) can be part of a complete cloud balancing architecture.



An automated scaling listener controls the cloud balancing process by routing cloud service consumer requests to redundant implementations of Cloud Service A distributed across multiple clouds (1). The failover system instills (promotes) resiliency within this architecture by providing cross- cloud failover (2).

The two mechanisms are utilized as follows:

- The automated scaling listener redirects cloud service consumer requests to ***one of several redundant IT resource implementations, based on current scaling and performance requirements.***
- The *failover system ensures that redundant IT resources are capable of cross-cloud failover in the event of a failure within an IT resource or its underlying hosting environment*.
- IT resource failures are announced so that the automated scaling listener can avoid inadvertently routing cloud service consumer requests to unavailable or unstable IT resources.

Some important point:-

1) For a cloud balancing architecture to function effectively, the ***automated scaling listener needs to be aware of all redundant IT resource implementations within the scope of the cloud balanced architecture.***

2) Note that if the manual synchronization of cross-cloud IT resource implementations is not possible, ***the resource replication mechanism may need to be incorporated to automate the synchronization***.

Resource Reservation Architecture

Problem statement:-Depending on how IT resources are designed for shared usage and depending on their available levels of capacity, concurrent access can lead to a runtime exception condition called resource constraint.

A resource constraint is a condition that occurs when two or more cloud consumers have been allocated to share an IT resource that does not have the capacity to accommodate the total processing requirements of the cloud consumers.

As a result, ***one or more of the cloud consumers encounter degraded performance or may be rejected altogether.***

The cloud service itself may go down, resulting in all cloud consumers being rejected.

Another problem:-Other types of runtime conflicts can occur when an IT resource (especially one not specifically designed to accommodate sharing) is concurrently accessed by different cloud service consumers.

For example, ***nested and sibling resource pools introduce the notion of resource borrowing, whereby one pool can temporarily borrow IT resources from other pools.***

A runtime conflict can be triggered when the borrowed IT resource is not returned due to prolonged usage by the cloud service consumer that is borrowing it.

This can inevitably lead back to the occurrence of ***resource constraints***.

Solution:-

The resource reservation architecture establishes a system whereby ***one of the following is set aside exclusively*** for a given cloud consumer

- Single IT resource
- Portion of an IT resource
- Multiple IT resources

This protects cloud consumers from each other by avoiding the aforementioned resource constraint and resource borrowing conditions.

The creation of an IT resource reservation system can require involving the resource management system mechanism, ***which is used to define the usage thresholds for individual IT resources and resource pools.***

Reservations lock the amount of IT resources that each pool needs to keep, with the balance of the pool's IT resources still available for sharing and borrowing.

The remote administration system mechanism is also used to enable front-end customization, so that cloud consumers have administration controls for the management of their reserved IT resource allocations.

The types of mechanisms that are commonly reserved within this architecture are cloud storage devices and virtual servers. Other mechanisms that may be part of the architecture can include:

- ***Audit Monitor*** – The audit monitor is used to check whether the resource reservation system is complying with cloud consumer auditing, privacy, and other regulatory requirements. For example, it may track the geographical location of reserved IT resources.
- ***Cloud Usage Monitor*** – A cloud usage monitor may oversee the thresholds that trigger the allocation of reserved IT resources.
- ***Hypervisor*** – The hypervisor mechanism may apply reservations for different cloud consumers to ensure that they are correctly allocated to their guaranteed IT resources.
- ***Logical Network Perimeter*** – This mechanism establishes the boundaries necessary to ensure that reserved IT resources are made exclusively available to cloud consumers.
- ***Resource Replication*** – This component needs to stay informed about each cloud consumer's limits for IT resource consumption, in order to replicate and provision new IT resource instances expediently.

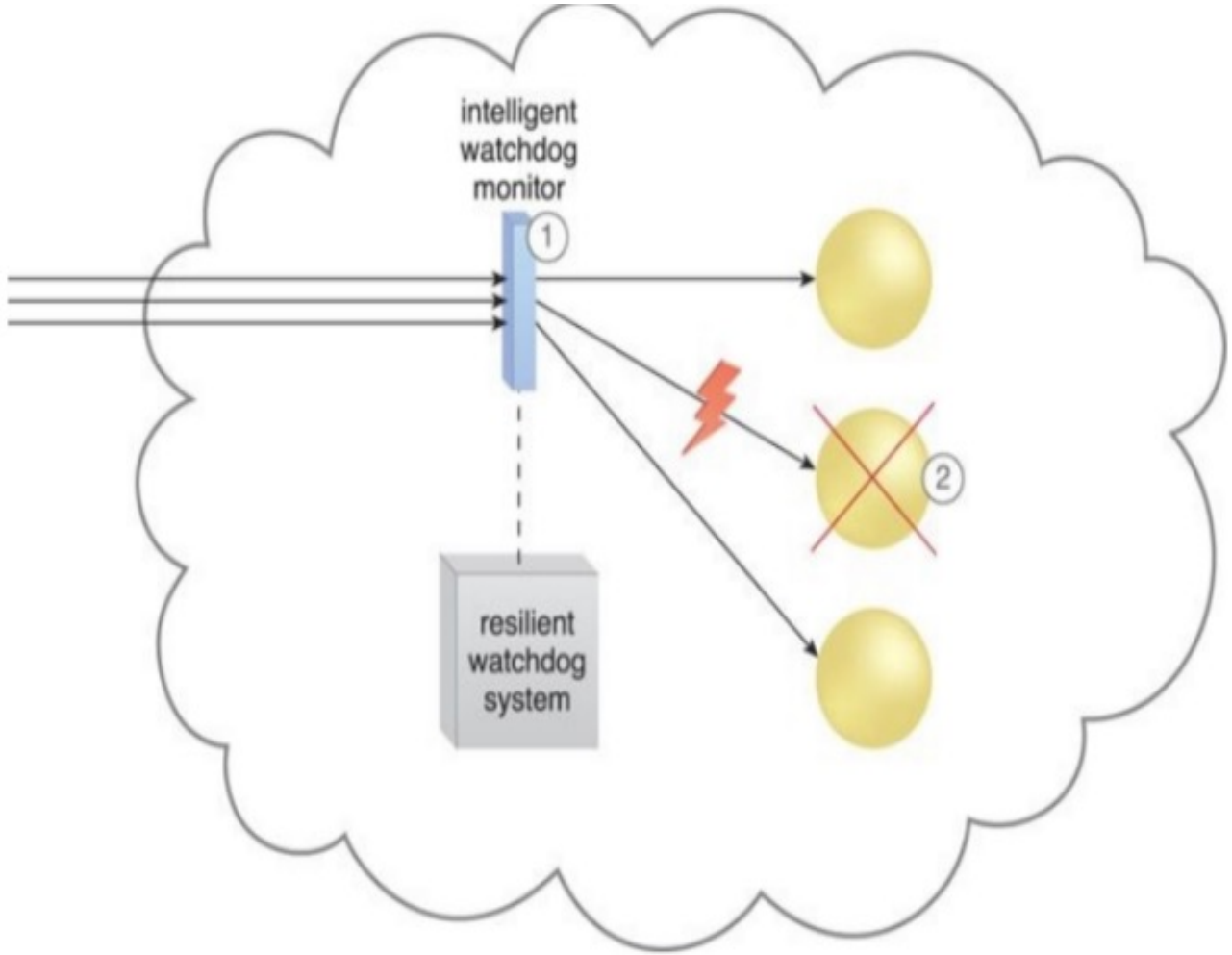
Dynamic Failure Detection and Recovery Architecture

Cloud-based environments can be comprised of vast quantities of IT resources that are simultaneously accessed by numerous cloud consumers.

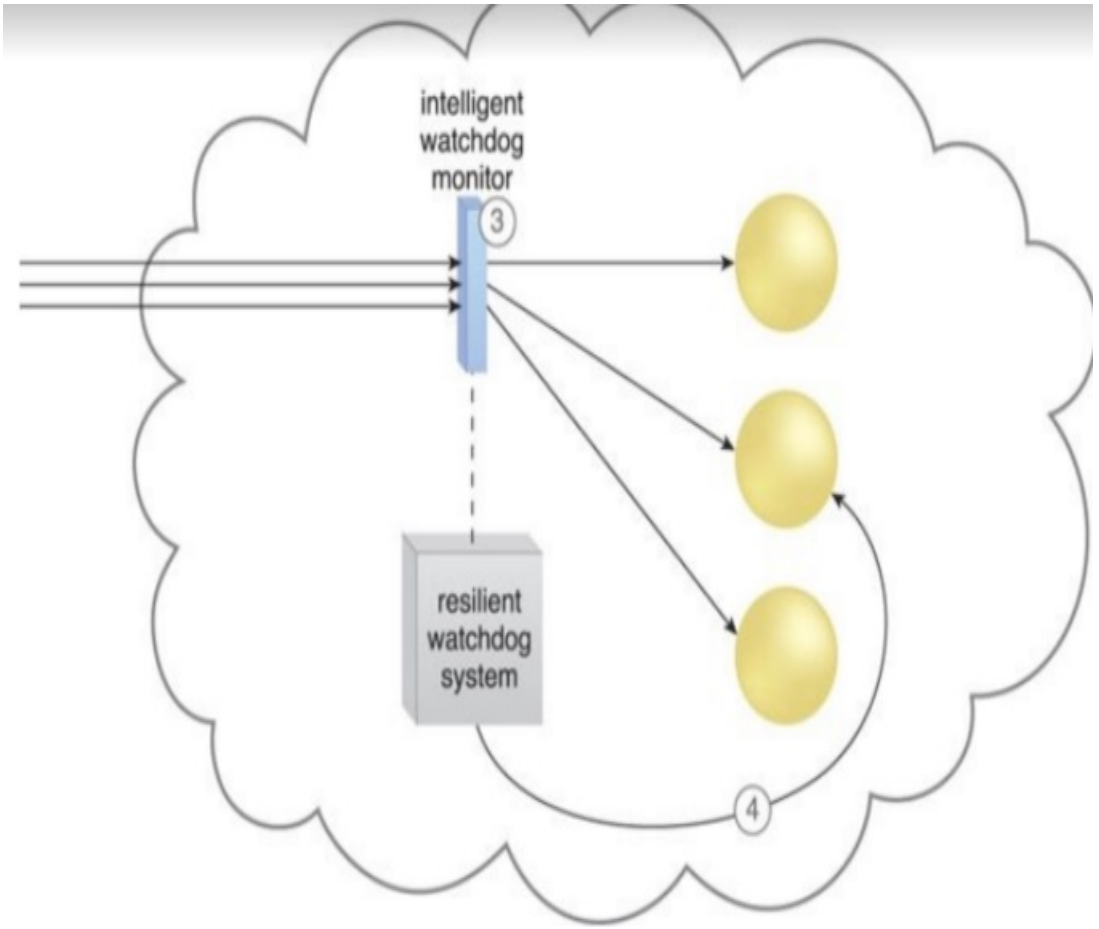
Any of those IT resources can experience failure conditions that require more than manual intervention to resolve.

Manually administering and solving IT resource failures is generally inefficient and impractical.

- 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* (Figures shown in the next slide).
- This system notifies and escalates the failure conditions that it cannot automatically resolve itself.
- It relies on a specialized cloud ***usage monitor called the intelligent watchdog monitor to actively track IT resources and take pre-defined actions in response to pre-defined events.***



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

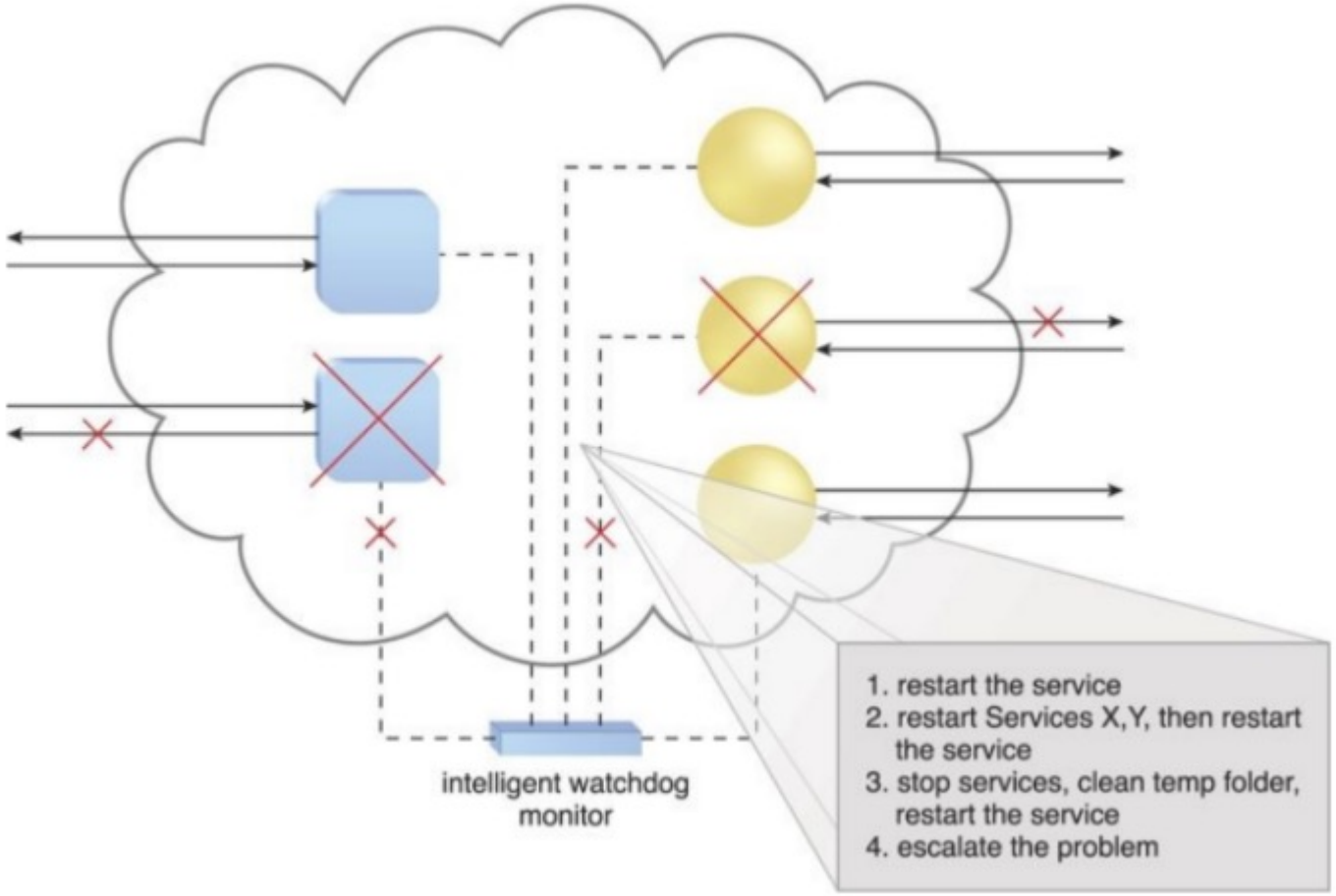


The intelligent watchdog monitor notifies the watchdog system (3), which restores the cloud service based on pre-defined policies. The cloud service resumes its runtime operation (4).

The resilient watchdog system performs the following five core functions:

- Watching
- Deciding upon an event
- Acting upon an event
- Reporting
- Escalating (fast operational)

Sequential recovery policies can be defined for each IT resource to determine the steps that the intelligent watchdog monitor needs to take when a failure condition occurs. ***For example, a recovery policy can state that one recovery attempt needs to be automatically carried out before issuing a notification.***



In the event of a failure, the intelligent watchdog monitor refers to its pre-defined policies to recover the cloud service step-by-step, escalating the process when a problem proves to be deeper than expected.

Some of the actions the intelligent watchdog monitor commonly takes to escalate an issue include:

- Running a batch file
- Sending a console message
- Sending a text message
- Sending an email message
- Sending an SNMP (Simple Network Management Protocol) trap: an agent can send an unrequested message to the manager to notify about an important event.
- Logging a ticket

There are varieties of programs and products that can act as intelligent watchdog monitors.

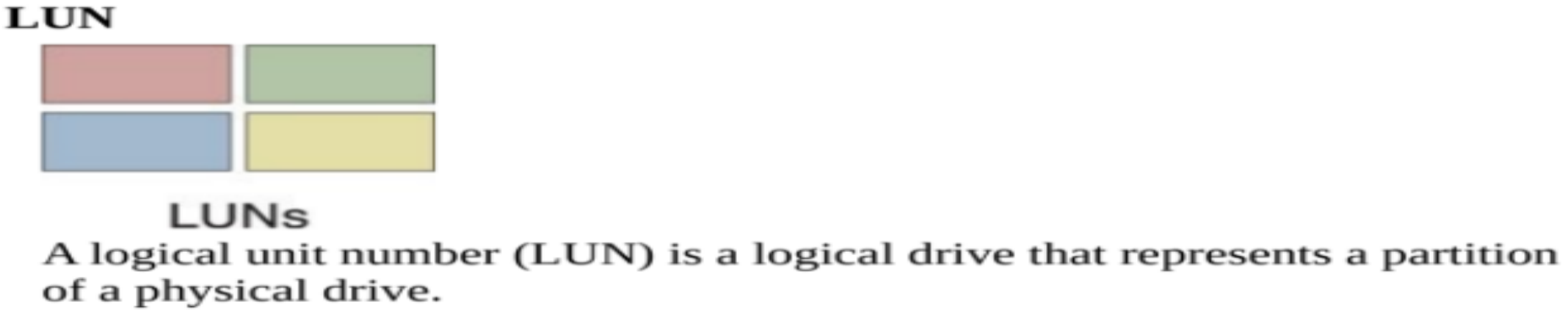
Most can be integrated with standard ticketing and event management systems.

This architectural model can further incorporate the following mechanisms:

- ***Audit Monitor*** – This mechanism is used to track whether data recovery is carried out in compliance with legal or policy requirements.
- ***Failover System*** – The failover system mechanism is usually used during the initial attempts to recover failed IT resources.
- ***SLA Management System and SLA Monitor*** – Since the functionality achieved by applying this architecture is closely associated with SLA guarantees, the system commonly relies on the information that is managed and processed by these mechanisms.

Storage Workload Management Architecture

Note: A *logical unit number (**LUN**) is a unique identifier* for designating an individual or collection of physical or virtual **storage** devices that execute input/output (I/O) commands with a host computer, as defined by the Small System Computer Interface (SCSI) standard.



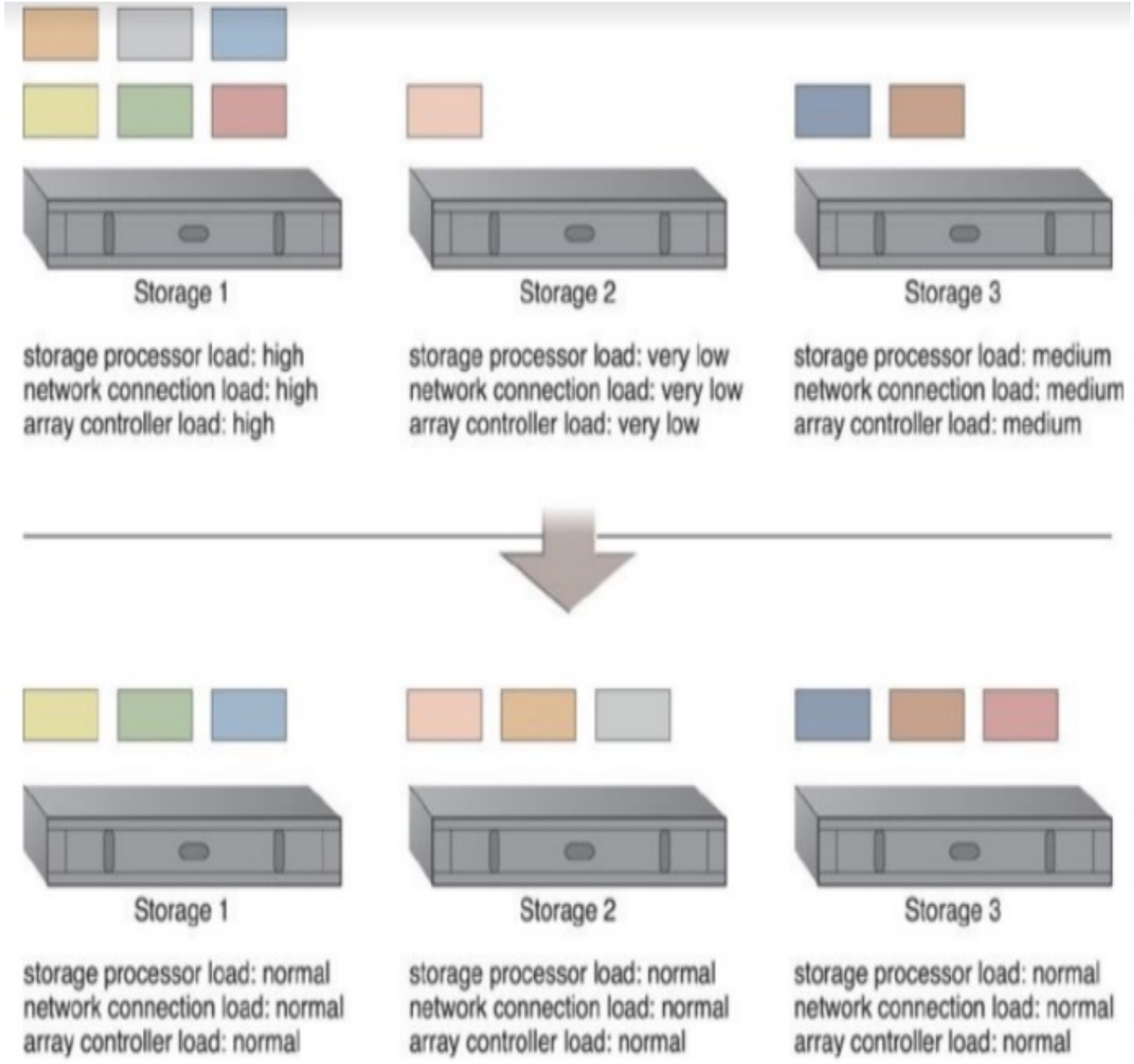
Over-utilized cloud storage devices increase the workload on the storage controller and can cause a range of performance challenges. Conversely, cloud storage devices that are under-utilized are wasteful due to lost processing and storage capacity potential.

LUN migration

LUN migration is a specialized storage program that is used to move LUNs from one storage device to another without interruption, while remaining transparent to cloud consumers.

LUN migration

The storage workload management architecture enables LUNs to be evenly distributed across available cloud storage devices, while a storage capacity system is established to ensure that runtime workloads are evenly distributed across the LUNs

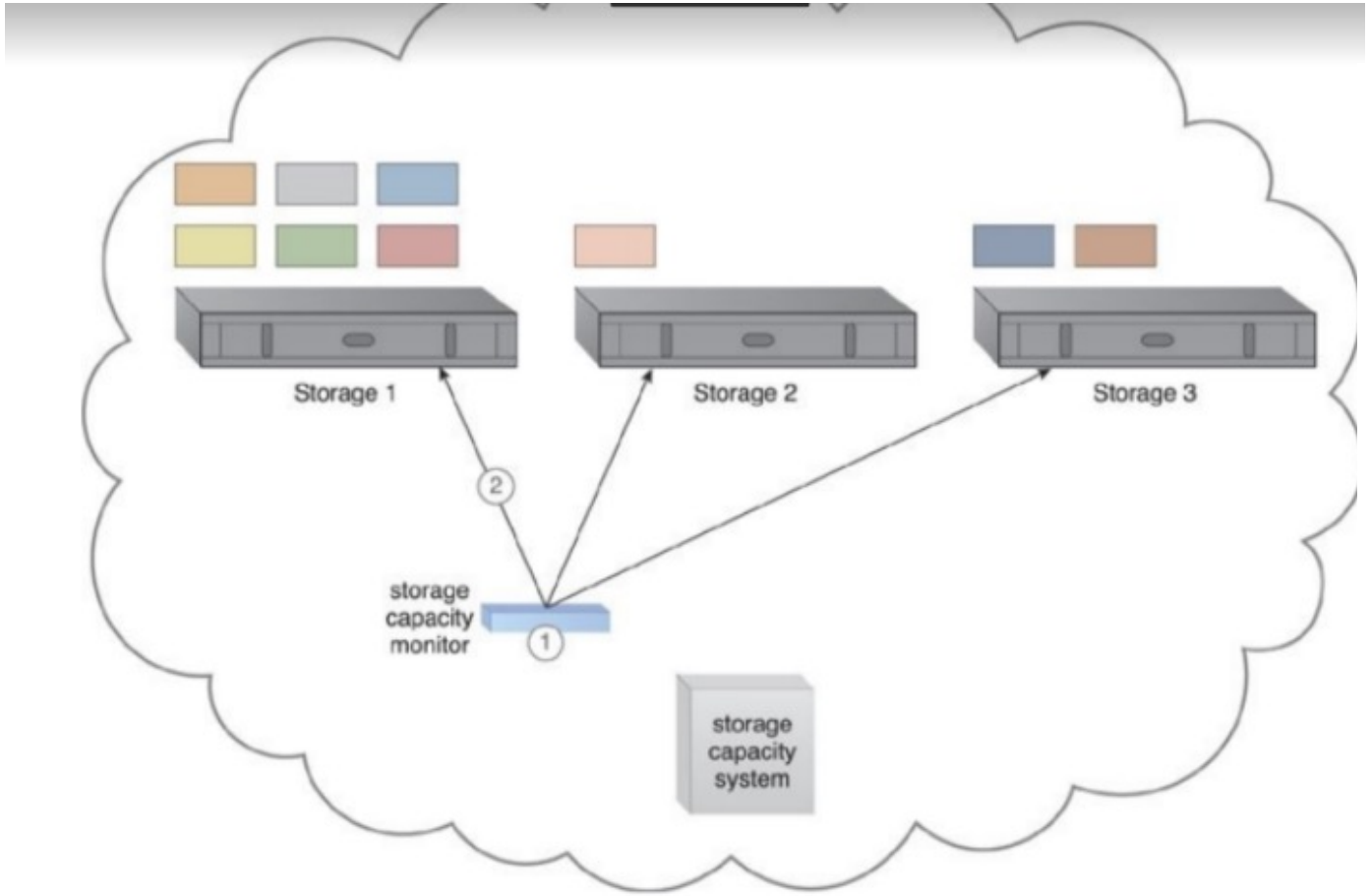


Workload should be evenly distributed amongst the LUNs

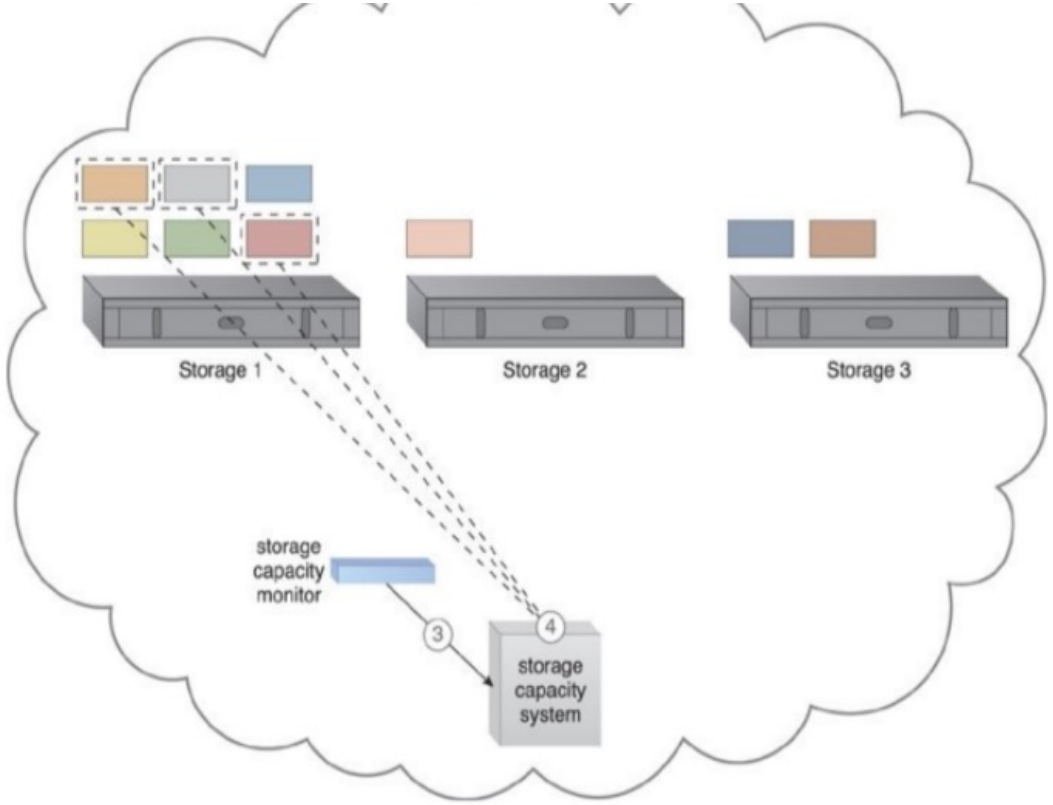
LUNs are dynamically distributed across cloud storage devices, resulting in more even distribution of associated types of workloads.

Combining cloud storage devices into a group allows LUN data to be distributed between available storage hosts equally.

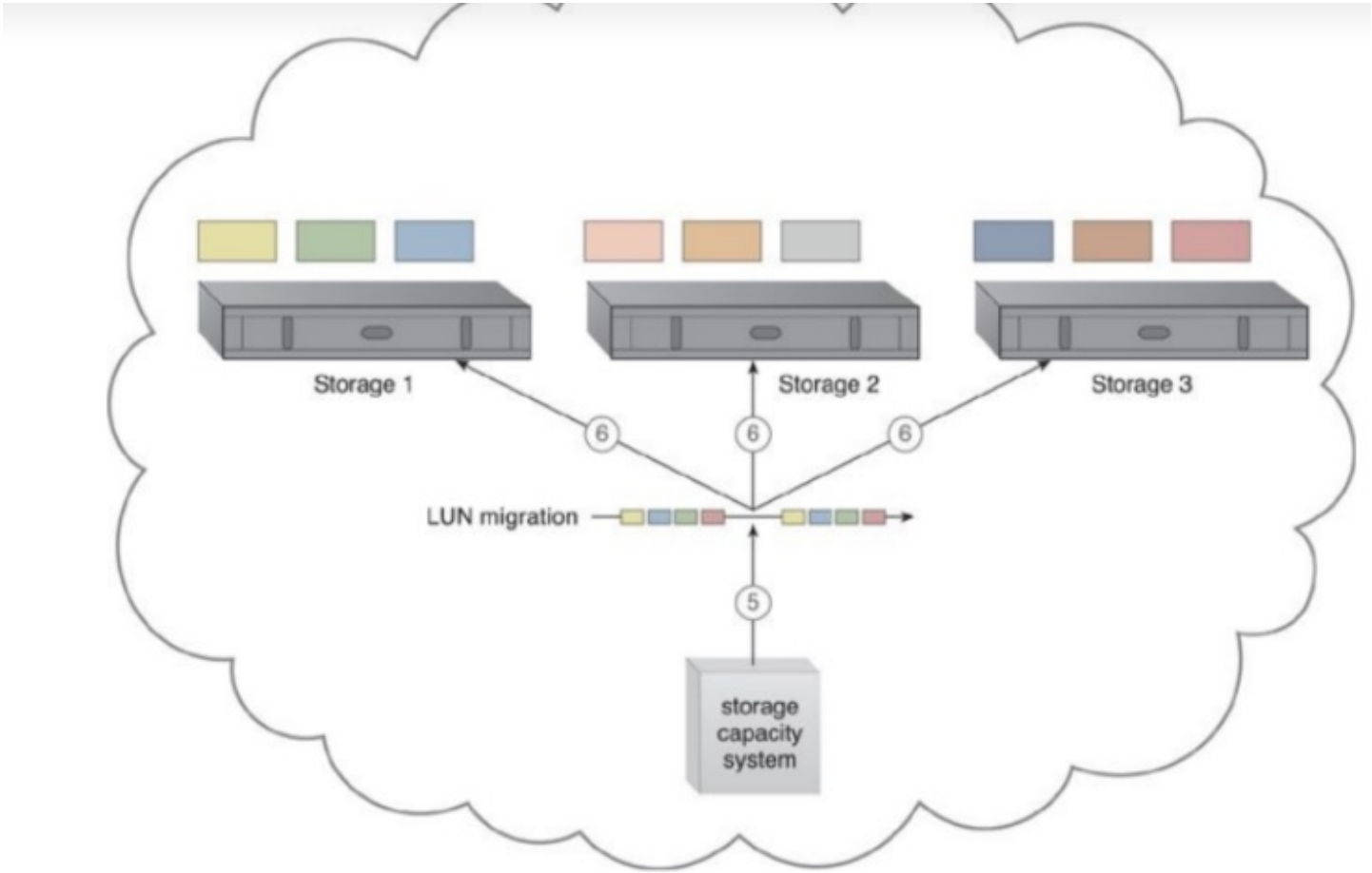
A storage management system is configured and an automated scaling listener is positioned to monitor and equalize runtime workloads among the grouped cloud storage devices.



The storage capacity system and storage capacity monitor are configured to survey three storage devices in realtime, whose workload and capacity thresholds are pre-defined (1). The storage capacity monitor determines that the workload on Storage 1 is reaching its threshold (2).



The storage capacity monitor informs the storage capacity system that Storage 1 is over-utilized (3). The storage capacity system identifies the LUNs to be moved from Storage 1 (4).



The storage capacity system calls for LUN migration to move some of the LUNs from Storage 1 to the other two storage devices (5). LUN migration transitions LUNs to Storage 2 and 3 to balance the workload.

The storage capacity system can keep the hosting storage device in power-saving mode for the periods when the LUNs are being accessed less frequently or only at specific times.

Some other mechanisms that can be included in the storage workload management architecture to accompany the cloud storage device are as follows:

- ***Audit Monitor*** – This monitoring mechanism is used to check for compliance with regulatory, privacy, and security requirements, since the system established by this architecture can physically relocate data.
- ***Automated Scaling Listener*** – The automated scaling listener is used to watch and respond to workload fluctuations.
- ***Cloud Usage Monitor*** – In addition to the capacity workload monitor, specialized cloud usage monitors are used to track LUN movements and collect workload distribution statistics.

- ***Load Balancer*** – This mechanism can be added to horizontally balance workloads across available cloud storage devices.
- ***Logical Network Perimeter*** – Logical network perimeters provide levels of isolation so that cloud consumer data that undergoes relocation remains inaccessible to unauthorized parties.

Unit IV Ends here…….