UNIT 4
CLOUD COMPUTING ARCHITECTURE

Fundamental Cloud Architectures: Workload Distribution Architecture– Resource Pooling Architecture– Dynamic Scalability Architecture– Elastic Resource Capacity Architecture– Service Load Balancing Architecture– Cloud Bursting Architecture – Elastic Disk Provisioning Architecture– Redundant Storage Architecture– **Advanced Cloud Architectures:** Hypervisor Clustering Architecture– Load Balanced Virtual Server Instances Architecture– Dynamic Failure Detection and Recovery Architecture – **Case Study** : AWS, Microsoft Azure

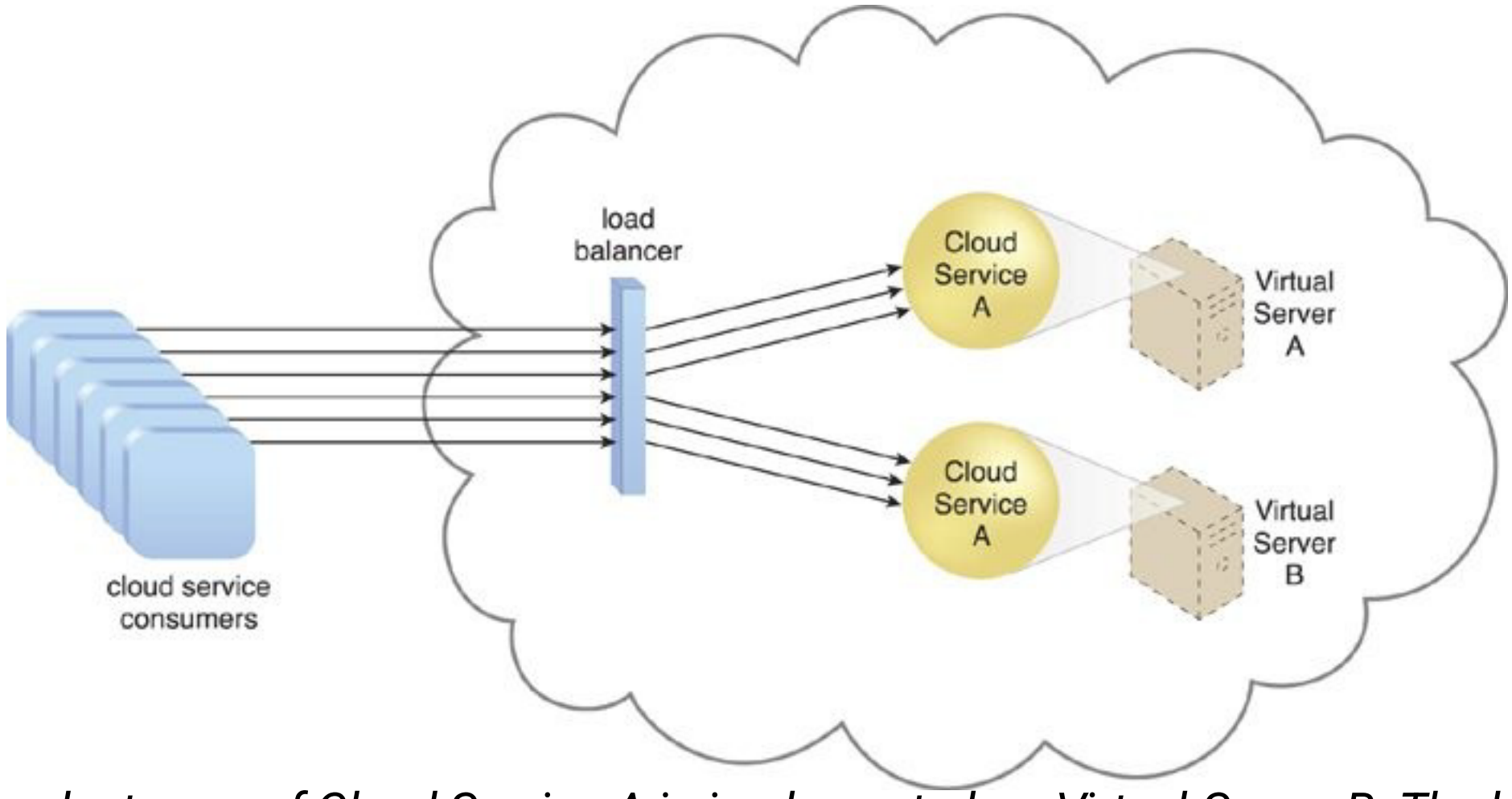
Fundamental Cloud Architectures

- **Workload Distribution Architecture**
- **Resource Pooling Architecture**
- **Dynamic Scalability Architecture**
- **Elastic Resource Capacity Architecture**
- **Service Load Balancing Architecture**
- **Cloud Bursting Architecture**
- **Elastic Disk Provisioning Architecture**
- **Redundant Storage Architecture**

Workload Distribution Architecture

- IT resources can be horizontally scaled via the addition of one or more identical IT resources, and a load balancer that provides runtime logic capable of evenly distributing the workload among the available IT resources
- The resulting *workload distribution architecture reduces both IT resource overutilization* and under-utilization to an extent dependent upon the sophistication of the load balancing algorithms and runtime logic.

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.

This fundamental architectural model can be applied to any IT resource, with workload distribution commonly carried out in support of distributed virtual servers, cloud storage devices, and cloud services.

Workload Distribution Architecture

- In addition to the base load balancer mechanism, and the virtual server and cloud storage device mechanisms, 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.

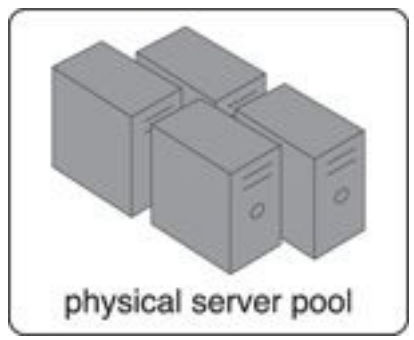
Logical Network Perimeter – The logical network perimeter isolates cloud consumer network boundaries in relation to how and where workloads are distributed.

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

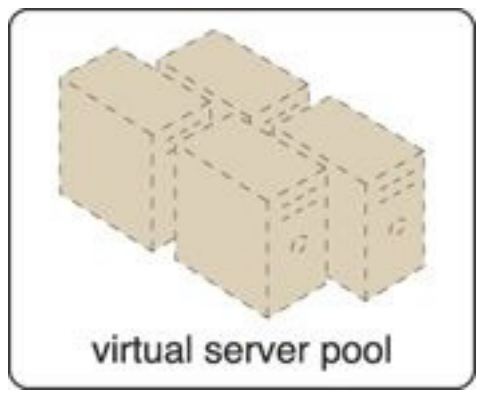
Resource Replication – This mechanism can generate new instances of virtualized IT resources in response to runtime workload distribution demands.

Resource Pooling Architecture

- A *resource pooling architecture is based on the use of one or more resource* pools, in which identical IT resources are grouped and maintained by a system that automatically ensures that they remain synchronized.

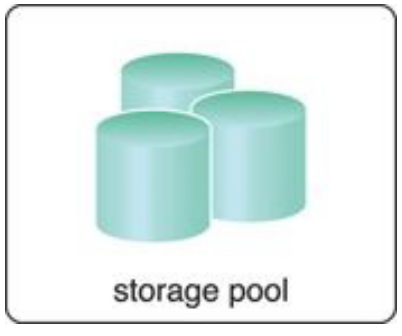


Physical server pools are composed of networked servers that have been installed with operating systems and other necessary programs and/or applications and are ready for immediate use.

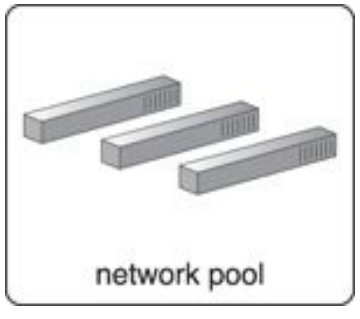


Virtual server pools are usually configured using one of several available templates chosen by the cloud consumer during provisioning. For example, a cloud consumer can set up a pool of mid-tier Windows servers with 4 GB of RAM or a pool of low-tier Ubuntu servers with 2 GB of RAM.

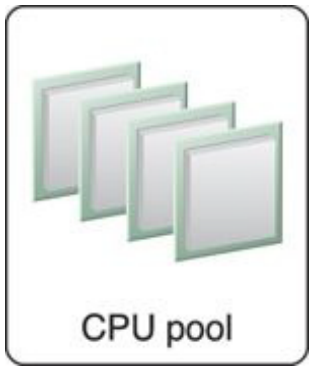
Resource Pooling Architecture



Storage pools, or cloud storage device pools, consist of file-based or block-based storage structures that contain empty and/or filled cloud storage devices

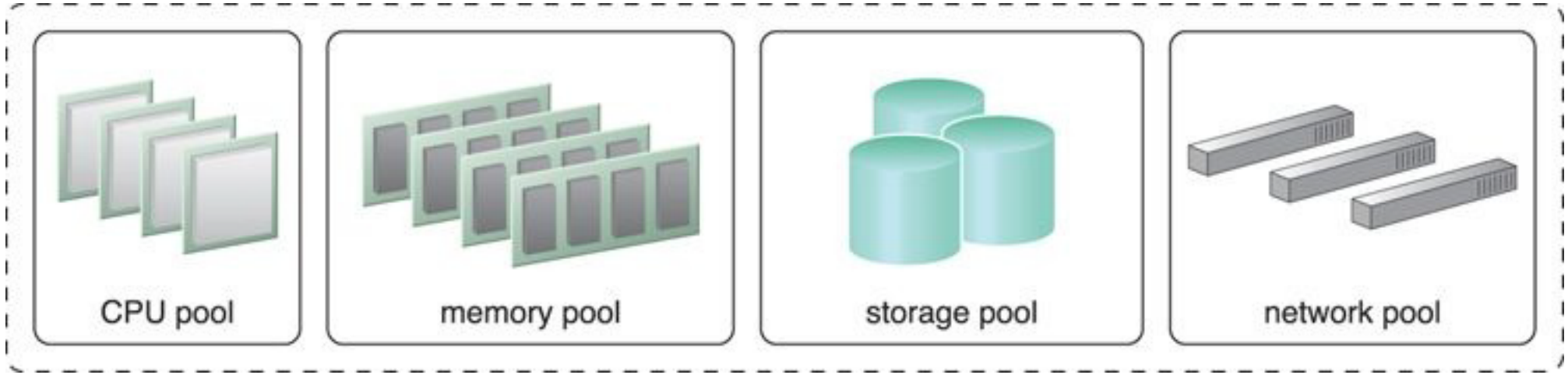


Network pools (or interconnect pools) are composed of different preconfigured network connectivity devices. For example, a pool of virtual firewall devices or physical network switches can be created for redundant connectivity, load balancing, or link aggregation.



CPU pools are ready to be allocated to virtual servers, and are typically broken down into individual processing cores.

Pools of physical RAM can be used in newly provisioned physical servers or to vertically scale physical servers.



Other mechanism part of resource pool architecture

- Audit Monitor
- Cloud usage monitor
- Hypervisor
- Logical network perimeter
- Pay per use monitor
- Remote administrator system
- Resource management system
- Resource replication

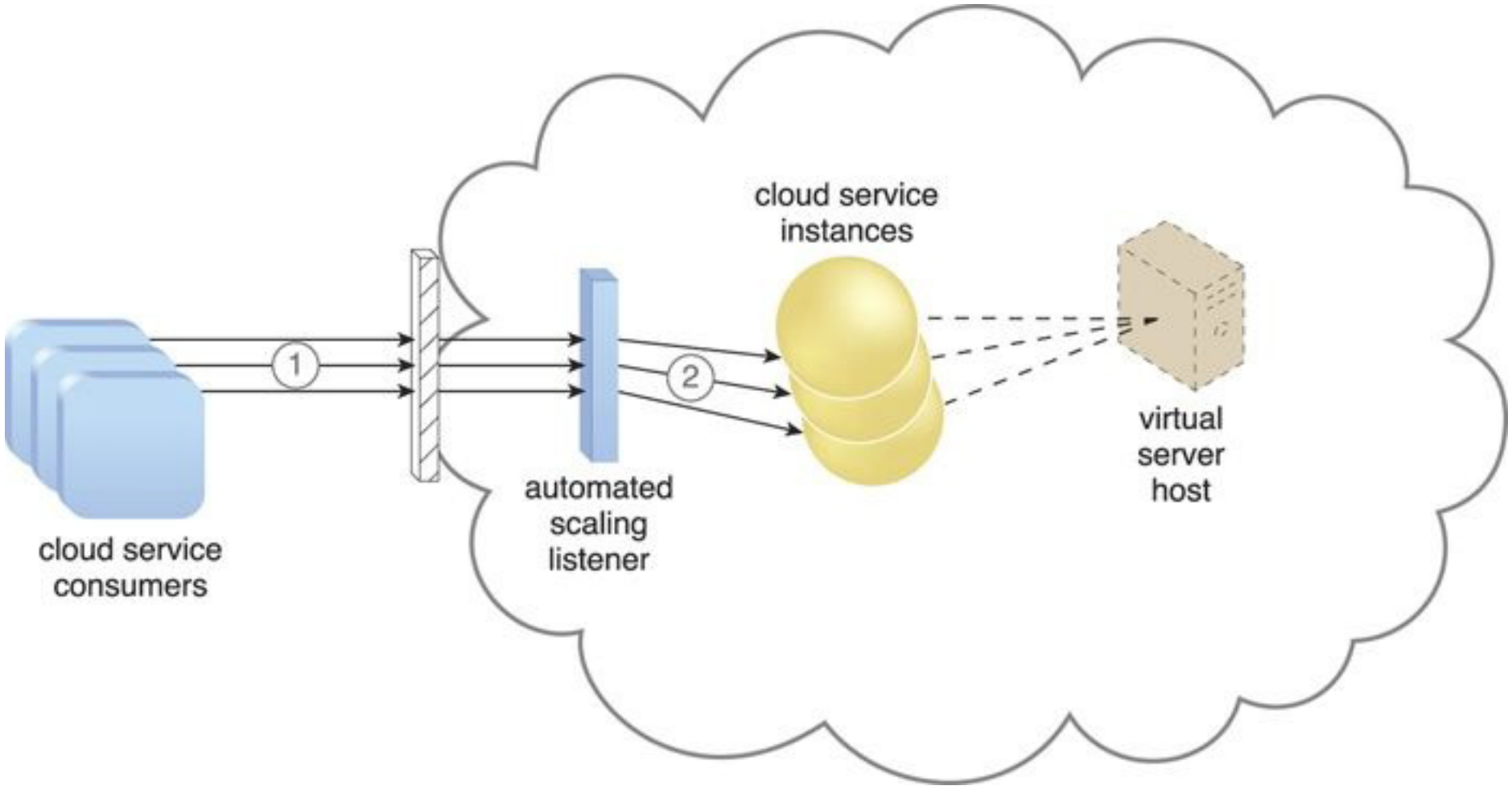
Dynamic Scalability Architecture

- The *dynamic scalability architecture is an architectural model based on a system* of predefined scaling conditions that trigger the dynamic allocation of IT resources from resource pools.
- Dynamic allocation enables variable utilization as dictated by usage demand fluctuations, since unnecessary IT resources are efficiently reclaimed without requiring manual interaction.
- The automated scaling listener is configured with workload thresholds that dictate when new IT resources need to be added to the workload processing.
- This mechanism can be provided with logic that determines how many additional IT resources can be dynamically provided, based on the terms of a given cloud consumer's provisioning contract.

The following types of dynamic scaling are commonly used:

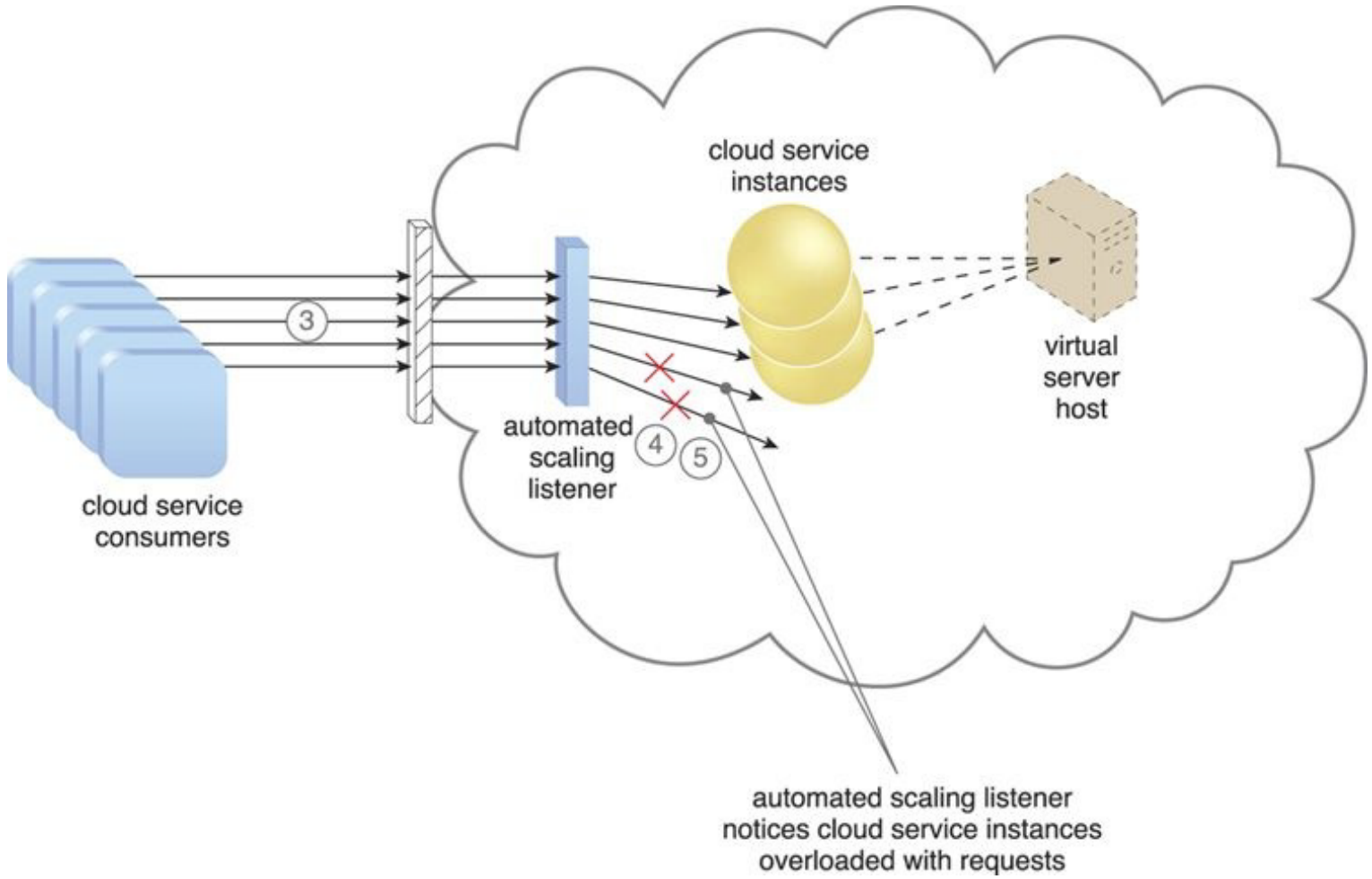
- *Dynamic Horizontal Scaling – IT resource instances are scaled out and in* to handle fluctuating workloads. The automatic scaling listener monitors requests and signals resource replication to initiate IT resource duplication, as per requirements and permissions.
- *Dynamic Vertical Scaling – IT resource instances are scaled up and down* when there is a need to adjust the processing capacity of a single IT resource. For example, a virtual server that is being overloaded can have its memory dynamically increased or it may have a processing core added.
- *Dynamic Relocation – The IT resource is relocated to a host with more*
- capacity. For example, a database may need to be moved from a tape-based SAN storage device with 4 GB per second I/O capacity to another disk based SAN storage device with 8 GB per second I/O capacity.

Process of Dynamic Scaling



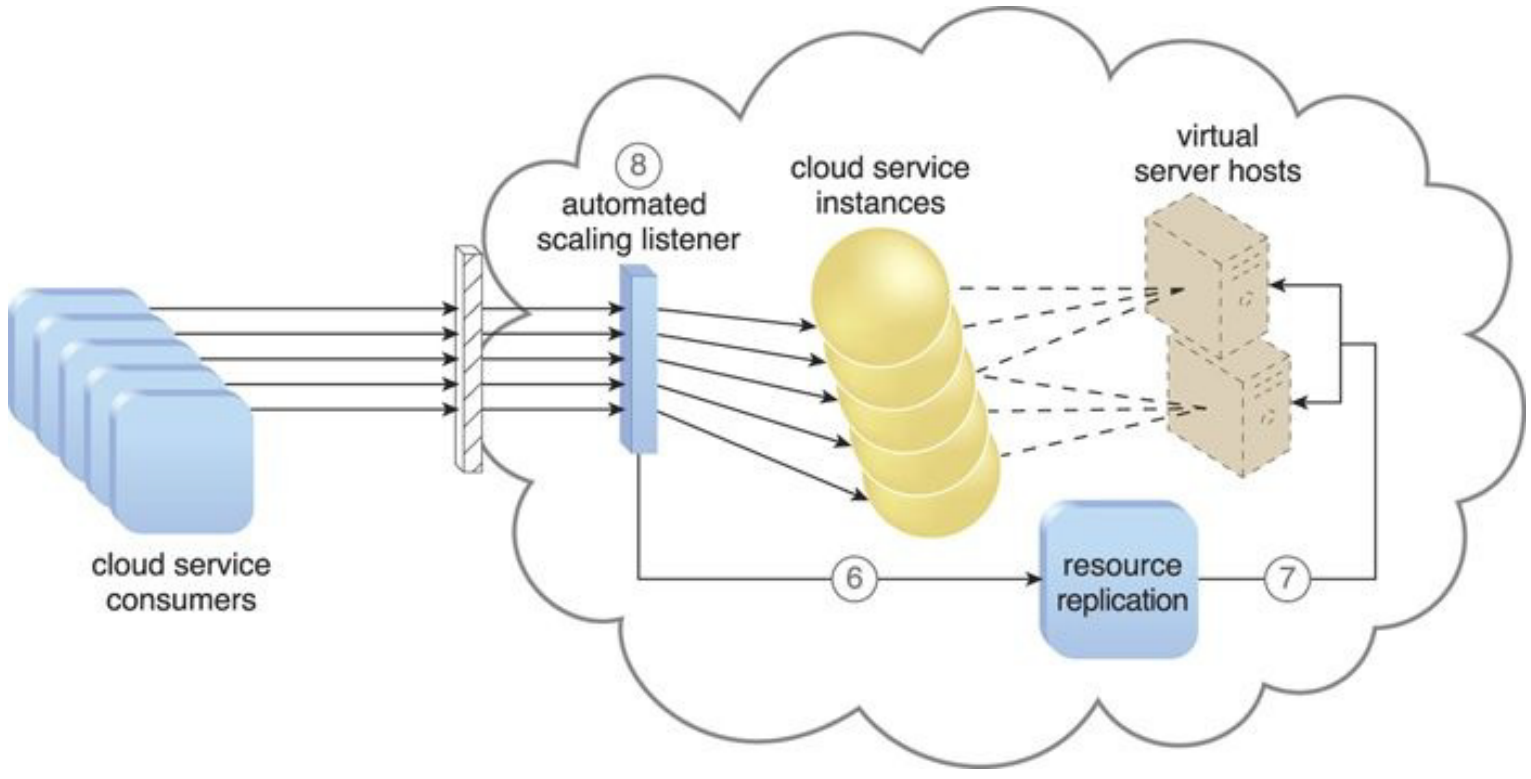
Cloud service consumers are sending requests to a cloud service
(1). The automated scaling listener monitors the cloud service to determine if predefined capacity thresholds are being exceeded (2).

Process of Dynamic Scaling



The number of requests coming from cloud service consumers increases (3).
The workload exceeds the performance thresholds. The automated scaling listener determines the next course of action based on a predefined scaling policy (4).
If the cloud service implementation is deemed eligible for additional scaling, the automated scaling listener initiates the scaling process (5).

Process of Dynamic Scaling



The automated scaling listener sends a signal to the resource replication mechanism (6), which creates more instances of the cloud service (7).
Now that the increased workload has been accommodated, the automated scaling listener resumes monitoring and detracting and adding IT resources, as required (8).

The dynamic scalability architecture can be applied to a range of IT resources, including virtual servers and cloud storage devices.

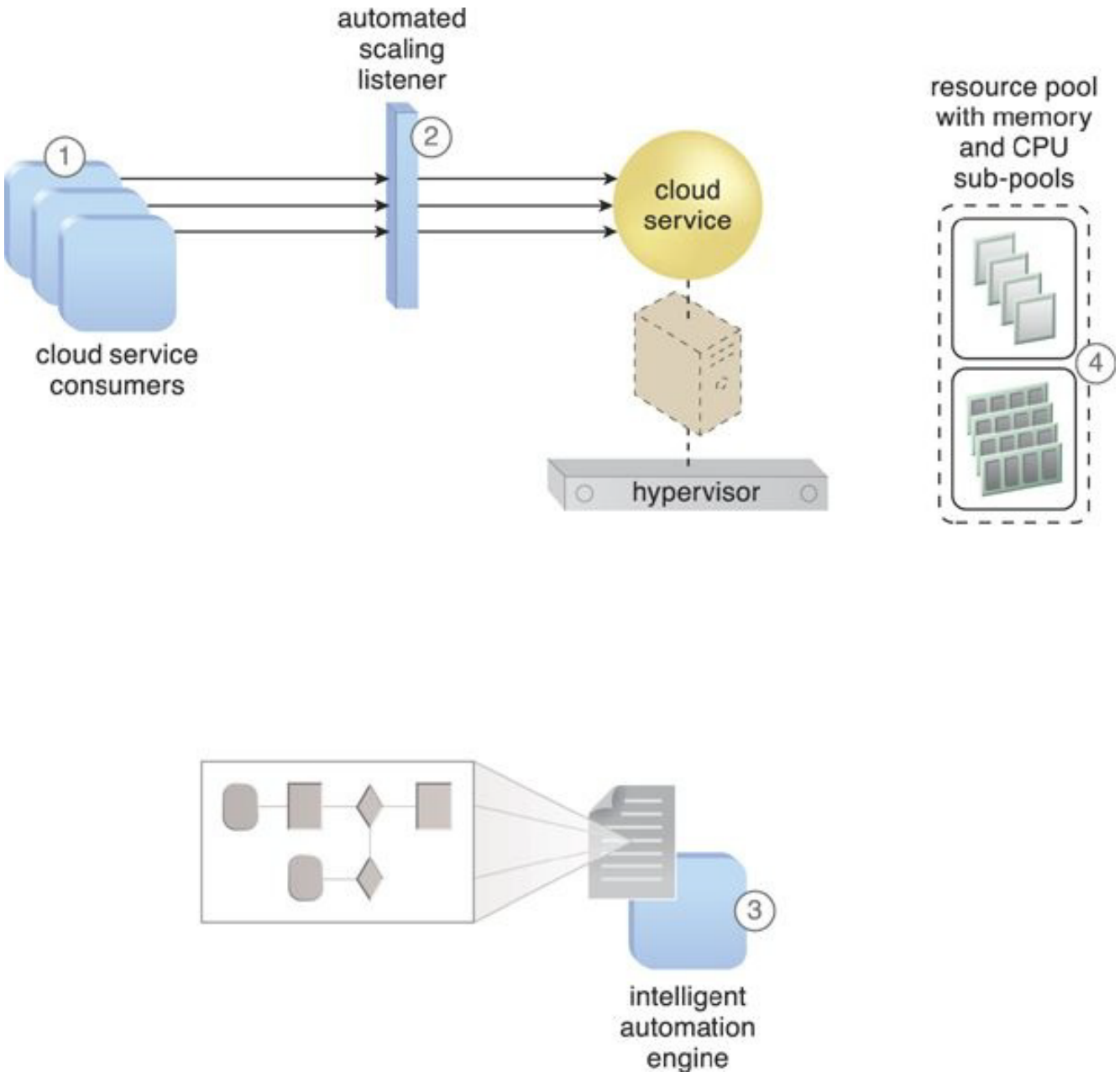
Besides the core automated scaling listener and resource replication mechanisms, the following mechanisms can also be used in this form of cloud architecture:

- *Cloud Usage Monitor – Specialized cloud usage monitors can track* runtime usage in response to dynamic fluctuations caused by this architecture.
- *Hypervisor – The hypervisor is invoked by a dynamic scalability system to* create or remove virtual server instances, or to be scaled itself.
- *Pay-Per-Use Monitor – The pay-per-use monitor is engaged to collect* usage cost information in response to the scaling of IT resources.

Elastic Resource Capacity Architecture

- The *elastic resource capacity architecture is primarily related to the dynamic* provisioning of virtual servers, using a system that allocates and reclaims CPUs and RAM in immediate response to the fluctuating processing requirements of hosted IT resources

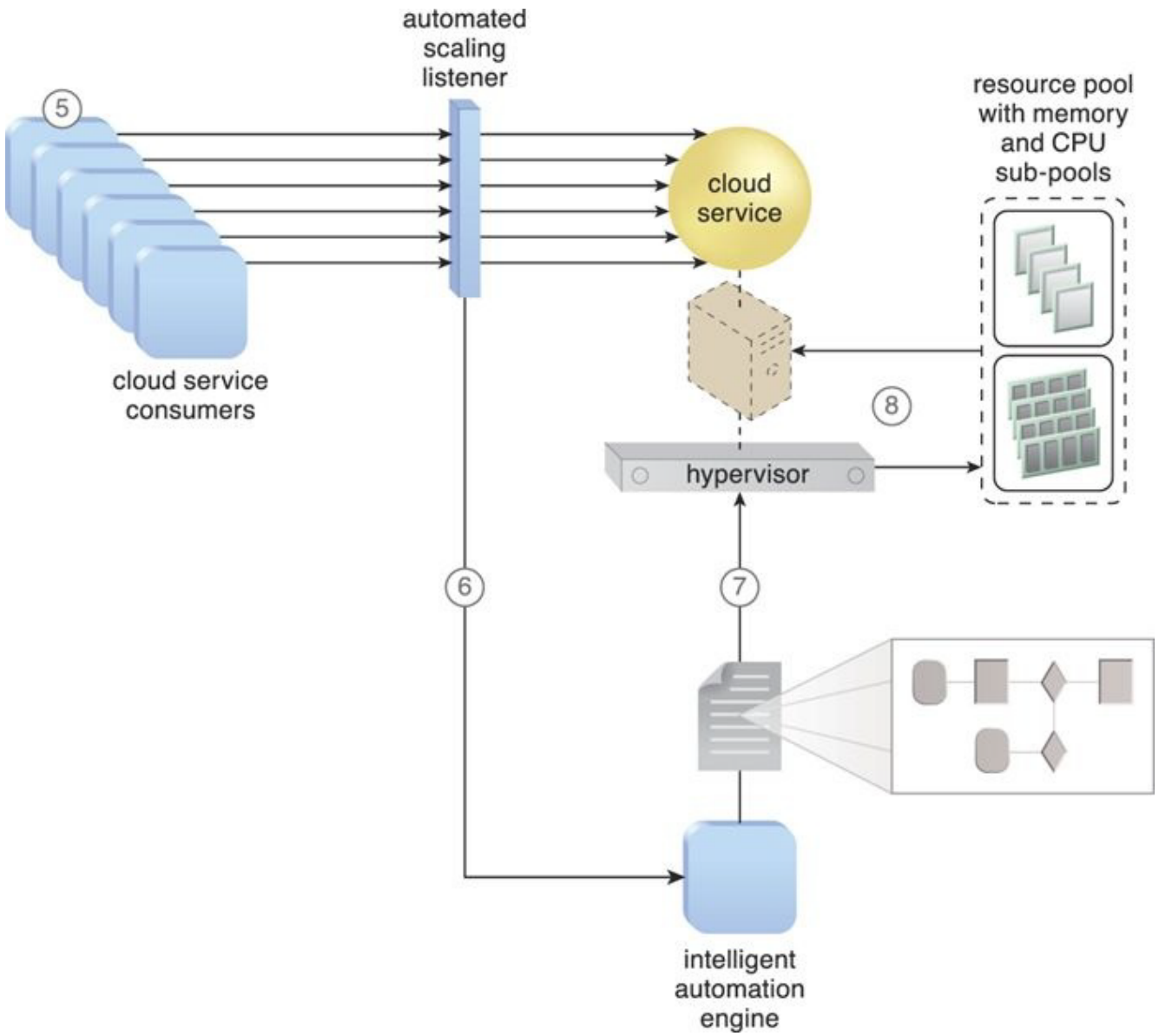
Elastic Resource Capacity Architecture



Cloud service consumers are actively sending requests to a cloud service (1), which are monitored by an automated scaling listener (2).

An intelligent automation engine script is deployed with workflow logic (3) that is capable of notifying the resource pool using allocation requests (4).

Elastic Resource Capacity Architecture



Cloud service consumer requests increase (5), causing the automated scaling listener to signal the intelligent automation engine to execute the script (6).

The script runs the workflow logic that signals the hypervisor to allocate more IT resources from the resource pools (7).

The hypervisor allocates additional CPU and RAM to the virtual server, enabling the increased workload to be handled (8).

Elastic Resource Capacity Architecture

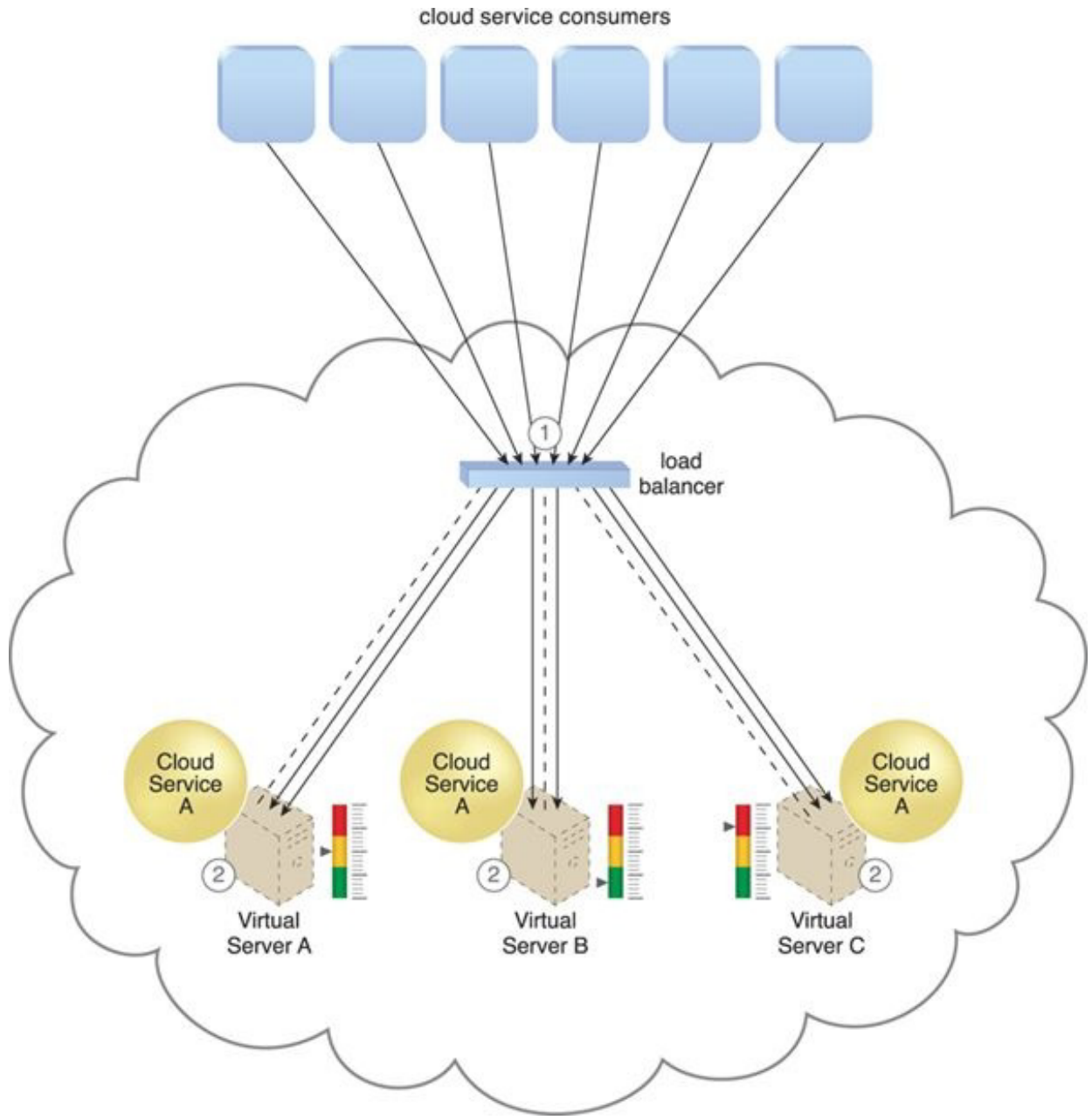
- Resource pools are used by scaling technology that interacts with the hypervisor and/or VIM to retrieve and return CPU and RAM resources at runtime.
- The runtime processing of the virtual server is monitored so that additional processing power can be leveraged from the resource pool via dynamic allocation, before capacity thresholds are met.
- The virtual server and its hosted applications and IT resources are vertically scaled in response.
- This type of cloud architecture can be designed so that the intelligent automation engine script sends its scaling request via the VIM instead of to the hypervisor directly.
- Virtual servers that participate in elastic resource allocation systems may require rebooting in order for the dynamic resource allocation to take effect.

- Some additional mechanisms that can be included in this cloud architecture are the following:
- *Cloud Usage Monitor – Specialized cloud usage monitors collect resource* usage information on IT resources before, during, and after scaling, to help define the future processing capacity thresholds of the virtual servers.
- *Pay-Per-Use Monitor – The pay-per-use monitor is responsible for* collecting resource usage cost information as it fluctuates with the elastic provisioning.
- *Resource Replication – Resource replication is used by this architectural* model to generate new instances of the scaled IT resources.

Service Load Balancing Architecture

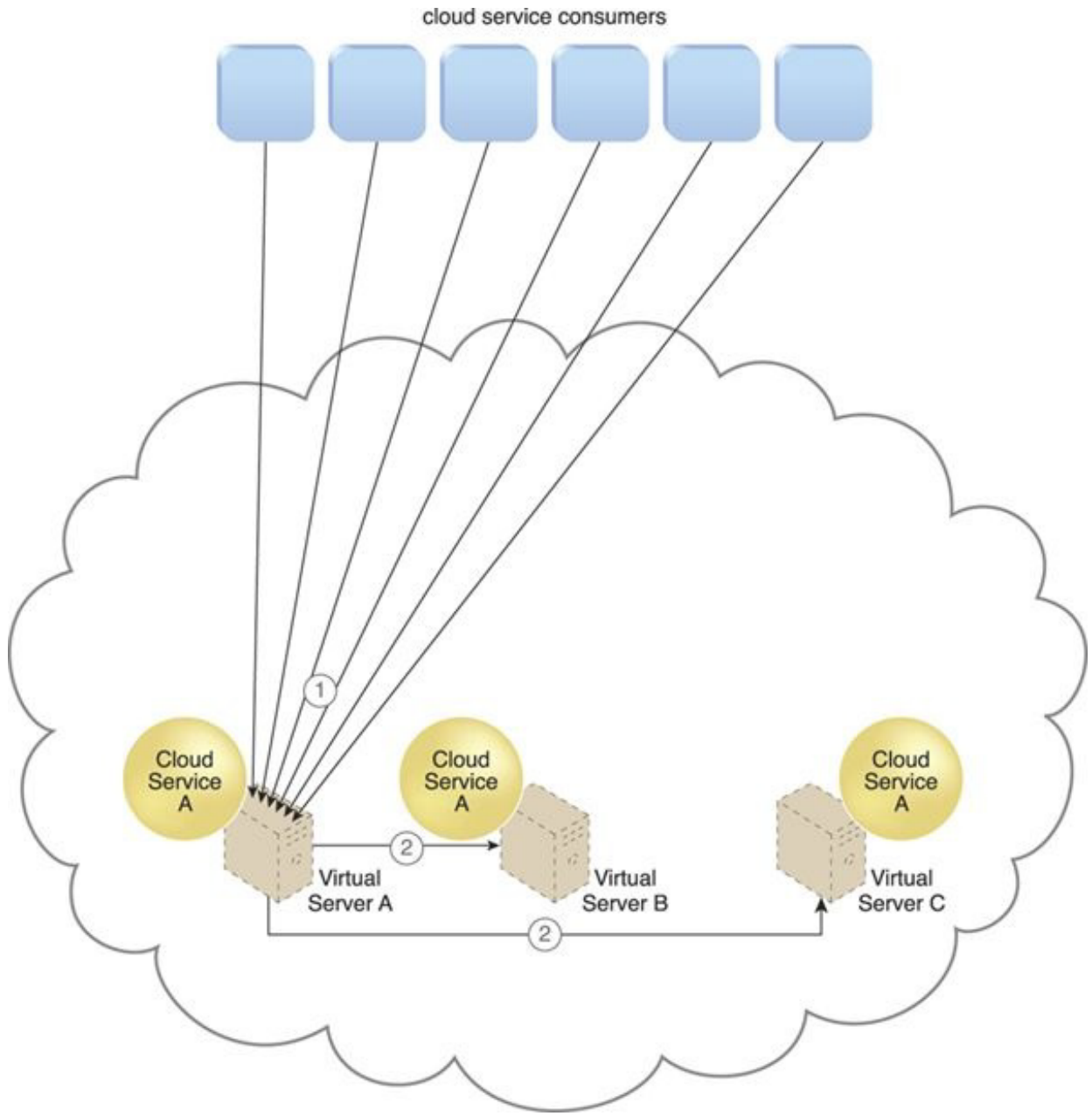
- The *service load balancing architecture can be considered a specialized* variation of the workload distribution architecture that is geared specifically for scaling cloud service implementations.
- Redundant deployments of cloud services are created, with a load balancing system added to dynamically distribute workloads.
- The duplicate cloud service implementations are organized into a resource pool, while the load balancer is positioned as either an external or built-in component to allow the host servers to balance the workloads themselves.
- Depending on the anticipated workload and processing capacity of host server environments, multiple instances of each cloud service implementation can be generated as part of a resource pool that responds to fluctuating request volumes more efficiently.
- The load balancer can be positioned either independent of the cloud services and their host servers (Figure 11.10), or built-in as part of the application or server's environment.
- In the latter case, a primary server with the load balancing logic can communicate with neighboring servers to balance the workload

load balancer can be positioned independent of the cloud services and their host servers



The load balancer intercepts messages sent by cloud service consumers (1) and forwards them to the virtual servers so that the workload processing is horizontally scaled (2).

a primary server with the load balancing logic can communicate with neighboring servers to balance the workload



Cloud service consumer requests are sent to Cloud Service A on Virtual Server A (1). The cloud service implementation includes built-in load balancing logic that is capable of distributing requests to the neighboring Cloud Service A implementations on Virtual Servers B and C (2).

The service load balancing architecture can involve the following mechanisms in addition to the load balancer:

Cloud Usage Monitor – Cloud usage monitors may be involved with monitoring cloud service instances and their respective IT resource consumption levels, as well as various runtime monitoring and usage data collection tasks.

Resource Cluster – Active-active cluster groups are incorporated in this architecture to help balance workloads across different members of the cluster.

Resource Replication – The resource replication mechanism is utilized to generate cloud service implementations in support of load balancing requirements.

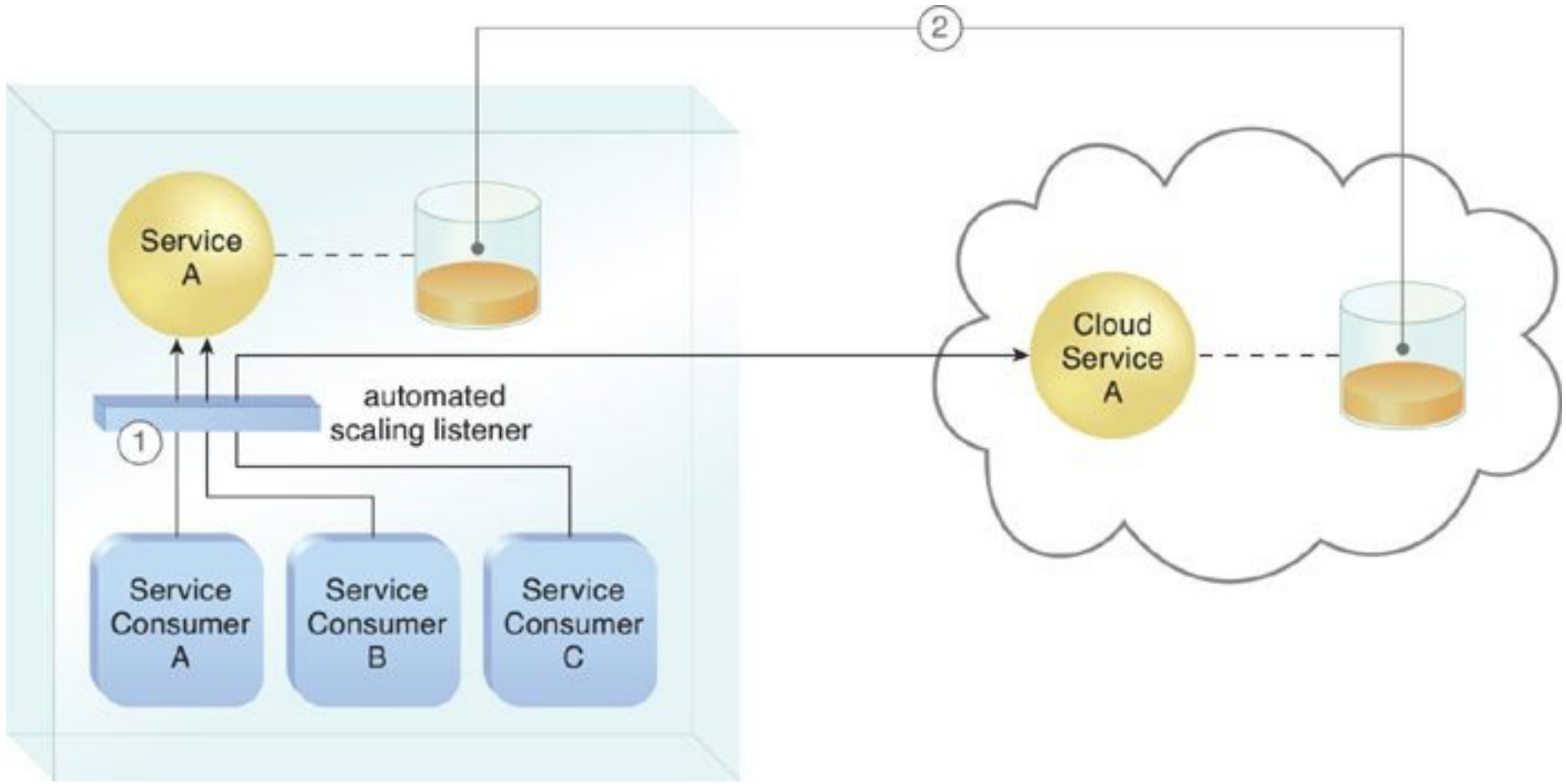
Cloud Bursting Architecture

- 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 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 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

Automated Scaling Listener and Resource Replication system

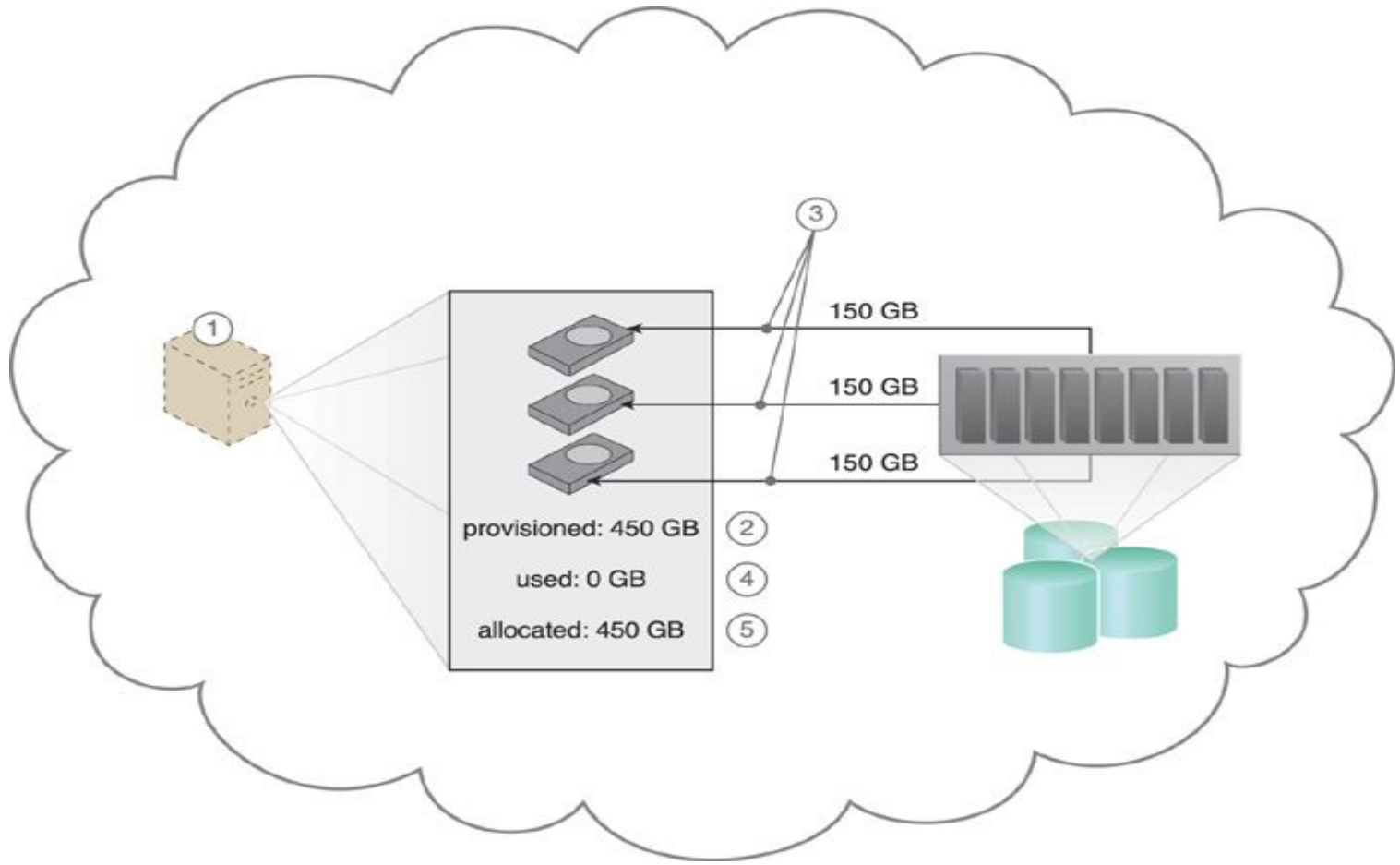


An automated scaling listener monitors the usage of on-premise Service A, and redirects Service Consumer C's request to Service A's redundant implementation in the cloud (Cloud Service A) once Service A's usage threshold has been exceeded (1). A resource replication system is used to keep state management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

Elastic Disk Provisioning Architecture

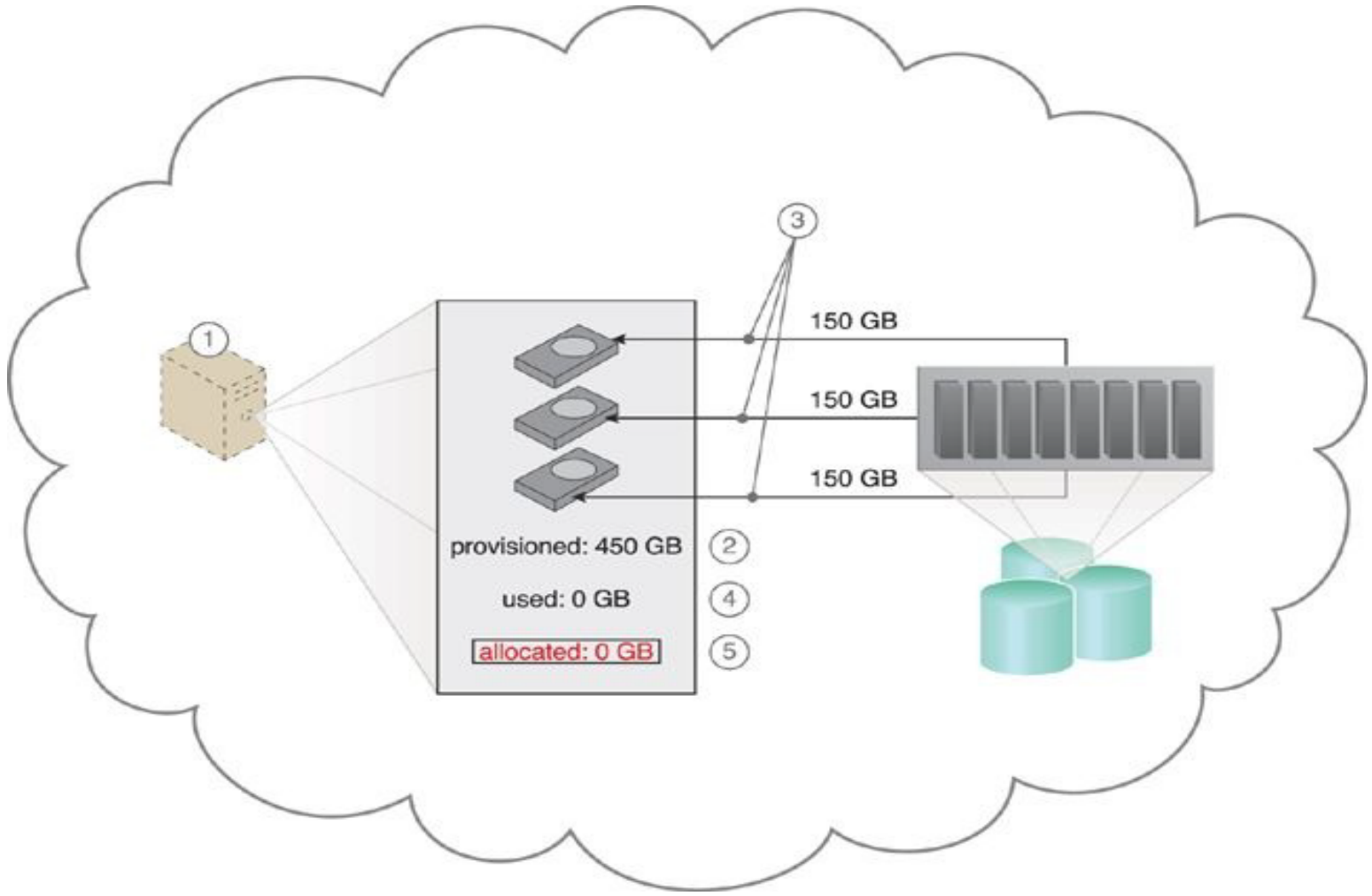
- Cloud consumers are commonly charged for cloud-based storage space based on fixed-disk storage allocation, meaning the charges are predetermined by disk capacity and not aligned with actual data storage consumption.
- The *elastic disk provisioning architecture establishes a dynamic storage* provisioning 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

Cloud Storage Allocation without **Elastic Disk Provisioning Architecture** - Example



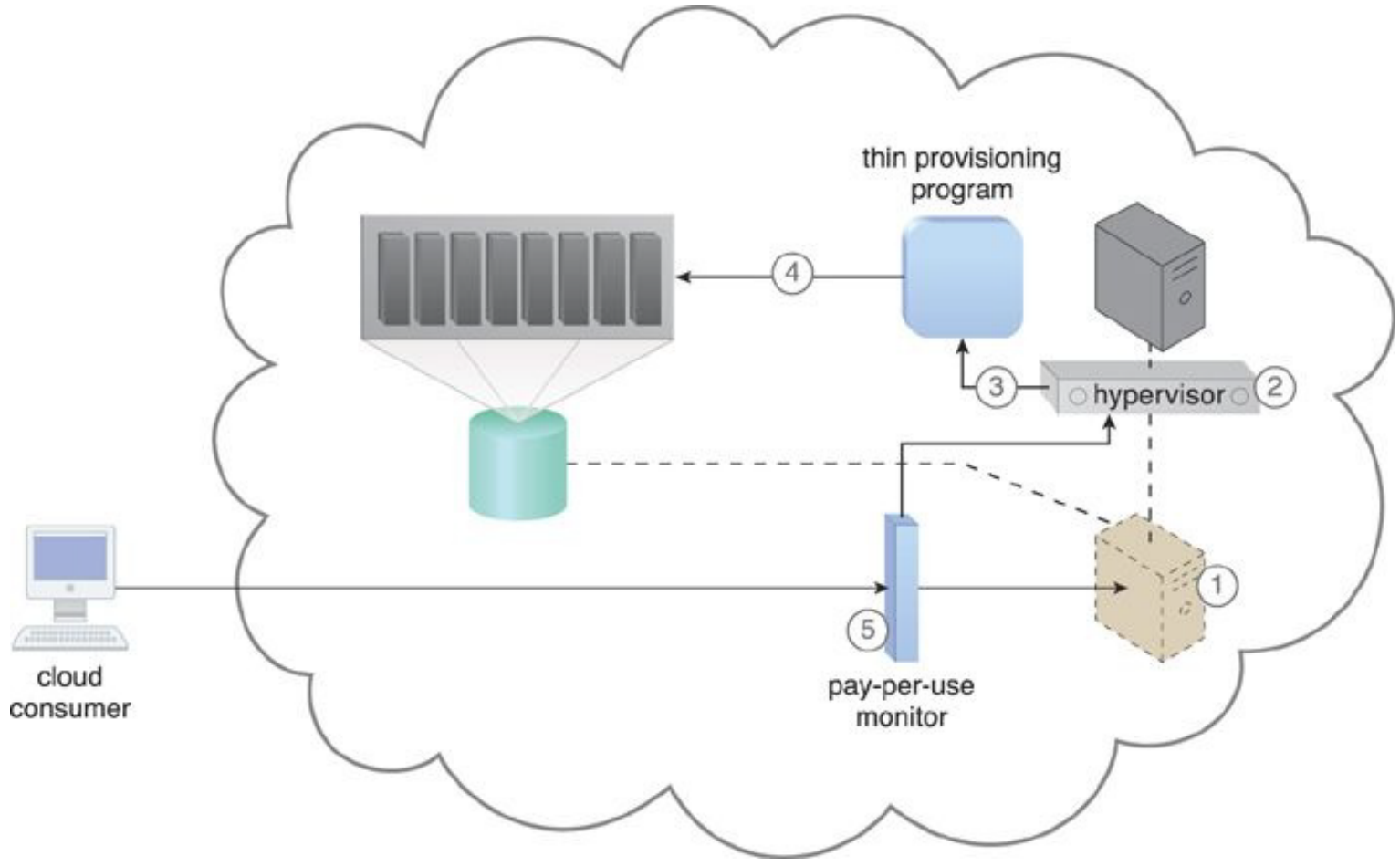
The cloud consumer requests a virtual server with three hard disks, each with a capacity of 150 GB (1). The virtual server is provisioned according to the elastic disk provisioning architecture, with a total of 450 GB of disk space (2). The 450 GB is allocated to the virtual server by the cloud provider (3). The cloud consumer has not installed any software yet, meaning the actual used space is currently 0 GB (4). Because the 450 GB are already allocated and reserved for the cloud consumer, it will be charged for 450 GB of disk usage as of the point of allocation (5).

Cloud Storage Allocation with **Elastic Disk Provisioning Architecture** - Example



The cloud consumer requests a virtual server with three hard disks, each with a capacity of 150 GB (1). The virtual server is provisioned by this architecture with a total of 450 GB of disk space (2). The 450 GB are set as the maximum disk usage that is allowed for this virtual server, although no physical disk space has been reserved or allocated yet (3). The cloud consumer has not installed any software, meaning the actual used space is currently at 0 GB (4). Because the allocated disk space is equal to the actual used space (which is currently at zero), the cloud consumer is not charged for any disk space usage (5).

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



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. The size of this folder and its files grow as operating applications are installed and additional files are copied onto the virtual server (4). The pay-per-use monitor tracks the actual dynamically allocated storage for billing purposes (5).

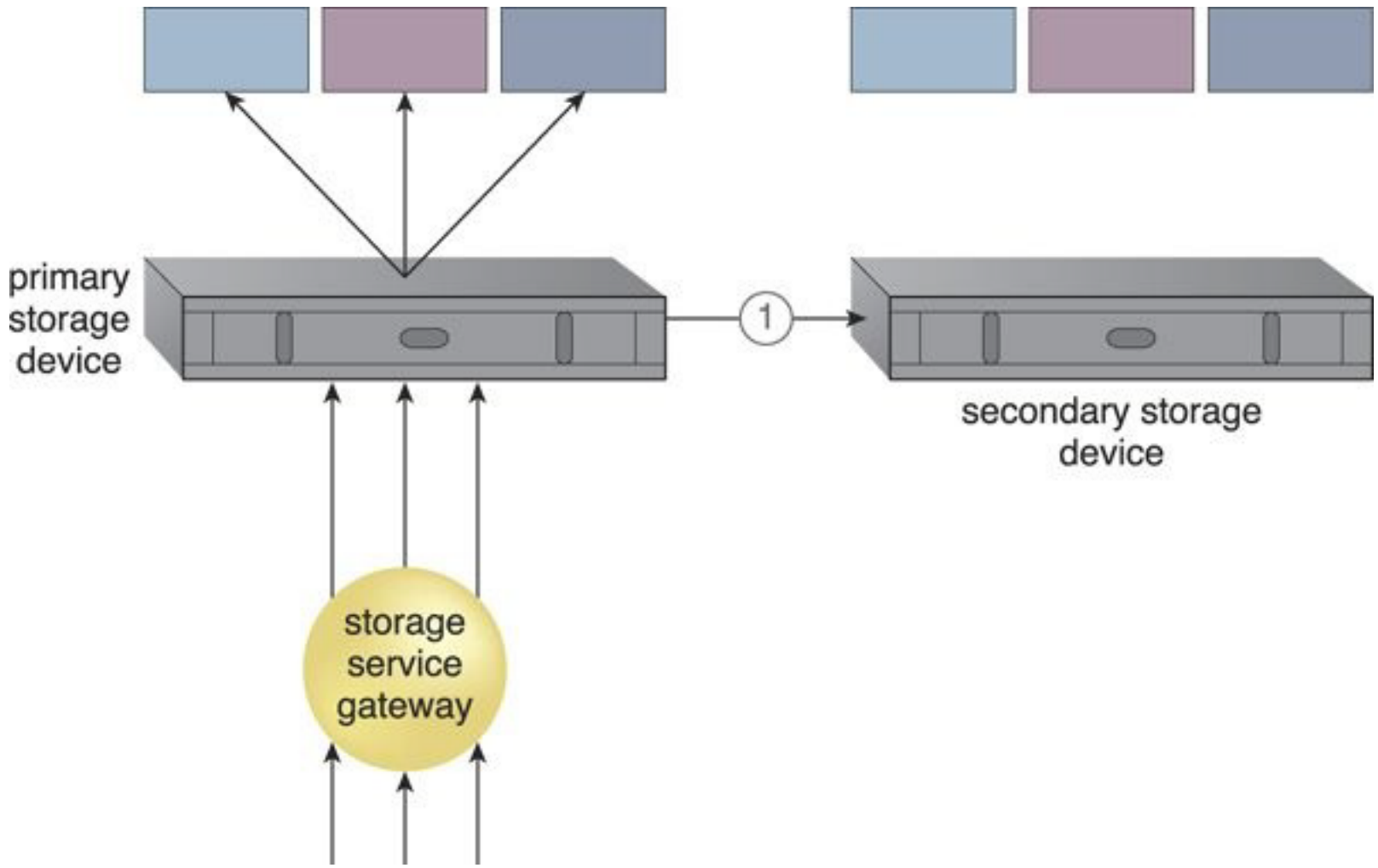
The following mechanisms can be included in this architecture in addition to the cloud storage device, virtual server, hypervisor, and pay-per-use monitor:

- *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.

Redundant Storage Architecture

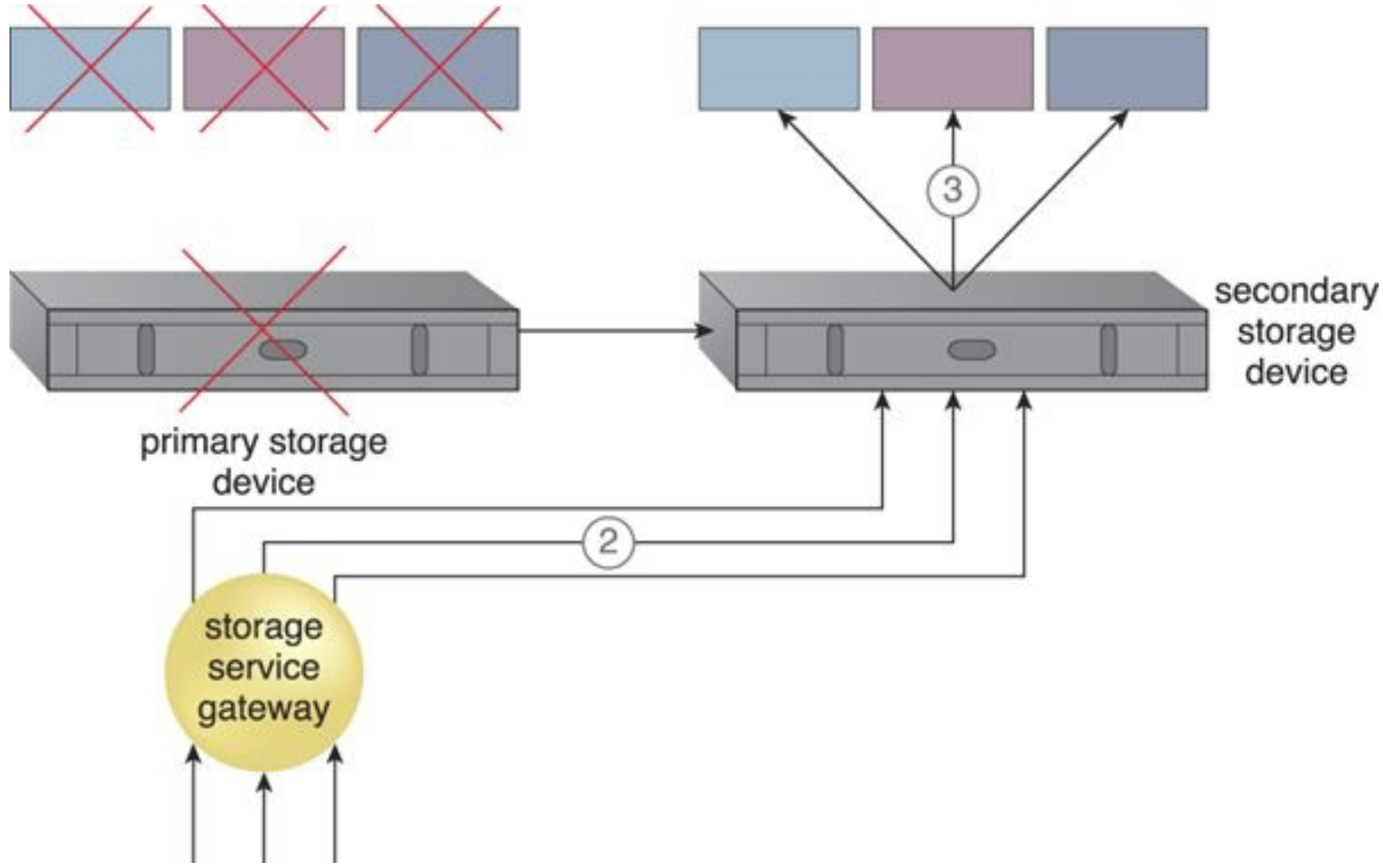
- Cloud storage devices are occasionally subject to failure and disruptions that are caused by network connectivity issues, controller or general hardware failure, or security breaches.
- A compromised cloud storage device's reliability can have a ripple effect and cause impact failure across all of the services, applications, and infrastructure components in the cloud that are reliant on its availability.
- The *redundant storage architecture introduces a secondary duplicate cloud* storage device as part of a failover system that synchronizes its data with the data in the primary cloud storage device.
- A storage service gateway diverts cloud consumer requests to the secondary device whenever the primary device fails

Redundant Storage Architecture



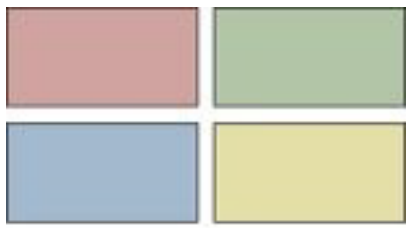
The primary cloud storage device is routinely replicated to the secondary cloud storage device (1).

Redundant Storage Architecture



The primary storage becomes unavailable and the storage service gateway forwards the cloud consumer requests to the secondary storage device (2). The secondary storage device forwards the requests to the LUNs, allowing cloud consumers to continue to access their data (3).

Redundant Storage Architecture



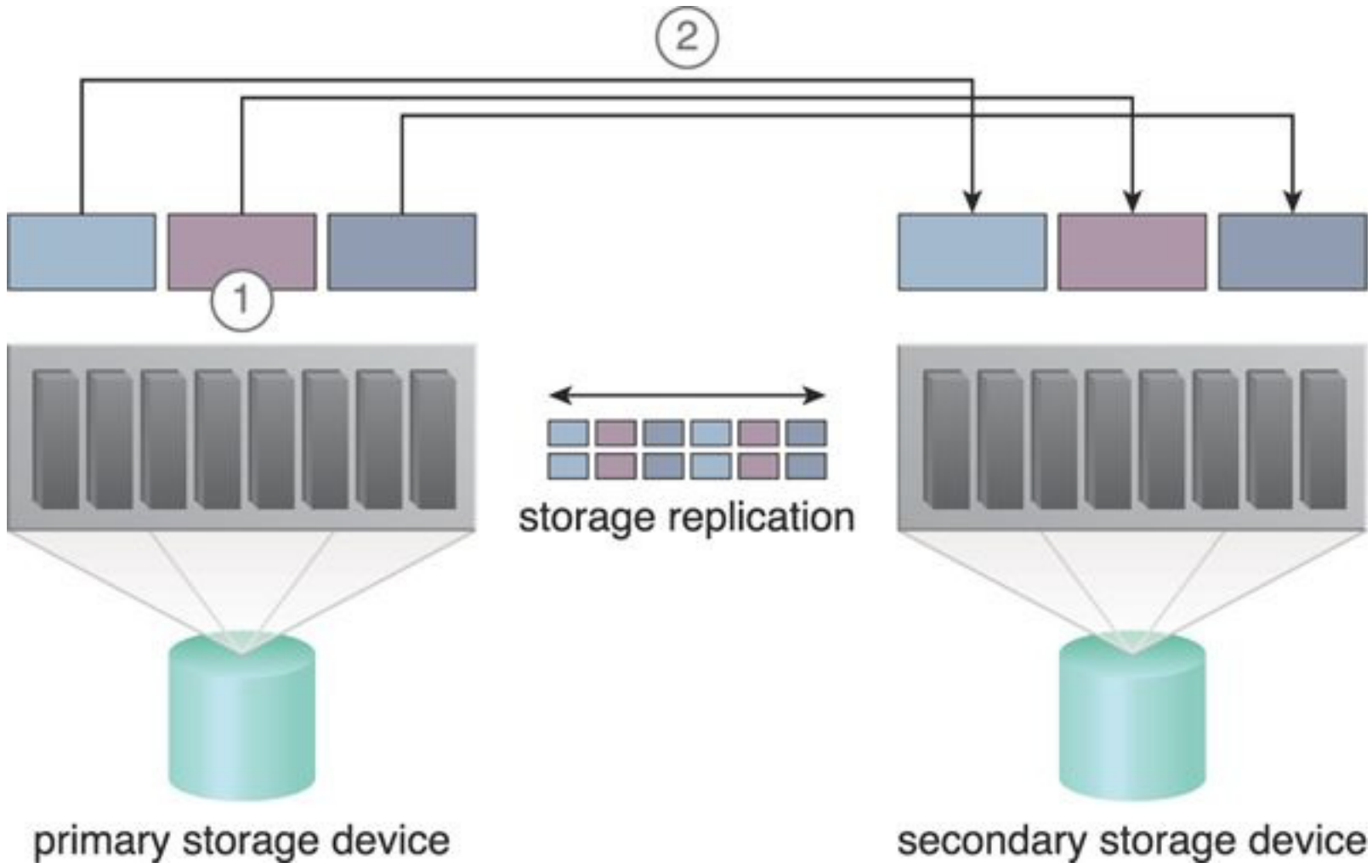
LUNs

A logical unit number (LUN) is a logical drive that represents a partition of a physical drive.

The storage service gateway is a component that acts as the external interface to cloud storage services, and is capable of automatically redirecting cloud consumer requests whenever the location of the requested data has changed.

Redundant Storage Architecture

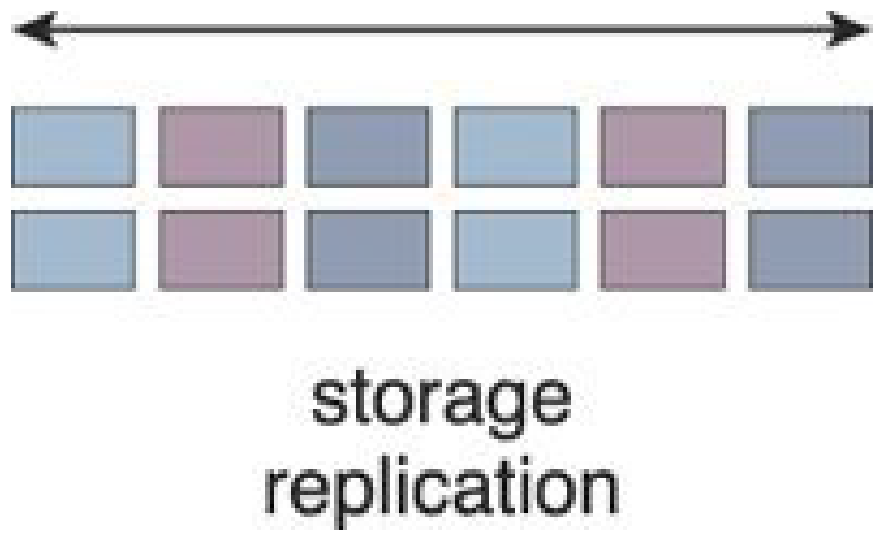
This cloud architecture primarily relies on a storage replication system that keeps the primary cloud storage device synchronized with its duplicate secondary cloud storage devices



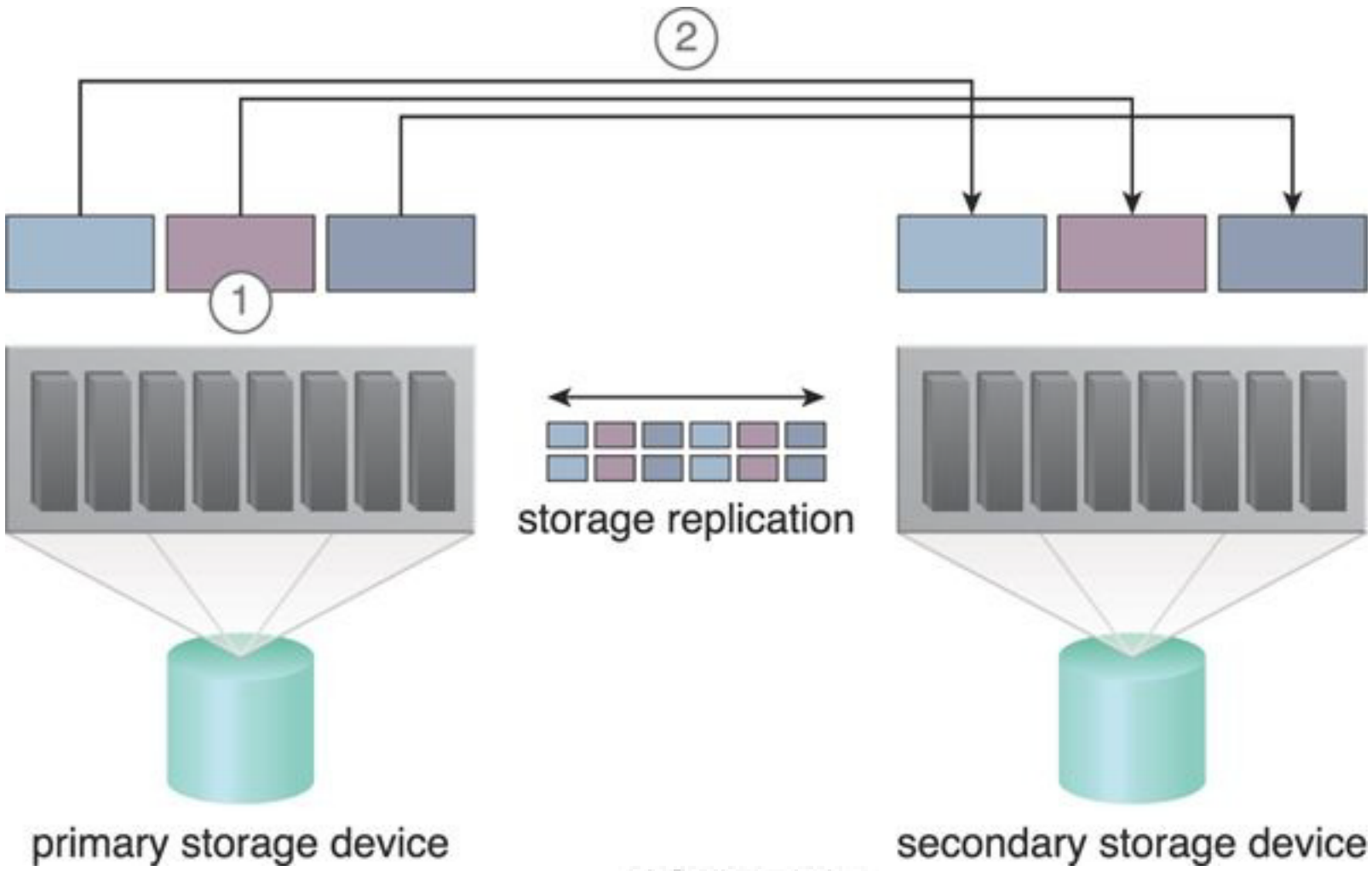
Storage replication is used to keep the redundant storage device synchronized with the primary storage device.

Storage Replication

- Storage replication is a variation of the resource replication mechanisms used to synchronously or asynchronously replicate data from a primary storage device to a secondary storage device. It can be used to replicate partial and entire LUNs.



Storage Replication



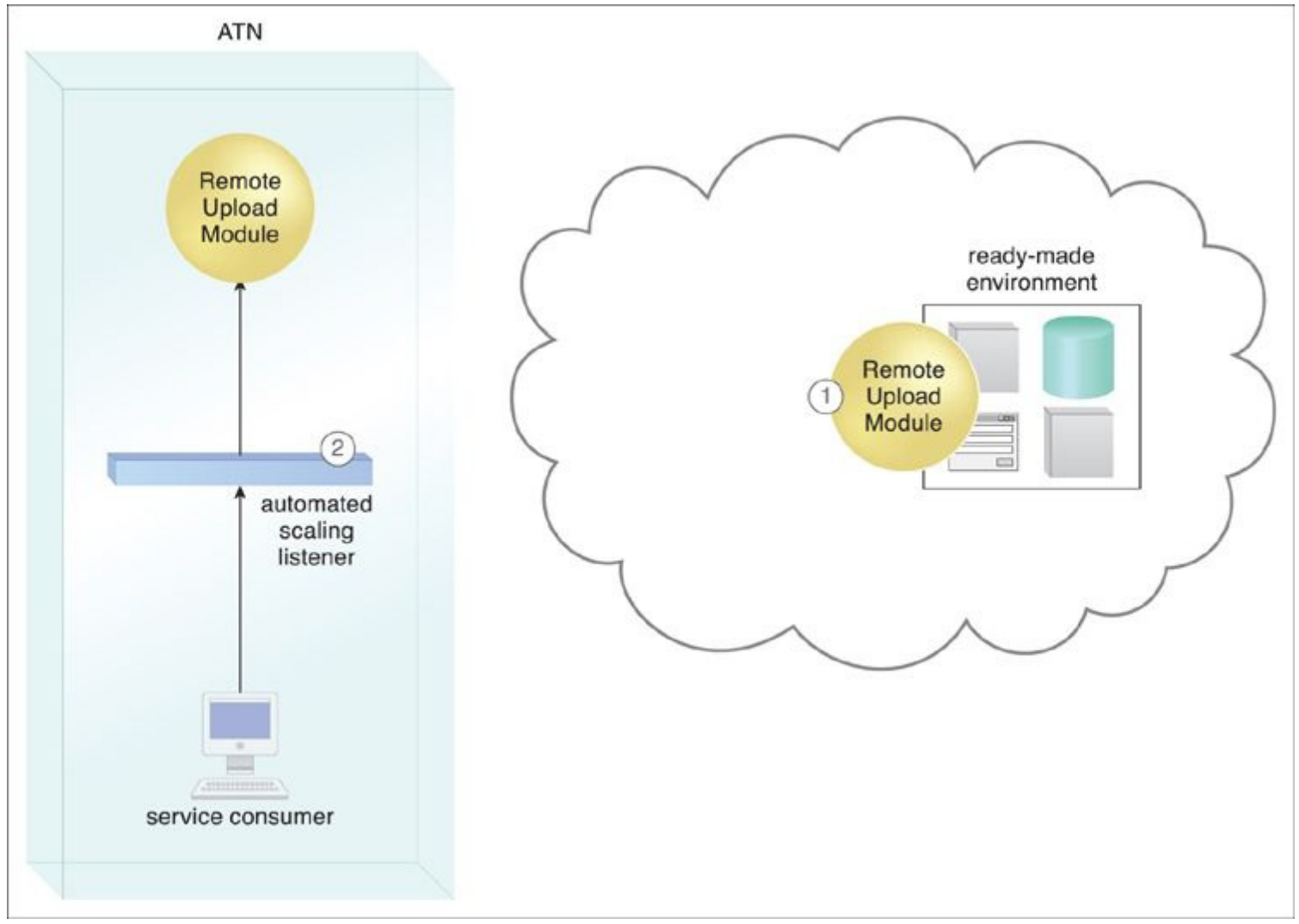
storage
replication

Storage replication is used to keep the redundant storage device synchronized with the primary storage device.

Storage Replication

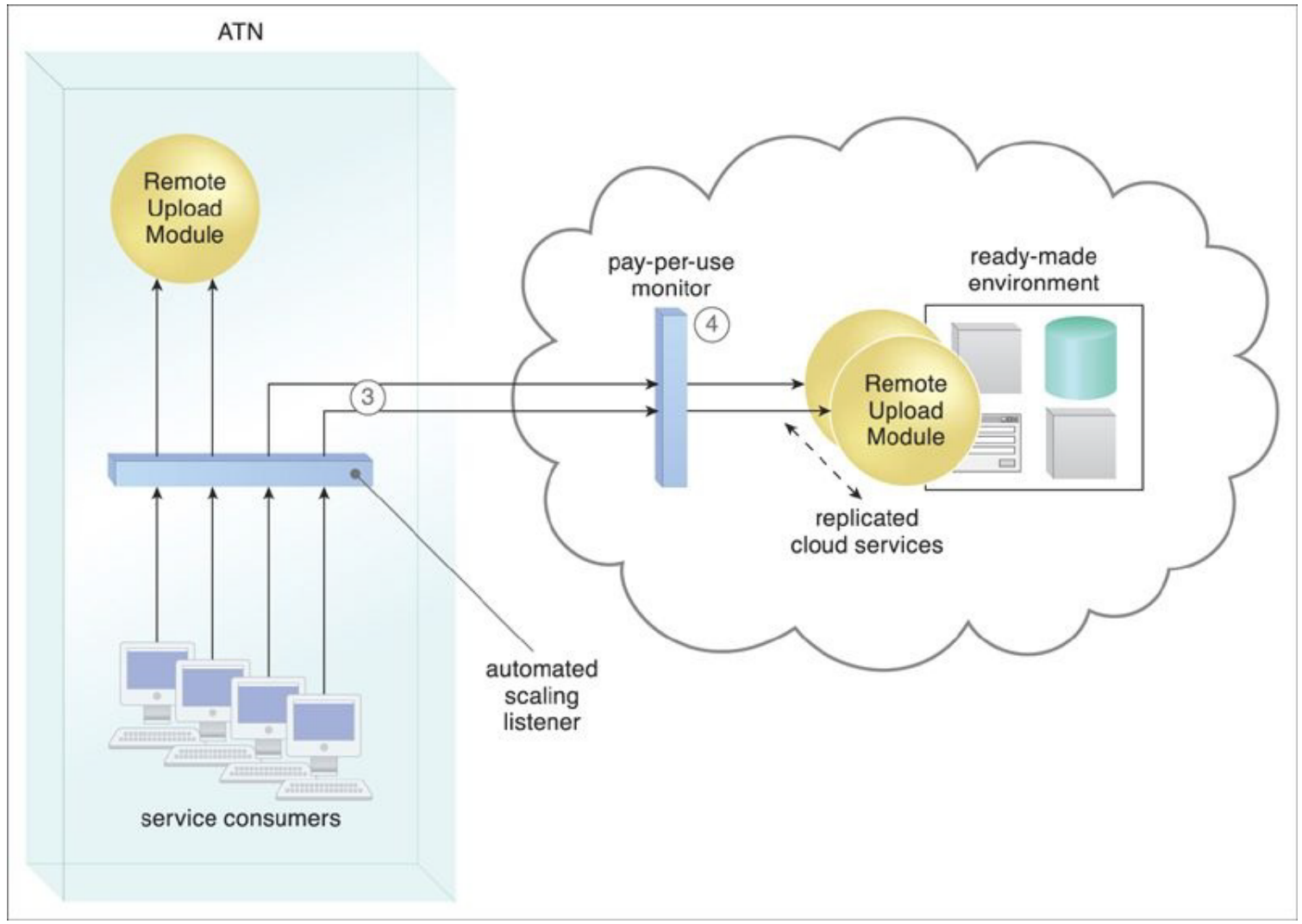
- Cloud providers may locate secondary cloud storage devices in a different geographical region than the primary cloud storage device, usually for economic reasons.
- However, this can introduce legal concerns for some types of data. The location of the secondary cloud storage devices can dictate the protocol and method used for synchronization, as some replication transport protocols have distance restrictions.
- Some cloud providers use storage devices with dual array and storage controllers to improve device redundancy, and place secondary storage devices in a different physical location for cloud balancing and disaster recovery purposes.
- In this case, cloud providers may need to lease a network connection via a third-party cloud provider in order to establish the replication between the two devices.

Storage Replication



A cloud-based version of the on-premise Remote Upload Module service is deployed on ATN's leased ready-made environment (1).
The automated scaling listener monitors service consumer requests (2).

Storage Replication



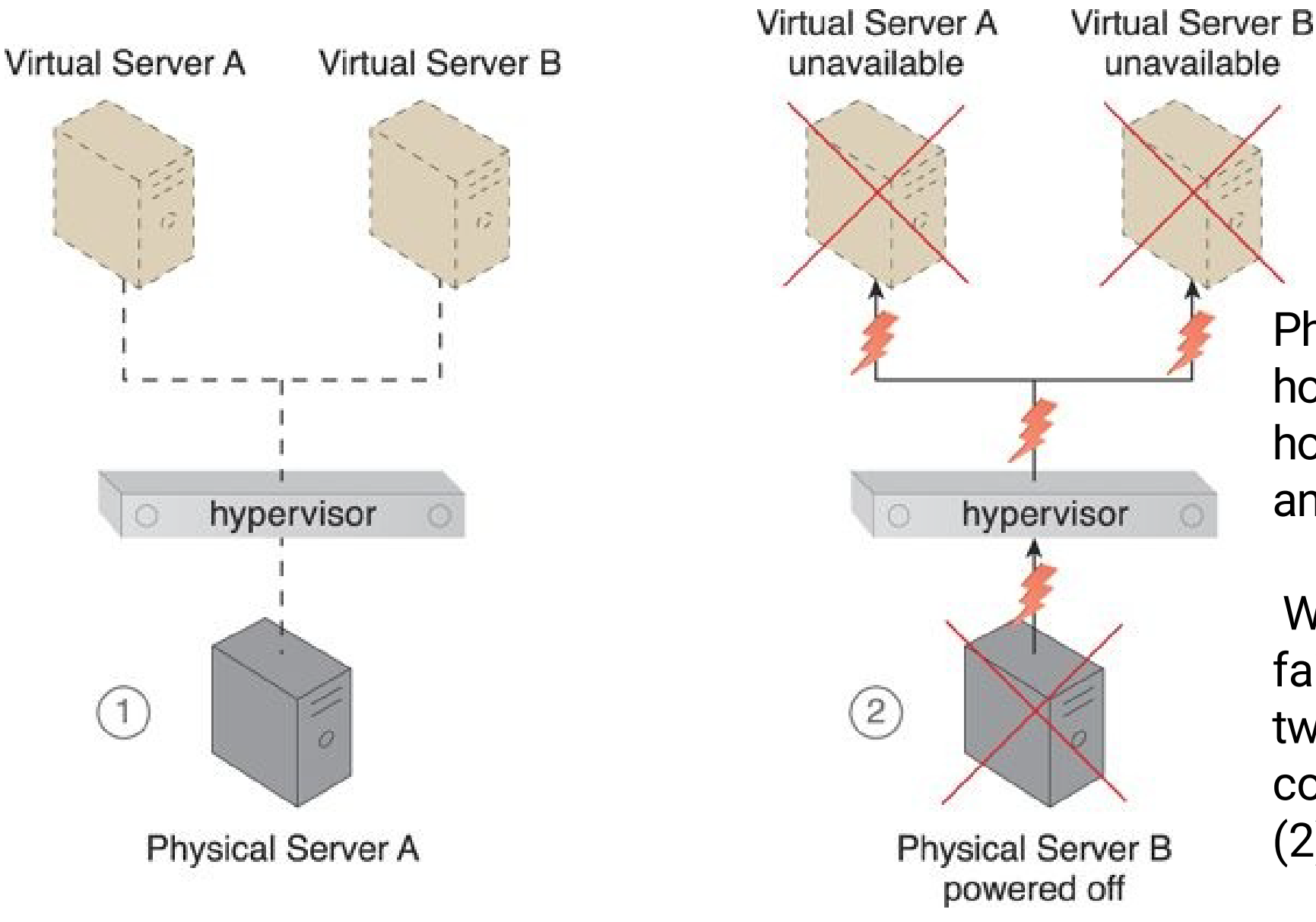
A cloud-based version of the on-premise Remote Upload Module service is deployed on ATN's leased ready-made environment (1).
The automated scaling listener monitors service consumer requests (2)
The automated scaling listener detects that service consumer usage has exceeded the local Remote Upload Module service's usage threshold, and begins diverting excess requests to the cloud-based Remote Upload Module implementation (3).
The cloud provider's pay-per-use monitor tracks the requests received from the on-premise automated scaling listener to collect billing data, and Remote Upload Module cloud service instances are created on demand via resource replication (4).

Advanced Cloud Architectures

- Hypervisor Clustering Architecture
- Load Balanced Virtual Server Instances Architecture
- Dynamic Failure Detection and Recovery Architecture

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).

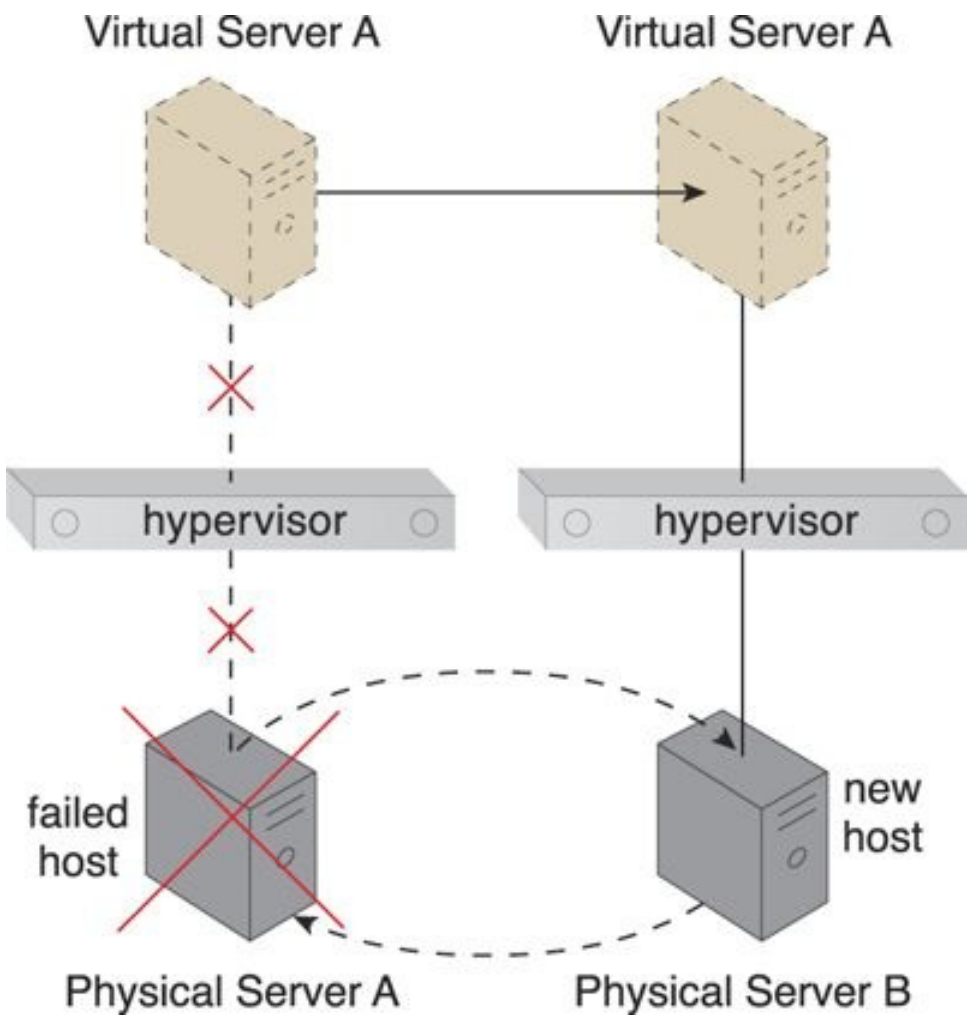
When Physical Server A fails, the hypervisor and two virtual servers consequently fail as well (2).

Hypervisor Clustering Architecture

Heartbeats

- 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.
- 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

Hypervisor Clustering Architecture

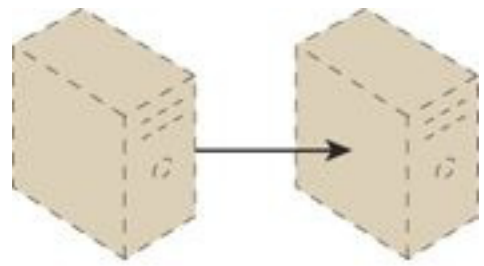


Physical Server A becomes unavailable and causes its hypervisor to fail.
Virtual Server A is migrated to Physical Server B, which has another hypervisor that is part of the cluster to which Physical Server A belongs.

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.

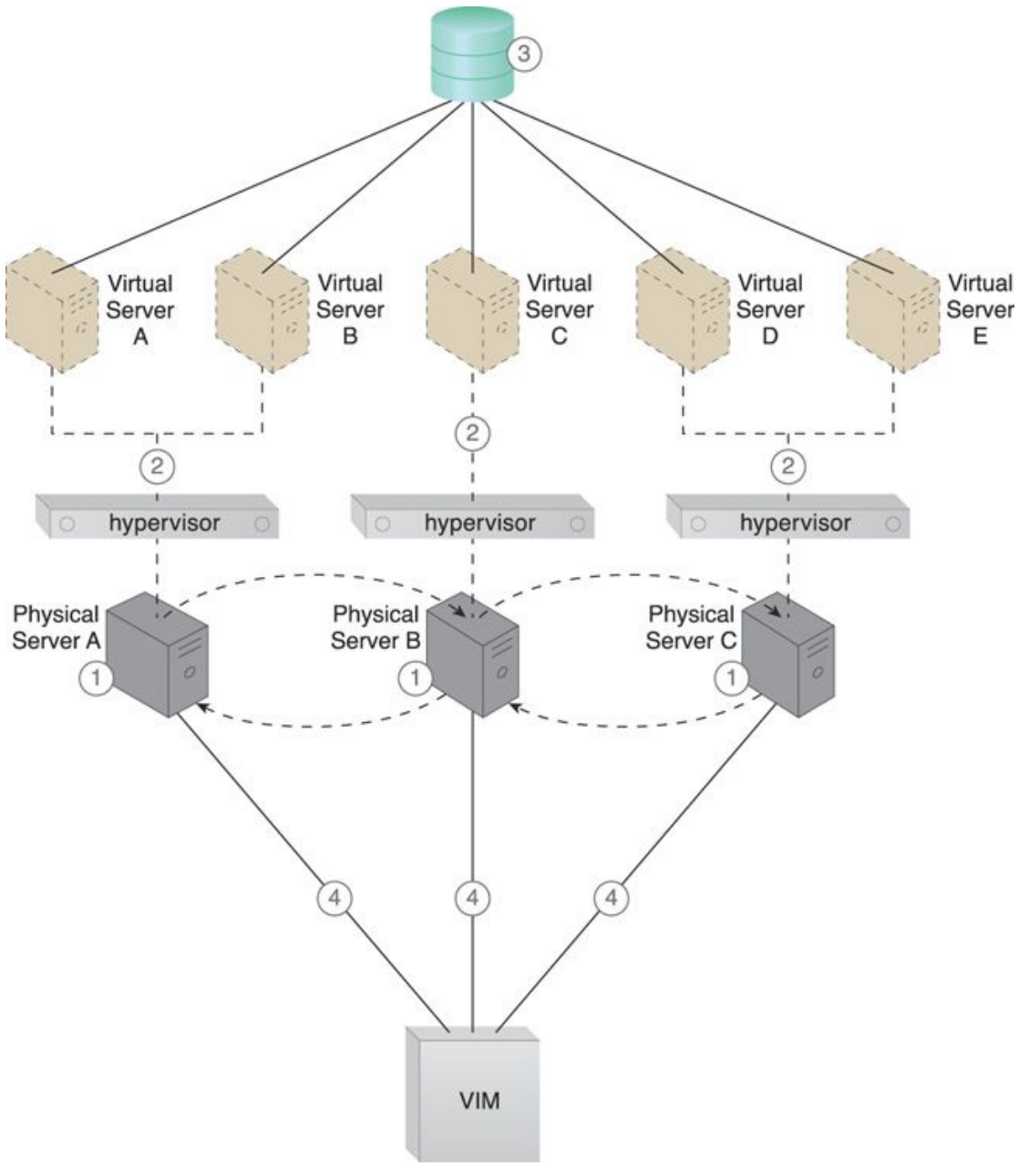
Live VM migration

- Live VM migration is a system that is capable of relocating virtual servers or virtual server instances at runtime.
- The hypervisor cluster uses a shared cloud storage device to live-migrate virtual servers



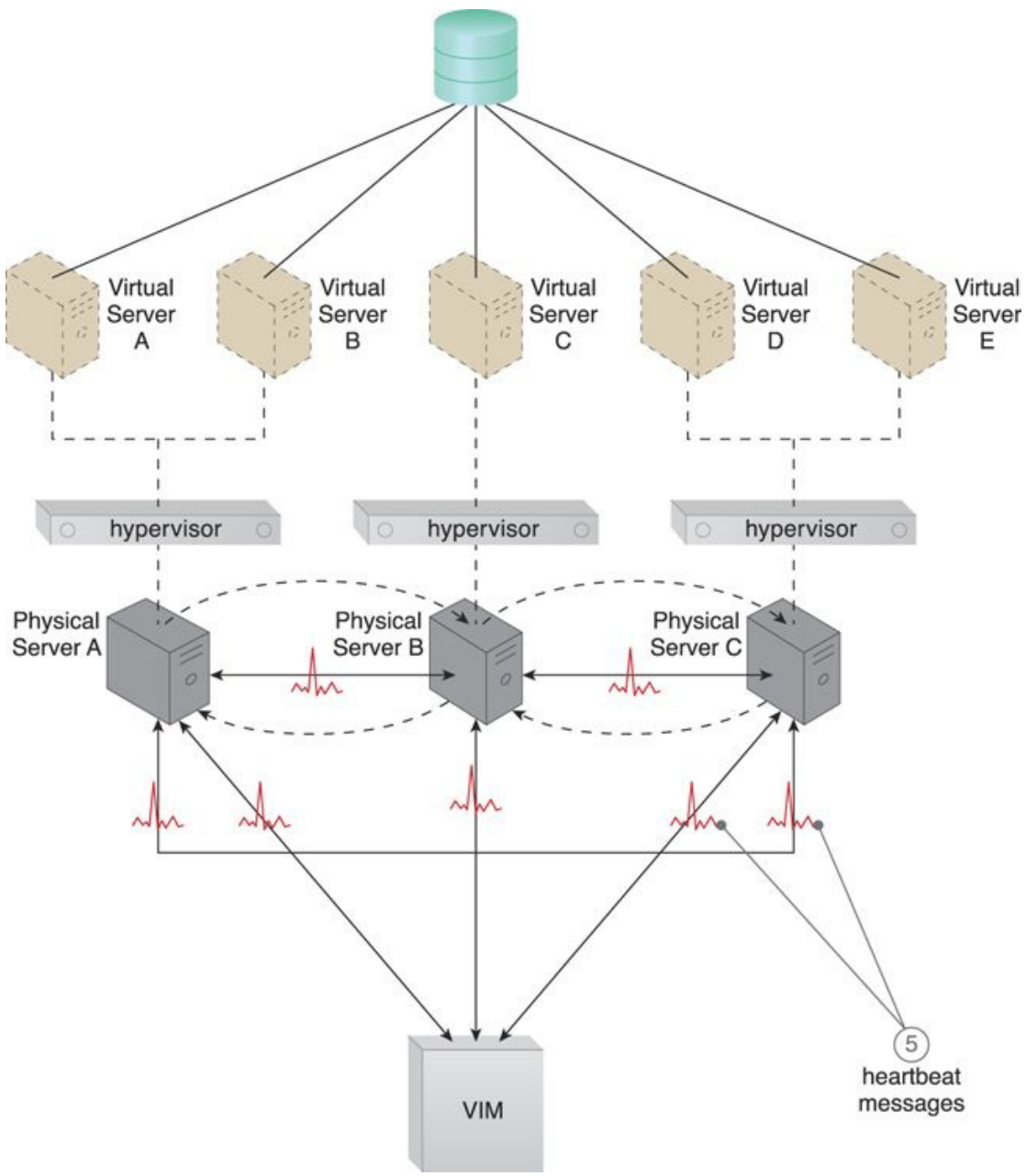
live VM migration

Live VM migration



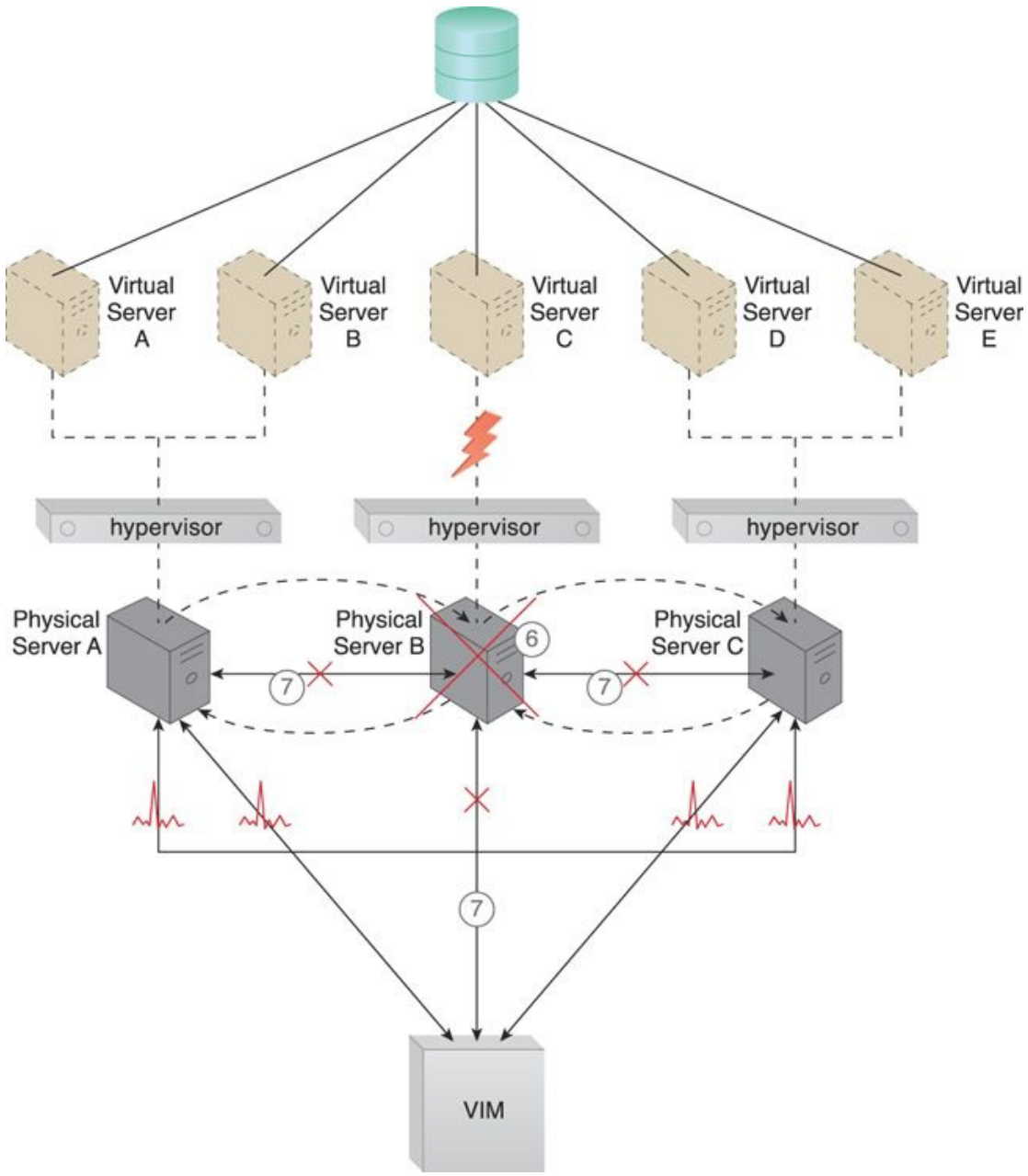
Hypervisors are installed on Physical Servers A, B, and C (1).
Virtual servers are created by the hypervisors (2).
A shared cloud storage device containing virtual server configuration files is positioned in a shared cloud storage device for access by all hypervisors (3).
The hypervisor cluster is enabled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

Live VM migration



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

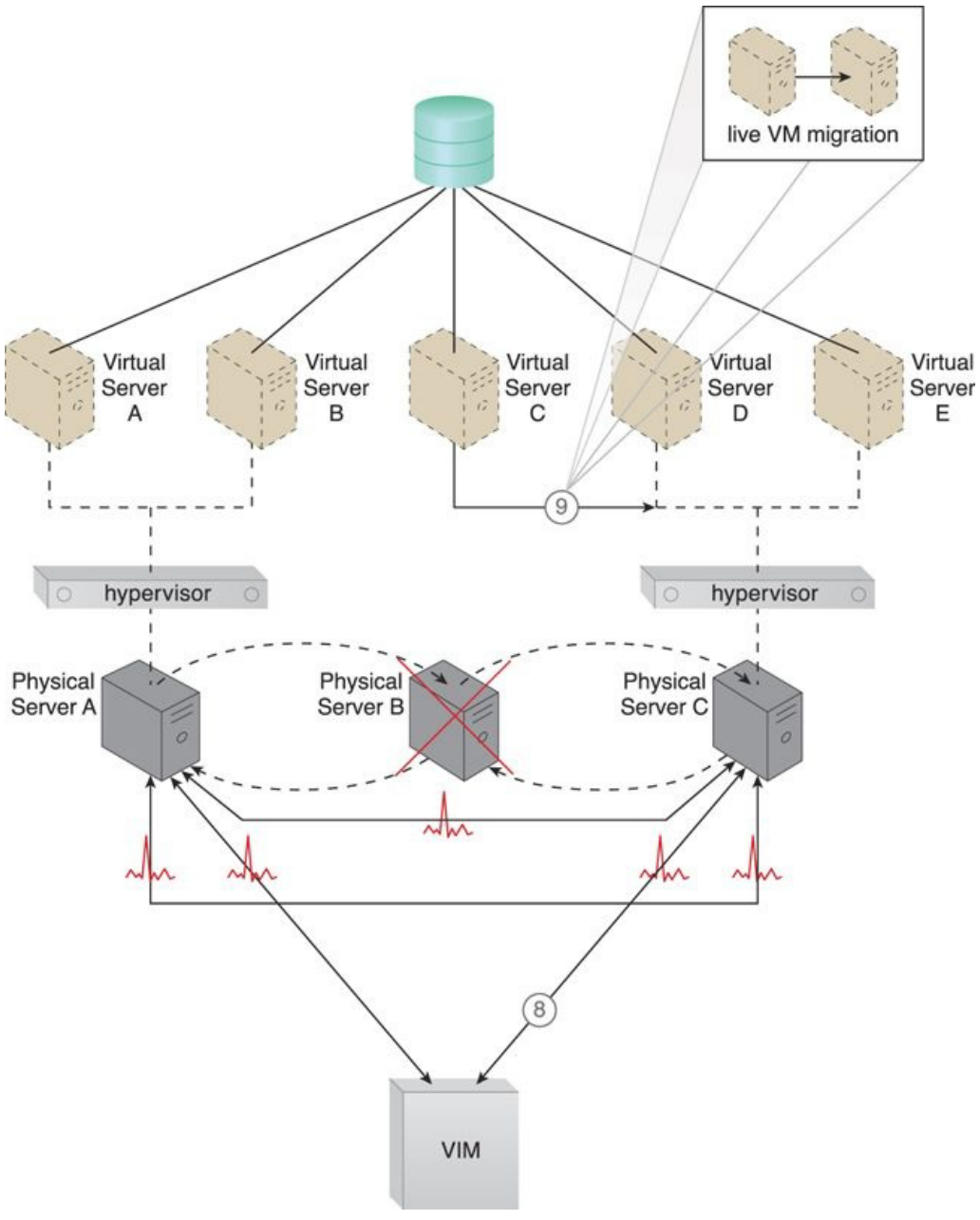
Live VM migration



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).

Live VM migration



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 running on Physical Server C, where restarting may be necessary before normal operations can be resumed (9).

Live VM migration

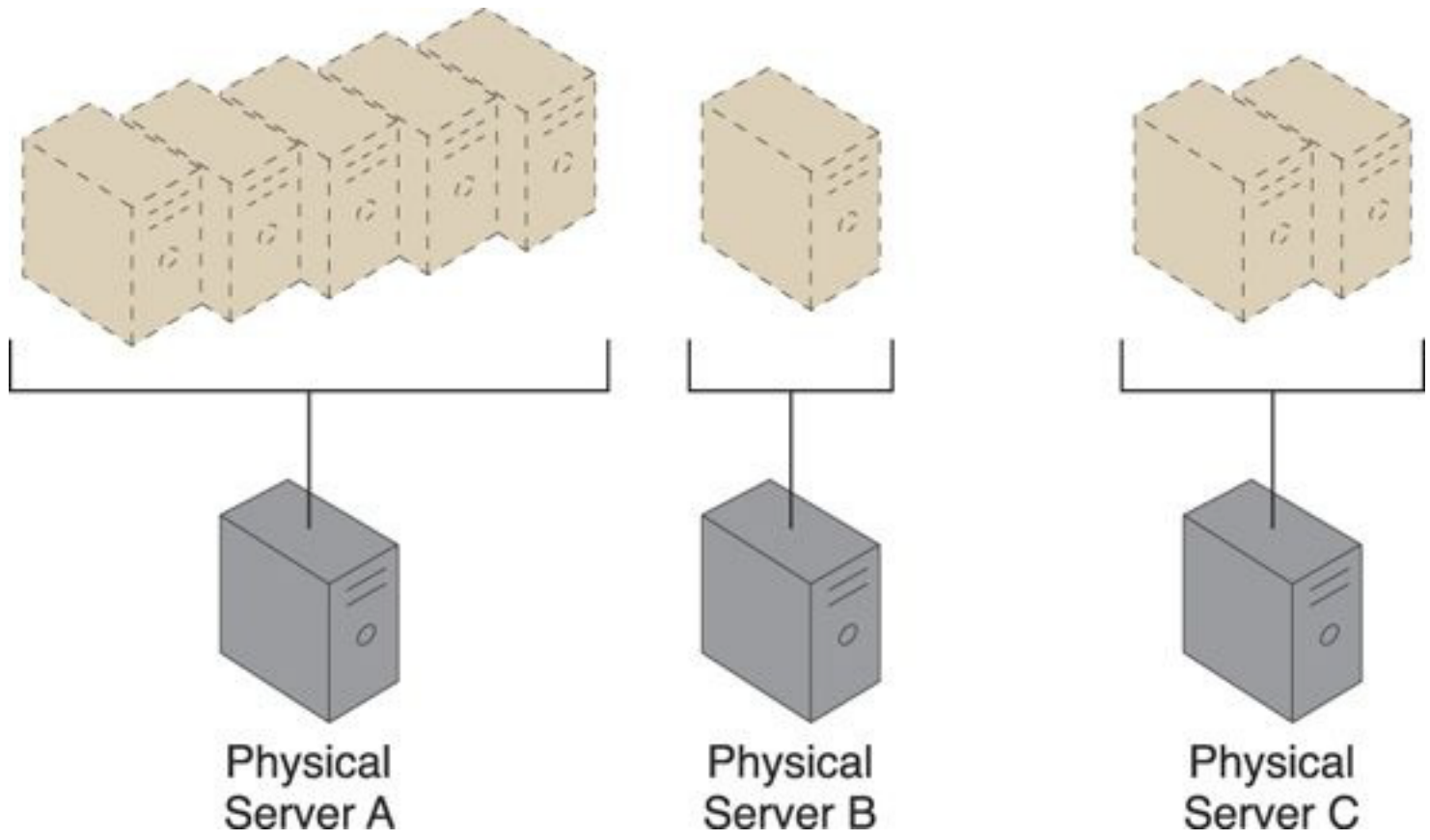
In addition to the hypervisor and resource cluster mechanisms that form the core of this architectural model and the virtual servers that are protected by the clustered environment, the following mechanisms can be incorporated:

- *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.
- *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 neighboring physical servers
- 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).

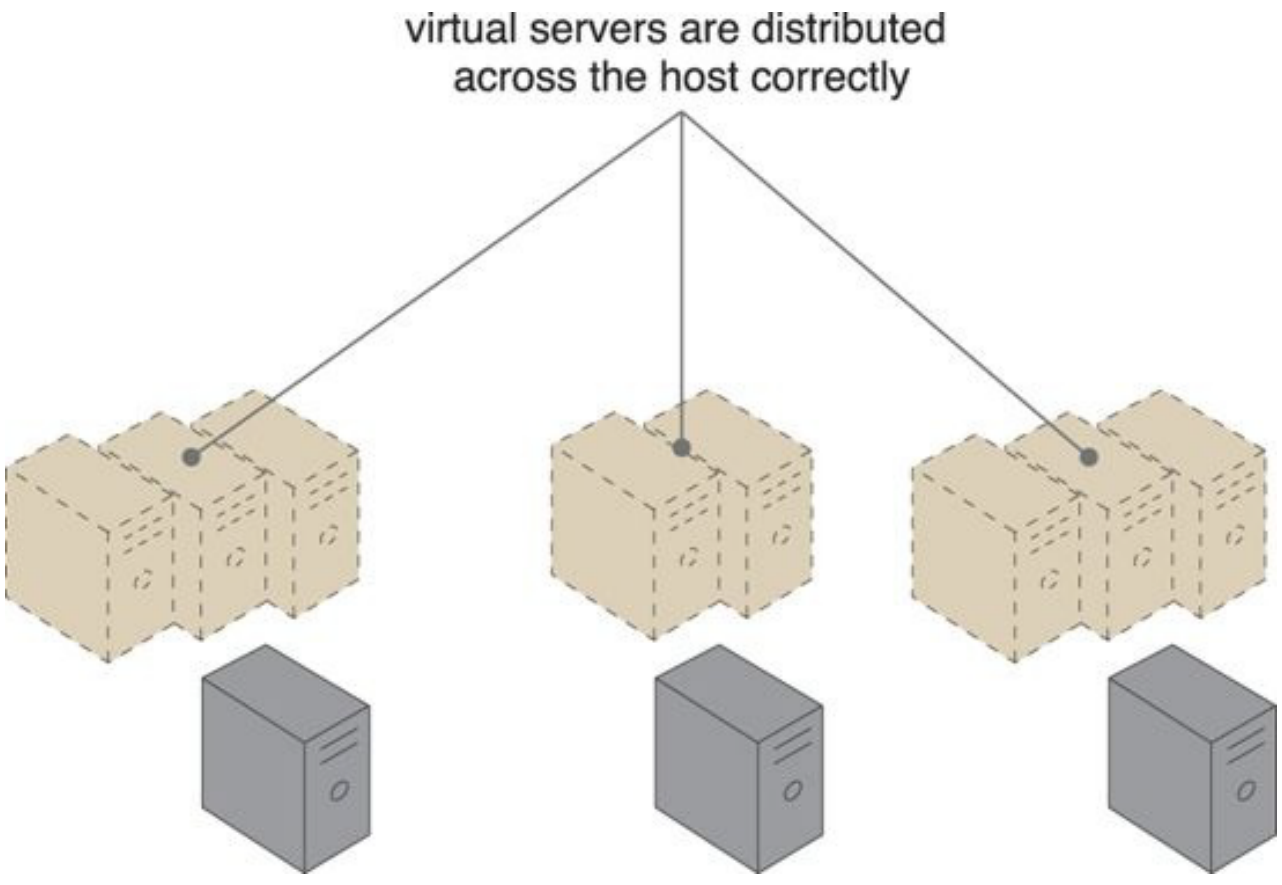
Load Balanced Virtual Server Instances Architecture



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

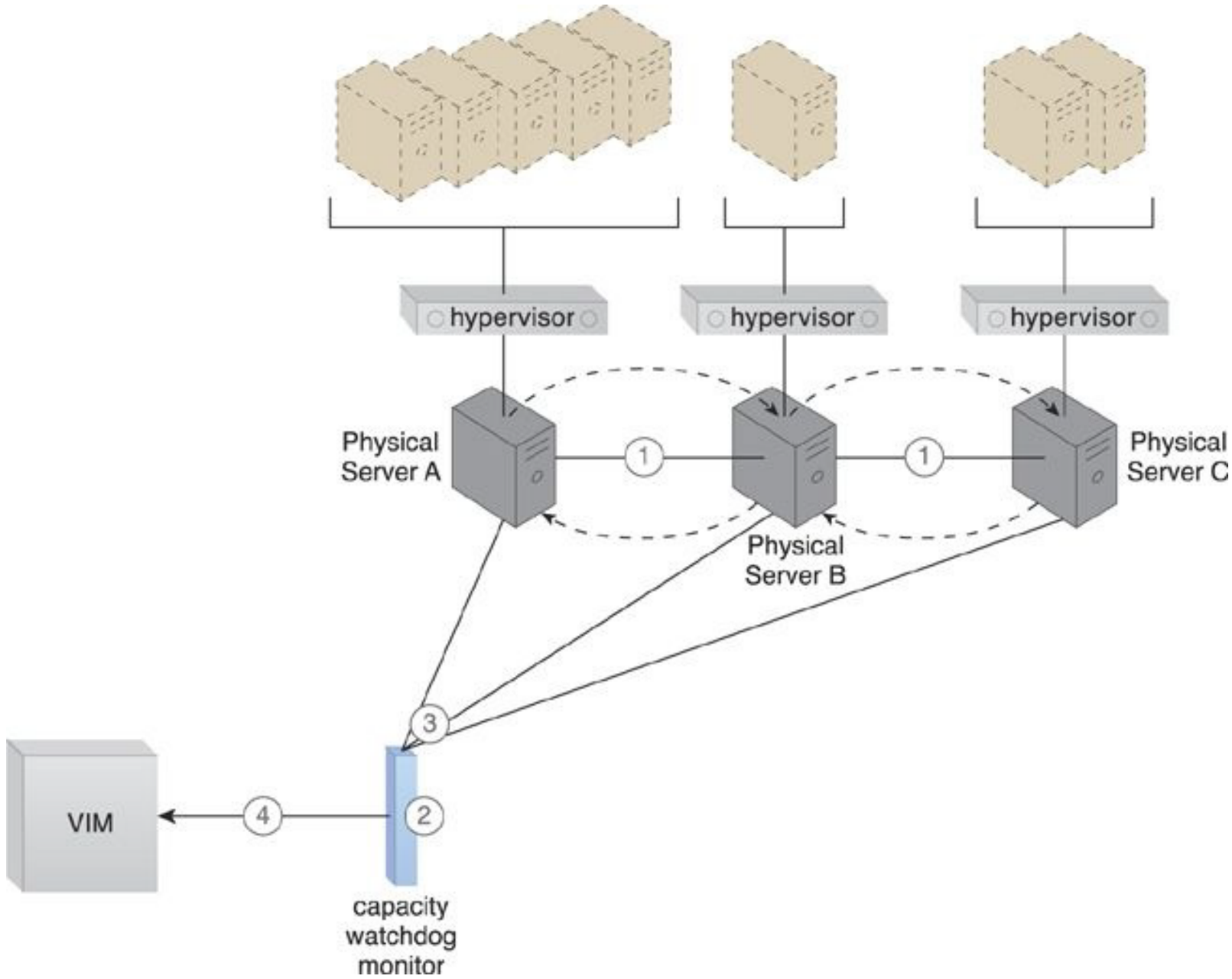
Load Balanced Virtual Server Instances Architecture

- 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

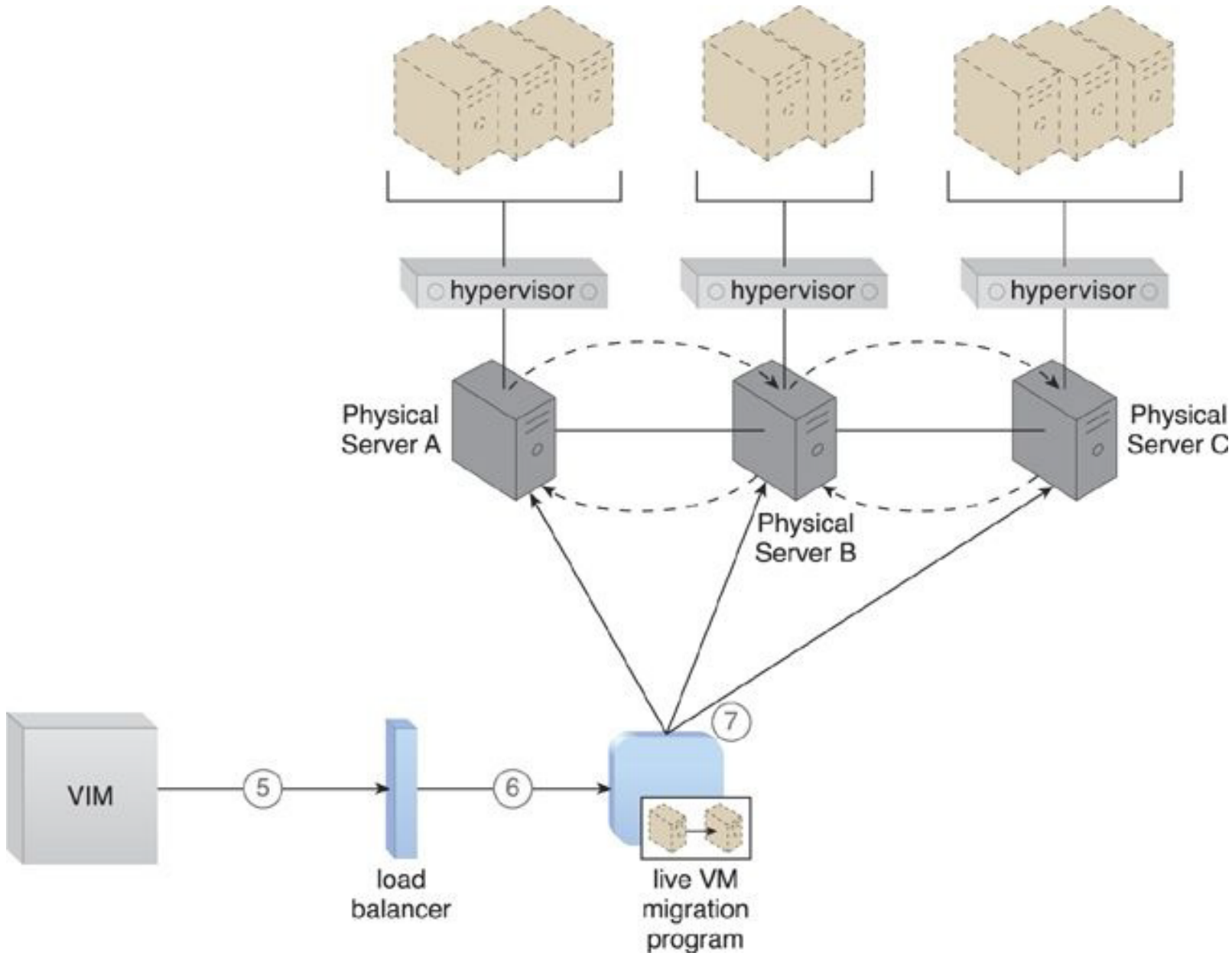


The virtual server instances are more evenly distributed across the physical server hosts.

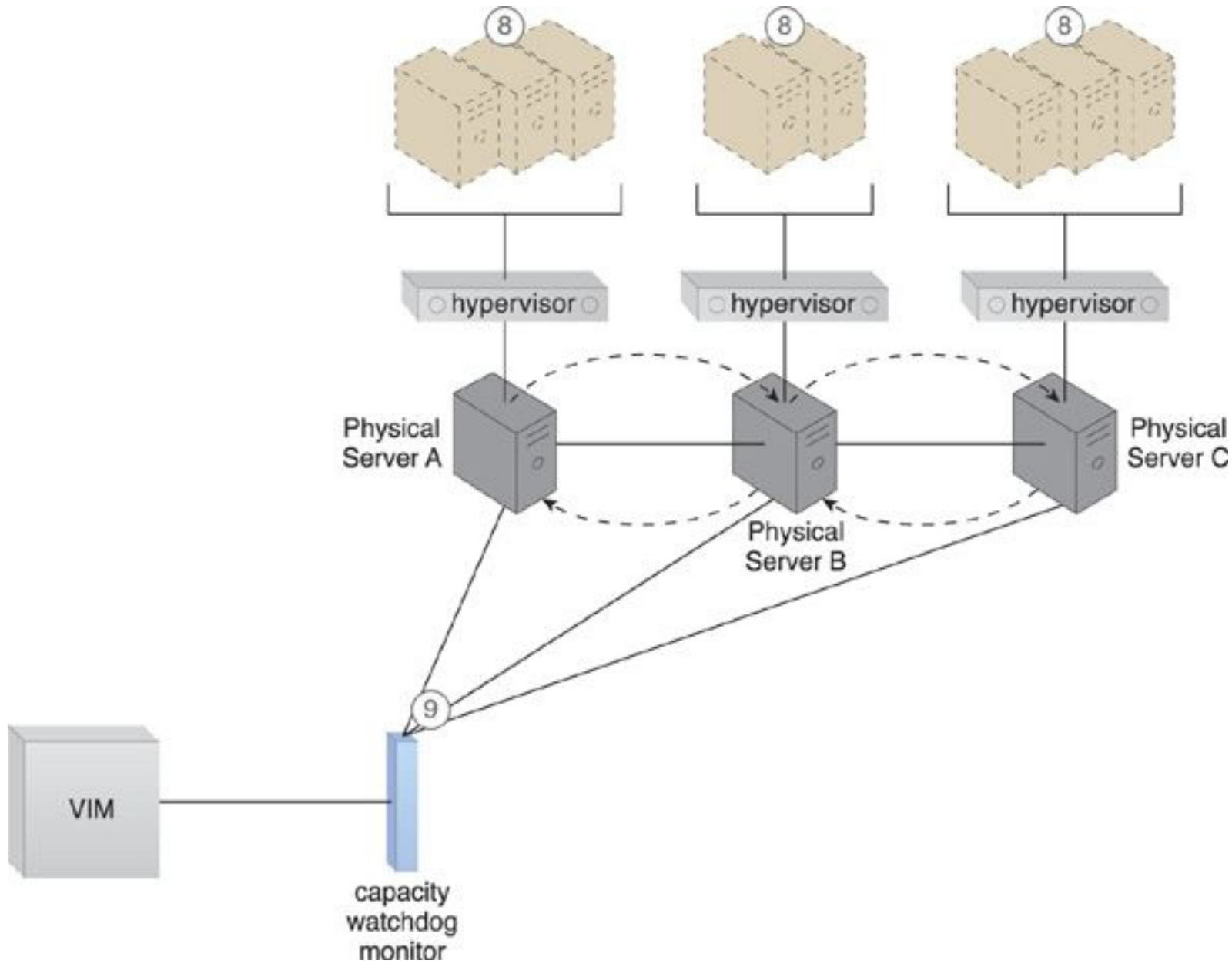
- The capacity watchdog system is comprised of a capacity watchdog cloud usage monitor, the live VM migration program, and a capacity planner. The capacity watchdog monitor tracks physical and virtual server usage and reports any significant fluctuations to the capacity planner, which is responsible 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 signaled to move the virtual server



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).



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).



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

Load Balanced Virtual Server Instances Architecture

- 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.

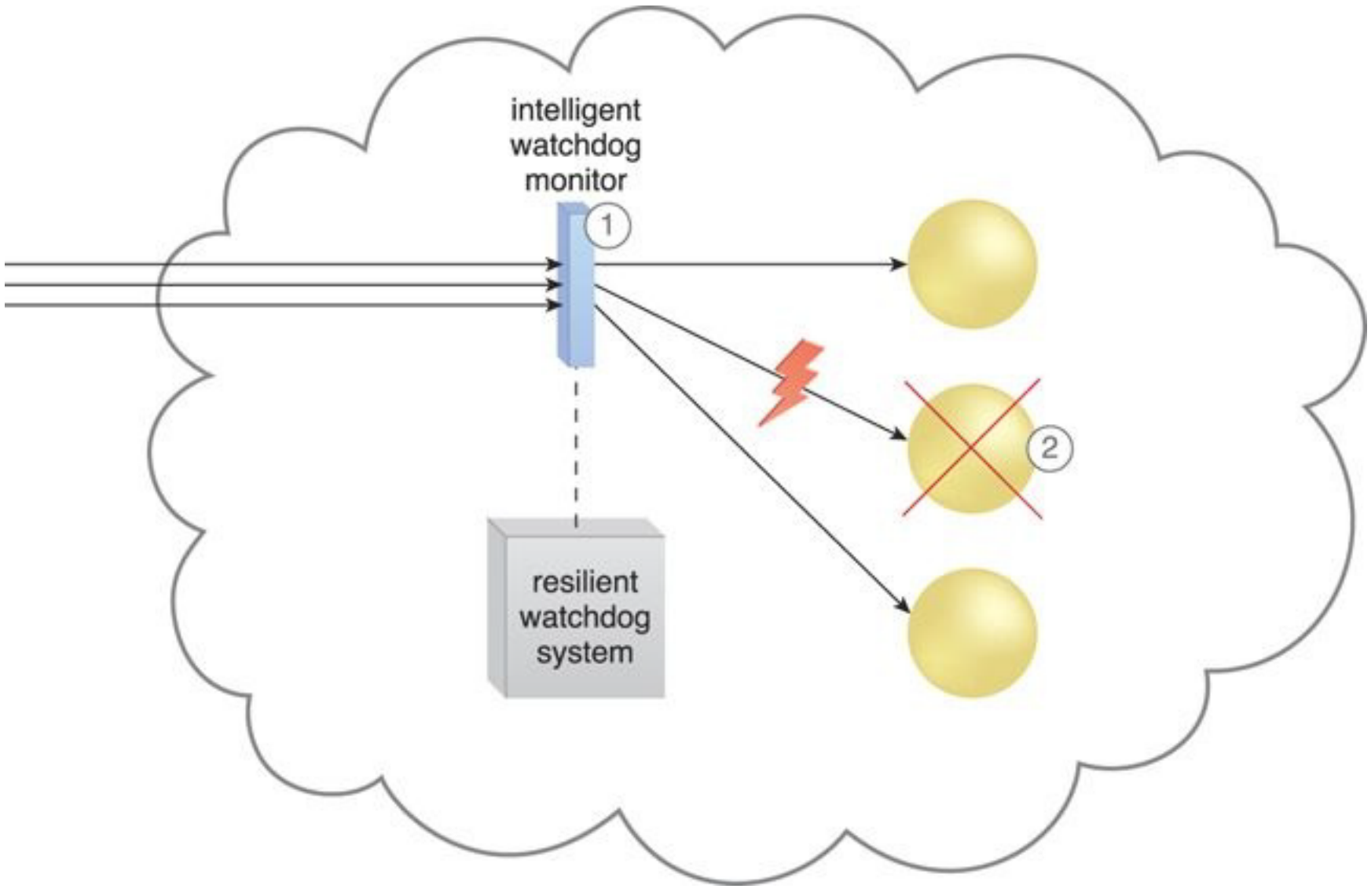
Dynamic Failure Detection and Recovery Architecture

- 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.

Dynamic Failure Detection and Recovery Architecture

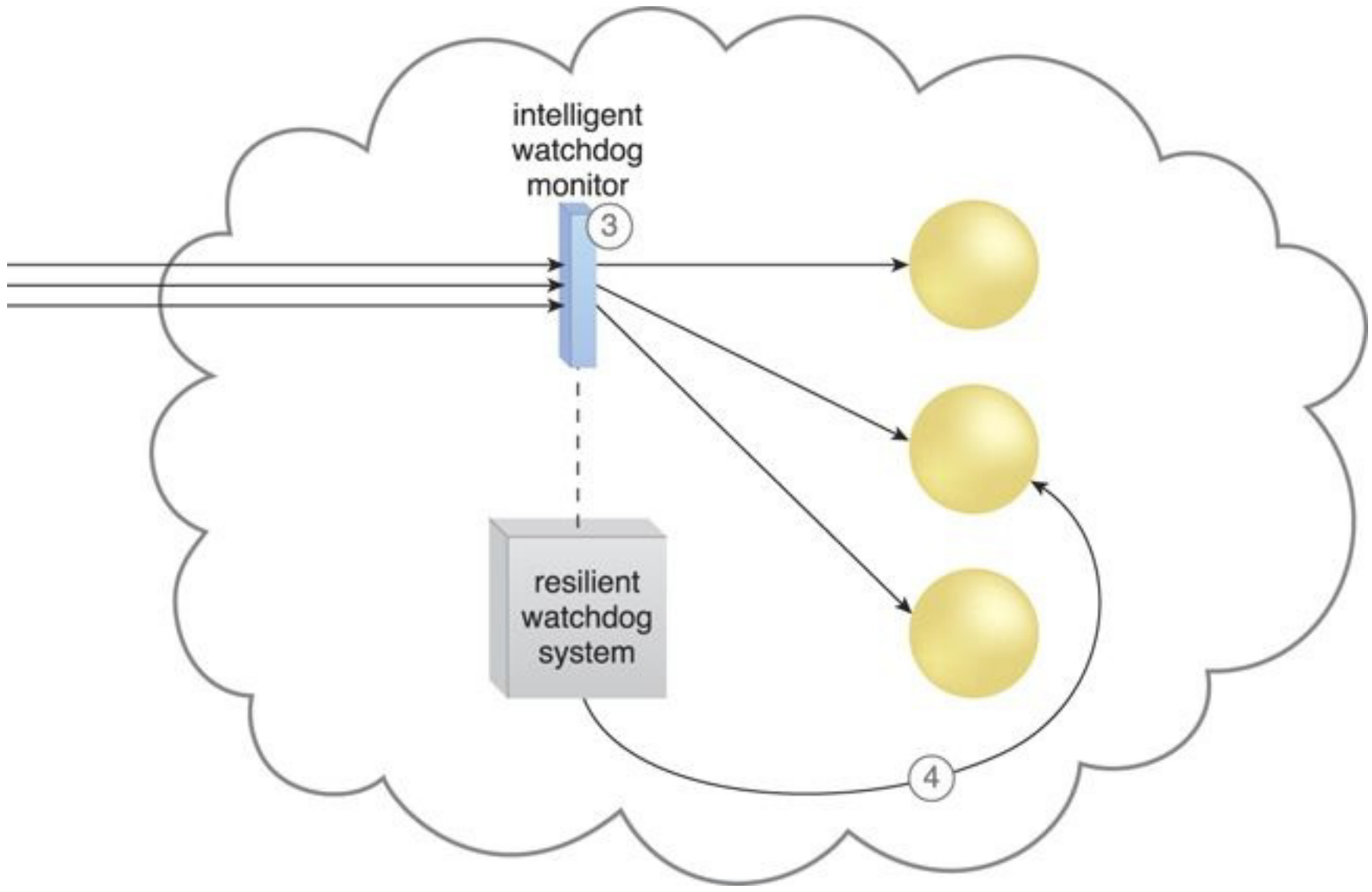
- The *dynamic failure detection and recovery architecture establishes a resilient* watchdog system to monitor and respond to a wide range of pre-defined failure scenarios.
- This system notifies and escalates the failure conditions that it cannot automatically resolve itself.
- It relies on a specialized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

Dynamic Failure Detection and Recovery Architecture



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

Dynamic Failure Detection and Recovery Architecture



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).

Dynamic Failure Detection and Recovery Architecture

- The resilient watchdog system performs the following five core functions:
- watching
- deciding upon an event
- acting upon an event
- reporting
- escalating

Dynamic Failure Detection and Recovery Architecture

- 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, one recovery policy can state that a recovery attempt needs to be carried out automatically before issuing a notification
- 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 trap
  - logging a ticket