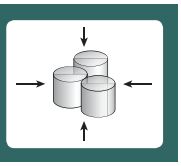
Shared Resources

How can the capacity of physical IT resources be used to their potential?



Problem	Allocating dedicated IT resources to individual consumers can be wasteful and under-utilize their collective capacity.
Solution	Physical IT resources are shared by partitioning them into lower capacity virtual IT resources that are provisioned to multiple cloud consumers.
Application	Virtualization technology is used to create virtual instances of physical IT resources. Each virtualized IT resource can be assigned to a cloud consumer, while the underlying physical IT resource is shared.
Mechanisms	Audit Monitor, Cloud Storage Device, Cloud Usage Monitor, Hypervisor, Logical Network Perimeter, Resource Replication, Virtual CPU, Virtual Infrastructure Manager (VIM), Virtual RAM, Virtual Server

Problem

Organizations commonly purchase physical on-premise IT resources, such as physical servers and storage devices, and allocate each to specific applications, users, or other types of consumers (Figure 3.1). The narrow scope of some IT resource usage results in the IT resource's overall capacity rarely being fully used. Over time, the processing potential of each IT resource is not reached. Consequently, the return on the investment of each IT resource is also not fully realized. The longer these types of dedicated IT resources are used, the more wasteful they become, and more opportunities to further leverage their potential are lost.

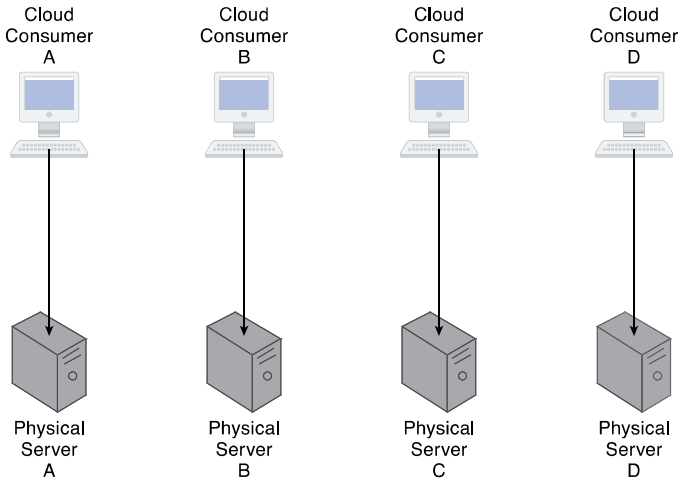


Figure 3.1
Each cloud consumer is allocated a dedicated physical server. It is likely that, over time, a significant amount of the physical servers' combined capacity will be under-utilized.

Solution

Virtual instances of physical IT resources are created and shared by multiple consumers, potentially to the extent to which the capacity of the physical IT resource can support (Figure 3.2). This maximizes the utilization of each physical IT resource, thereby also maximizing the return on its investment.

This pattern further forms the fundamental basis of a model by which virtual instances of the physical IT resource can be used (and leased) temporarily.

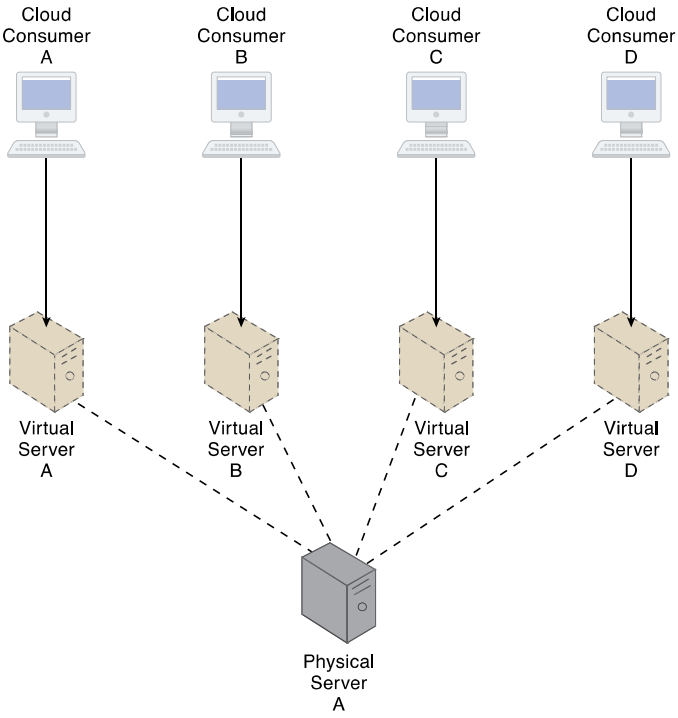


Figure 3.2
Each cloud consumer is allocated a virtual server instance of a single underlying physical server. In this case, the physical server is likely greater than if each cloud consumer were given its own physical server. However, the cost of one high-capacity physical server is lower than four medium-capacity physical servers and its processing potential will be utilized to a greater extent.

Application

The most common technology used to apply this pattern is virtualization. The specific components and mechanisms that are used depend on what type of IT resource needs to be shared. For example, the virtual server mechanism is used to share a physical server's processing capacity and the hypervisor mechanism is utilized to create instances of the virtual server. The VIM component can be further incorporated to manage hypervisors, virtual server instances, and their distribution.

It is important to note how the Shared Resources pattern is positioned among compound patterns, especially given its fundamental nature in relation to cloud platforms:

The Shared Resources pattern is:

- an optional member of the Private Cloud (474) compound pattern because, although common in private clouds, the virtualization of physical IT resources for cloud consumer sharing purposes is an option that can be chosen in support of the business requirements of the organization acting as cloud provider.
- a required member of the Public Cloud (476) compound pattern because of its inherent need to share IT resources to numerous cloud consumers.
- an optional member of the IaaS (482) compound pattern because the cloud provider may allow the cloud consumer access to administer raw physical IT resources and the decision of whether and how to use virtualization technology is left to the cloud consumer.
- a required member of the PaaS (480) compound pattern because the ready-made environment mechanism itself is naturally virtualized.
- a required member of the SaaS (478) compound pattern because SaaS offerings are naturally virtualized.
- a required member of the Multitenant Environment (486) compound pattern because this pattern provides a cloud technology architecture that specifically addresses the sharing of IT resources.

The sharing of IT resources introduces risks and challenges:

- One physical IT resource can become a single point of failure for multiple virtual IT resources and multiple corresponding cloud consumers.
- The virtualized physical IT resource may become over-utilized and therefore unable to fulfill all of the processing demands of its virtualized instances. This is referred to as a resource constraint and represents a condition that can lead to degradation of performance and various runtime exceptions.
- The virtualized instances of an underlying physical IT resource shared by multiple cloud consumers can introduce overlapping trust boundaries that can pose a security concern.

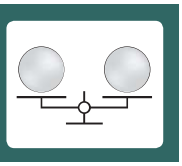
These and other problems raised by the application of this pattern are addressed by other patterns, such as Resource Pooling (99) and Resource Reservation (106).

Mechanisms

- *Audit Monitor* – When the Shared Resources pattern is applied, it can change how and where data is processed and stored. This may require the use of an audit monitor mechanism to ensure that the utilization of shared IT resources does not inadvertently violate legal requirements or regulations.
- *Cloud Storage Device* – This mechanism represents a common type of IT resource that is shared by the application of this pattern.
- *Cloud Usage Monitor* – Various cloud usage monitors may be involved with tracking the shared usage of IT resources.
- *Hypervisor* – A hypervisor can provide virtual servers with access to shared IT resources hosted by the hypervisor.
- *Logical Network Perimeter* – This mechanism provides network-level isolation that helps protect shared IT resources and their cloud consumers.
- *Resource Replication* – The resource replication mechanism may be used to generate new instances of IT resources made available for shared usage.
- *Virtual CPU* – This mechanism is used to share the hypervisor's physical CPU between virtual servers.
- *Virtual Infrastructure Manager (VIM)* – This mechanism is used to configure how physical resources are to be shared between virtual servers in order to send the configurations to the hypervisors.
- *Virtual RAM* – This mechanism is used to determine how a hypervisor's physical memory is to be shared between virtual servers.
- *Virtual Server* – Virtual servers may be shared or may host shared IT resources.

Workload Distribution

How can IT resource over-utilization be avoided?



Problem	IT resources subjected to high volumes of concurrent usage can suffer degraded performance, reduced availability and reliability, and can become susceptible to overall failure.
Solution	The IT resource is horizontally scaled and a load balancing system is used to distribute runtime workloads across multiple IT resources.
Application	Load balancing technology is incorporated into the cloud architecture and configured with appropriate load balancing algorithms to ensure effective workload distribution.
Mechanisms	Audit Monitor, Cloud Storage Device, Cloud Usage Monitor, Hypervisor, Load Balancer, Logical Network Perimeter, Resource Cluster, Resource Replication, Virtual Server

Problem

IT resources that are shared or are made available to consumers with unpredictable usage requirements can become over-utilized when usage demands near or exceed their capacities (Figure 3.3). This can result in runtime exceptions and failure conditions that cause the affected IT resources to reject consumer requests or shut down altogether.

Solution

The IT resource is horizontally scaled via the addition of one or more identical IT resources and a load balancing system further extends the cloud architecture to provide runtime logic capable of evenly distributing the workload across all available IT resources (Figure 3.4). This minimizes the chances that any one of the IT resources will be over-utilized (or under-utilized).

Application

This pattern is primarily applied via the use of the load balancer mechanism, of which variations with different types of load balancing algorithms exist. The automated scaling listener mechanism can also be used in a similar capacity to respond when an IT resource's thresholds are reached.

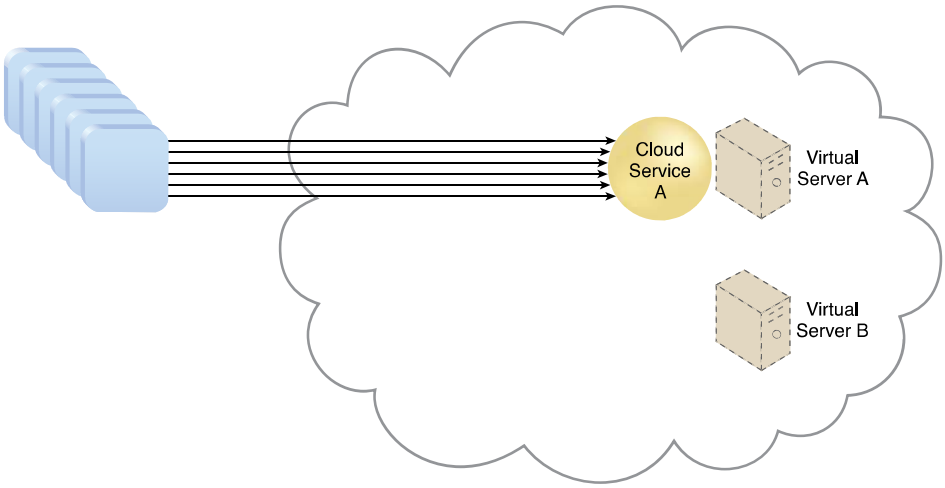


Figure 3.3
A group of cloud service consumers simultaneously access Cloud Service A, which is hosted by Virtual Server A. Another virtual server is available but is not being utilized. As a result, Virtual Server A is over-utilized.

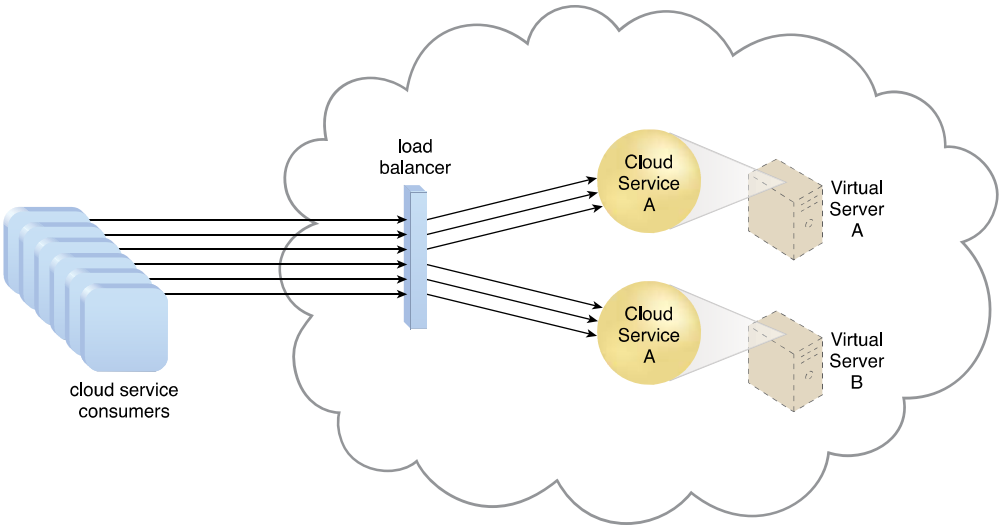


Figure 3.4
A redundant copy of Cloud Service A is implemented on Virtual Server B. The load balancer intercepts the cloud service consumer requests and directs them to both Virtual Server A and B to ensure even distribution of the workload.

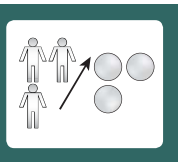
In addition to the distribution of conventional cloud service access and data exchanges, this pattern can also be applied to the load balancing of cloud storage devices and connectivity devices.

Mechanisms

- *Audit Monitor* – When distributing runtime workloads, the types of IT resources processing data and the geographical location of the IT resources (and the data) may need to be monitored for legal and regulatory requirements.
- *Cloud Storage Device* – This is one type of mechanism that may be used to distribute workload as a result of the application of this pattern.
- *Cloud Usage Monitor* – Various monitors may be involved with the runtime tracking of workload and data processing as part of a cloud architecture resulting from the application of this pattern.
- *Hypervisor* – Workloads between hypervisors and virtual servers hosted by hypervisors may need to be distributed.
- *Load Balancer* – This is a fundamental mechanism used to establish the base workload balancing logic in order to carry out the distribution of the workload.
- *Logical Network Perimeter* – The logical network perimeter isolates cloud consumer network boundaries in relation to how and to where workloads may be distributed.
- *Resource Cluster* – Clustered IT resources in active/active mode are commonly used to support the workload balancing between the different cluster nodes.
- *Resource Replication* – This mechanism may generate new instances of virtualized IT resources in response to runtime workload distribution demands.
- *Virtual Server* – Virtual servers may be the target of workload distribution or may themselves be hosting IT resources that are part of workload distribution architectures.

Dynamic Scalability

How can IT resources be scaled automatically in response to fluctuating demand?



Problem	It is challenging to equip an IT resource to match its processing requirements. If the demand for the IT resource is below its capacity, then it is under-utilized and if the demand is above its capacity it is over-utilized or unable to meet the demand.
Solution	The IT resource can be integrated with a reactive cloud architecture capable of automatically scaling it horizontally or vertically in response to fluctuating demand.
Application	Dynamic horizontal scaling can be enabled via the use of pools of identical IT resources and components capable of dispersing and retracting workloads across each pool. Dynamic vertical scaling can be enabled via technology capable of swapping IT resource components at runtime.
Mechanisms	Automated Scaling Listener, Cloud Storage Device, Cloud Usage Monitor, Hypervisor, Pay-Per-Use Monitor, Resource Replication, Virtual Server

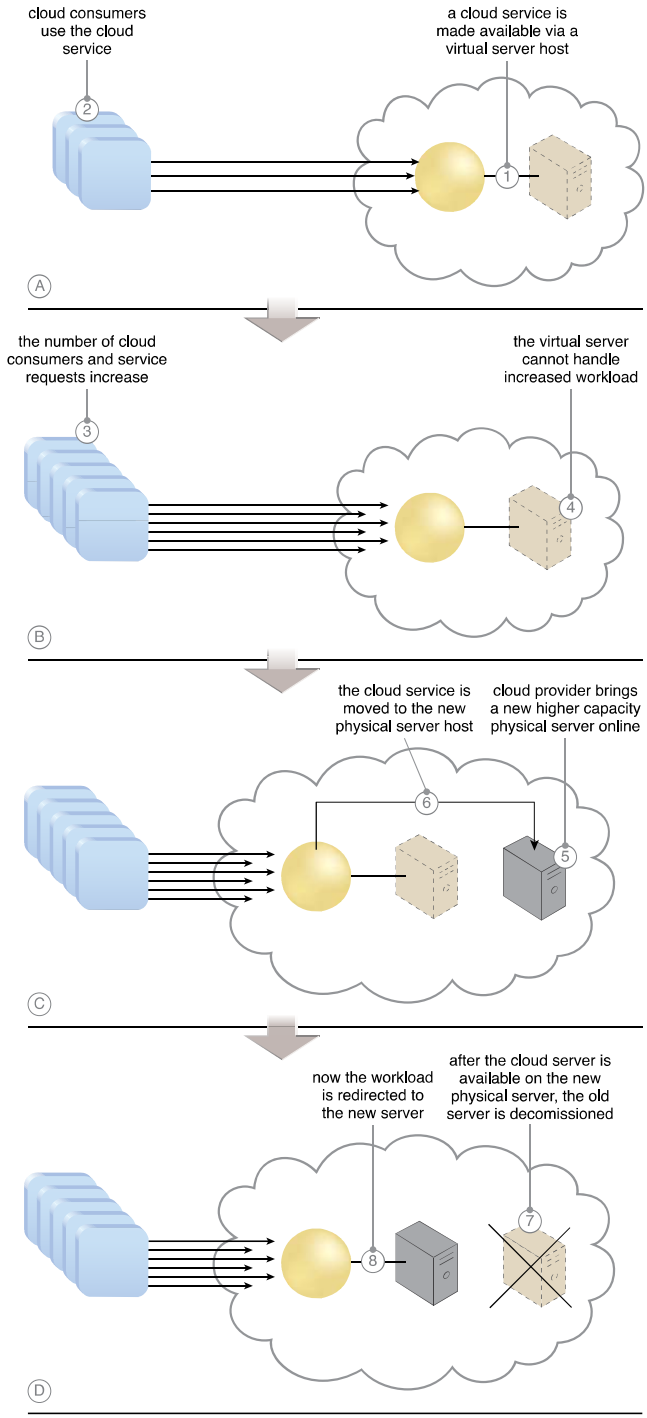
Problem

Manually preparing or extending IT resources in response to workload fluctuations is time-intensive and unacceptably inefficient. Determining when to add new IT resources to satisfy anticipated workload peaks is often speculative and generally risky. These additional IT resources can either remain under-utilized (and a failed financial investment), or fail to alleviate runtime performance and reliability problems when demand exceeds even the addition of their capacity.

The following steps are shown in Figures 3.5 and 3.6:

1. The cloud provider offers cloud services to cloud consumers.
2. Cloud consumers can scale the cloud services, as needed.
3. Over time, the number of cloud consumers increases.
4. The cloud provider's virtual server is overwhelmed with the increased workload capacity.

Figure 3.5
A non-dynamic cloud architecture in which vertical scaling is carried out in response to usage fluctuations (Part I).



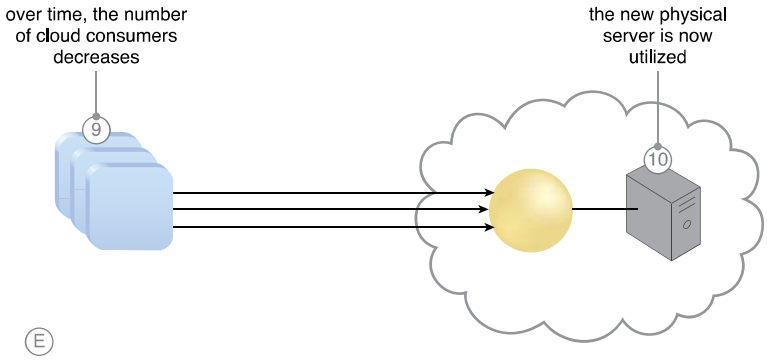


Figure 3.6
A non-dynamic cloud architecture in which vertical scaling is carried out in response to usage fluctuations (Part II).

5. The cloud provider brings a new, higher-capacity server online to handle an increased workload.
6. Because the required IT resources are not organized for sharing and are unprepared for allocation, the virtual server must have the operating system, required applications, and cloud services installed after being created.
7. Once the new server is ready, the old server is taken offline.
8. Now service requests are redirected to the new server.
9. After the peak usage period has ended, the number of cloud consumers and service requests naturally decrease.
10. Without properly implementing a process of under-utilized IT resource recovery, the new server's sizable workload capacity will not be fully utilized.

Solution

A system of predefined scaling conditions that trigger the dynamic allocation of IT resources can be introduced (Figure 3.7). The IT resources are allocated from resource pools to allow for variable utilization as dictated by demand fluctuations. Unneeded IT resources are efficiently reclaimed without requiring manual interaction.

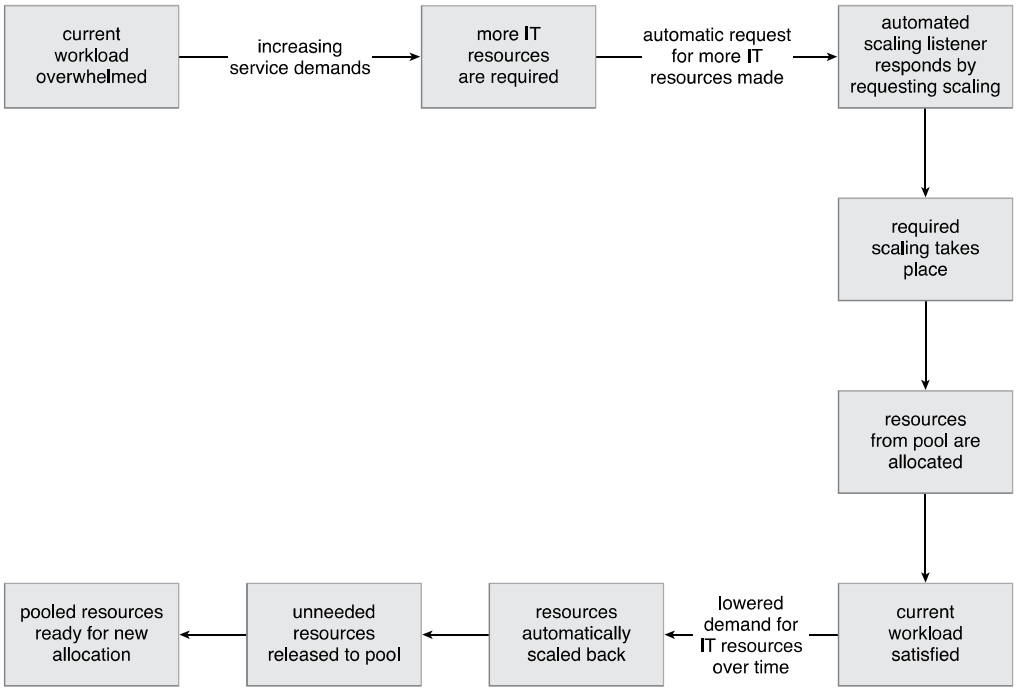


Figure 3.7
A sample dynamic scaling process.

Application

The fundamental Dynamic Scalability pattern primarily relies on the application of Resource Pooling (99) and the implementation of the automated scaling listener.

The automated scaling listener is configured with workload thresholds that determine when new IT resources need to be included in the workload processing. The automated scaling listener can further be provided with logic that allows it to verify the extent of additional IT resources a given cloud consumer is entitled to, based on its leasing arrangement with the cloud provider.

The following types of dynamic scaling are common:

- *Dynamic Horizontal Scaling* – In this type of dynamic scaling the number of IT resource instances is scaled to handle fluctuating workloads. The automatic scaling listener monitors requests and, if scaling is required, signals a resource

replication mechanism to initiate the duplication of the IT resources, as per requirements and permissions. (Figures 3.8 to 3.10 demonstrate this type of scaling.)

- *Dynamic Vertical Scaling* – This type of scaling occurs when there is a need to increase the processing capacity of a single IT resource. For instance, if a virtual server is being overloaded, it can dynamically have its memory increased or it may have a processing core added.
- *Dynamic Relocation* – The IT resource is relocated to a higher capacity host. For example, there may be a need to move a cloud service database from a tape-based SAN storage device with 4 Gbps I/O capacity to another disk-based SAN storage device with 8 Gbps I/O capacity.

Figures 3.8 to 3.10 demonstrate dynamic horizontal scaling in the following steps:

1. Cloud service consumers are sending requests to a cloud service.
2. The automated scaling listener monitors the cloud service to determine if pre-defined capacity thresholds are being exceeded.
3. The number of service requests coming from cloud service consumers further increases.
4. The workload exceeds the performance thresholds of the automated scaling listener. It determines the next course of action based on a pre-defined scaling policy.
5. If the cloud service implementation is deemed eligible for additional scaling, the automated scaling listener initiates the scaling process.
6. The automated scaling listener sends a signal to the resource replication mechanism.
7. The resource replication mechanism then creates more instances of the cloud service.
8. Now that the increased workload is accommodated, the automated scaling listener resumes monitoring and the detracting or adding of necessary IT resources.

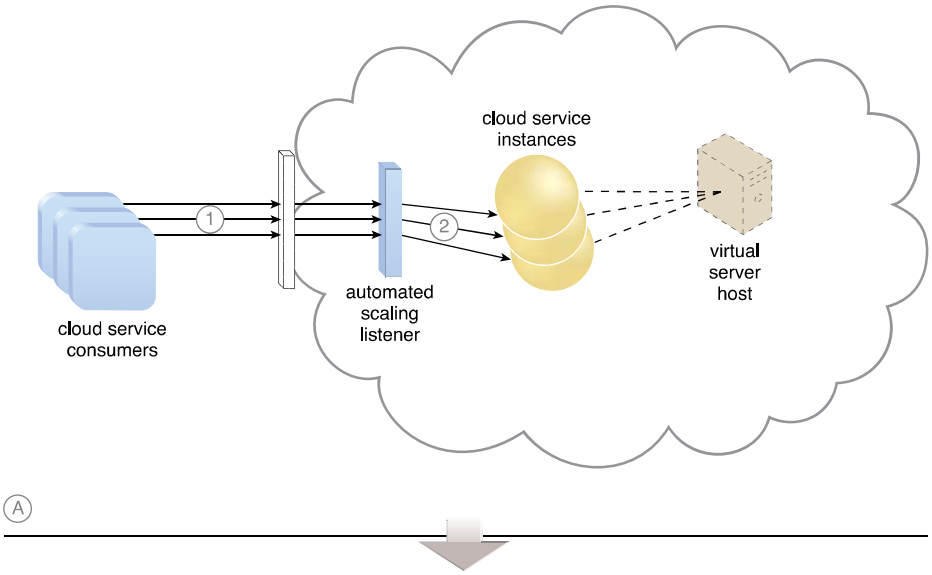


Figure 3.8
An example of a dynamic scaling architecture involving an automated scaling mechanism (Part I).

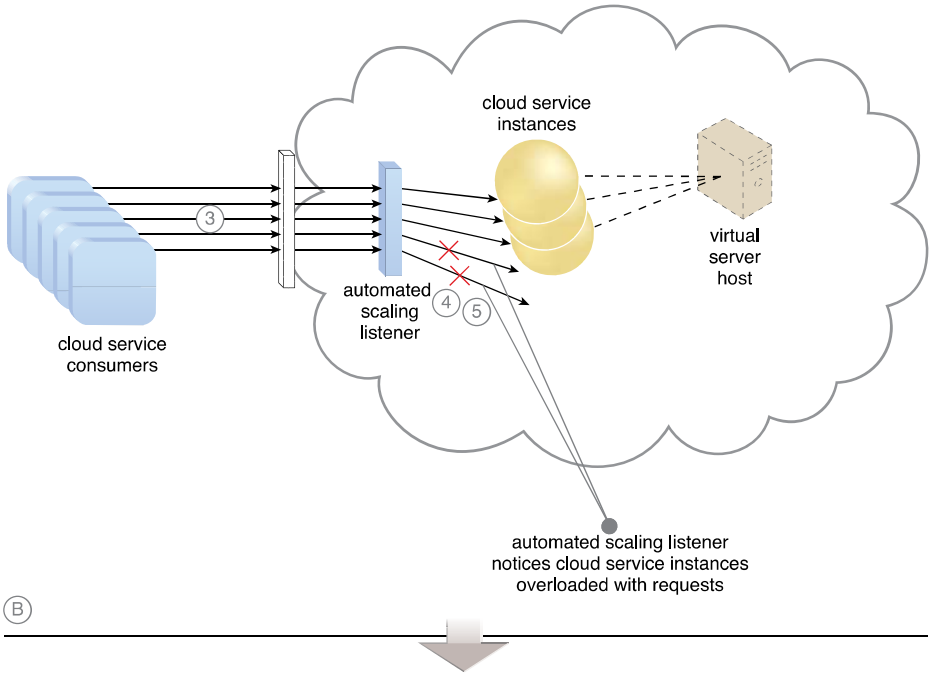


Figure 3.9
An example of a dynamic scaling architecture involving an automated scaling mechanism (Part II).

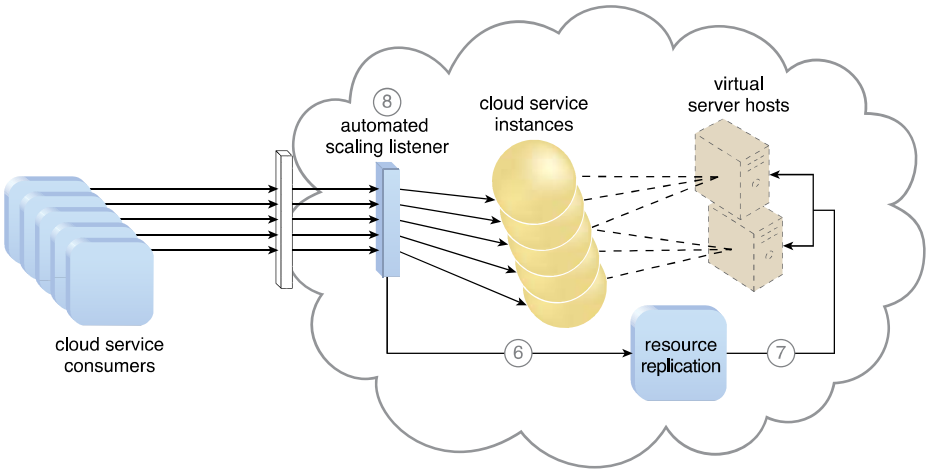


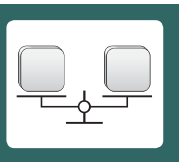
Figure 3.10
An example of a dynamic scaling architecture involving an automated scaling mechanism (Part III).

Mechanisms

- *Automated Scaling Listener* – The automated scaling listener is directly associated with the Dynamic Scalability pattern in that it monitors and compares workloads with predefined thresholds to initiate scaling in response to usage fluctuations.
- *Cloud Storage Device* – This mechanism and the data it stores may be scaled by the system established by this pattern.
- *Cloud Usage Monitor* – As per the automated scaling listener, cloud usage monitors are used to track runtime usage to initiate scaling in response to fluctuations.
- *Hypervisor* – The hypervisor may be invoked by a dynamic scalability system to create or remove virtual server instances. Alternatively, the hypervisor itself may be scaled.
- *Pay-Per-Use Monitor* – The pay-per-use monitor collects usage cost information in tandem with how IT resources are scaled.
- *Resource Replication* – This mechanism supports dynamic horizontal scaling by replicating IT resources, as required.
- *Virtual Server* – The virtual server may be scaled by the system established by this pattern.

Service Load Balancing

How can a cloud service accommodate increasing workloads?



Problem	A single cloud service implementation has a finite capacity, which leads to runtime exceptions, failure, and performance degradation when its processing thresholds are exceeded.
Solution	Redundant deployments of the cloud service are created and a load balancing system is added to dynamically distribute workloads across cloud service implementations.
Application	The duplicate cloud service implementations are organized into a resource pool. The load balancer is positioned as an external component or may be built-in, allowing hosting servers to balance workloads among themselves.
Mechanisms	Cloud Usage Monitor, Load Balancer, Resource Cluster, Resource Replication

Problem

Regardless of the processing capacity of its immediate hosting environment, a cloud service architecture may inherently be limited in its ability to accommodate high volumes of concurrent cloud service consumer requests. The cloud service's processing restrictions may be such that it is unable to leverage underlying cloud-based IT resources that normally support dynamic scalability. For example, the processing restrictions may originate from its architectural design, the complexity of its application logic, or inhibitive programming algorithms it is required to carry out at runtime. Such a cloud service may be forced to reject cloud service consumer requests when its processing capacity thresholds are reached (Figure 3.11).

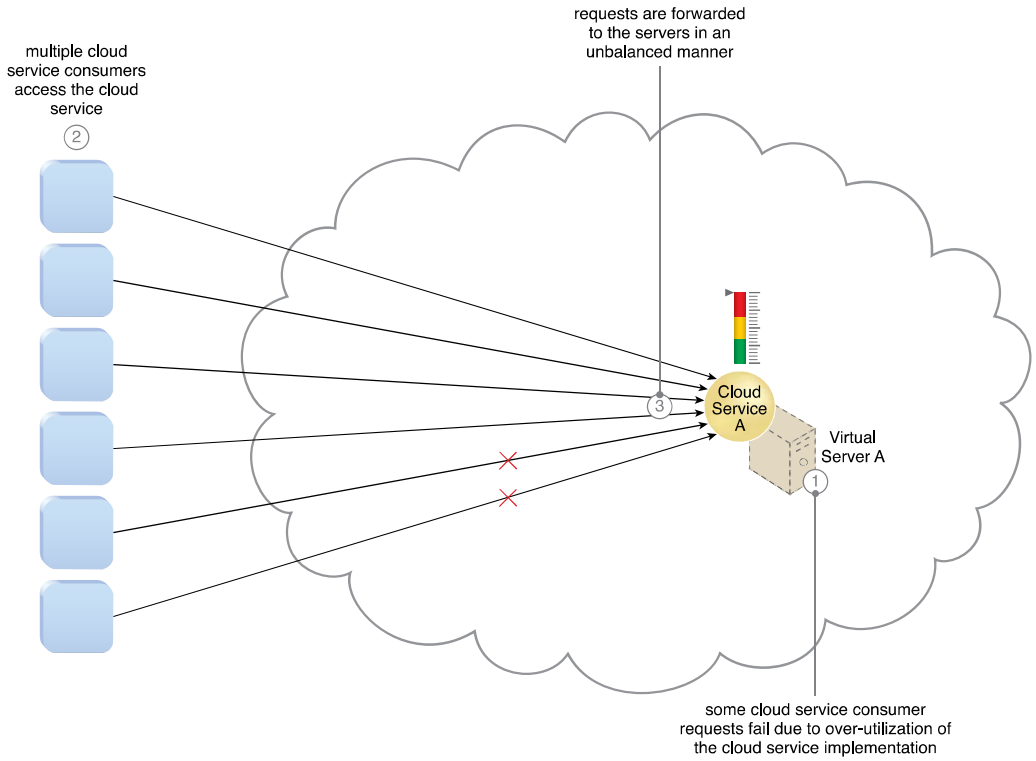


Figure 3.11
A single cloud service implementation reaches its runtime processing capacity and consequently rejects subsequent cloud service consumer requests.

Solution

Redundant implementations of the cloud service are created, each located on a different hosting server. A load balancer is utilized to intercept cloud service consumer requests in order to evenly distribute them across the multiple cloud service implementations (Figure 3.12).

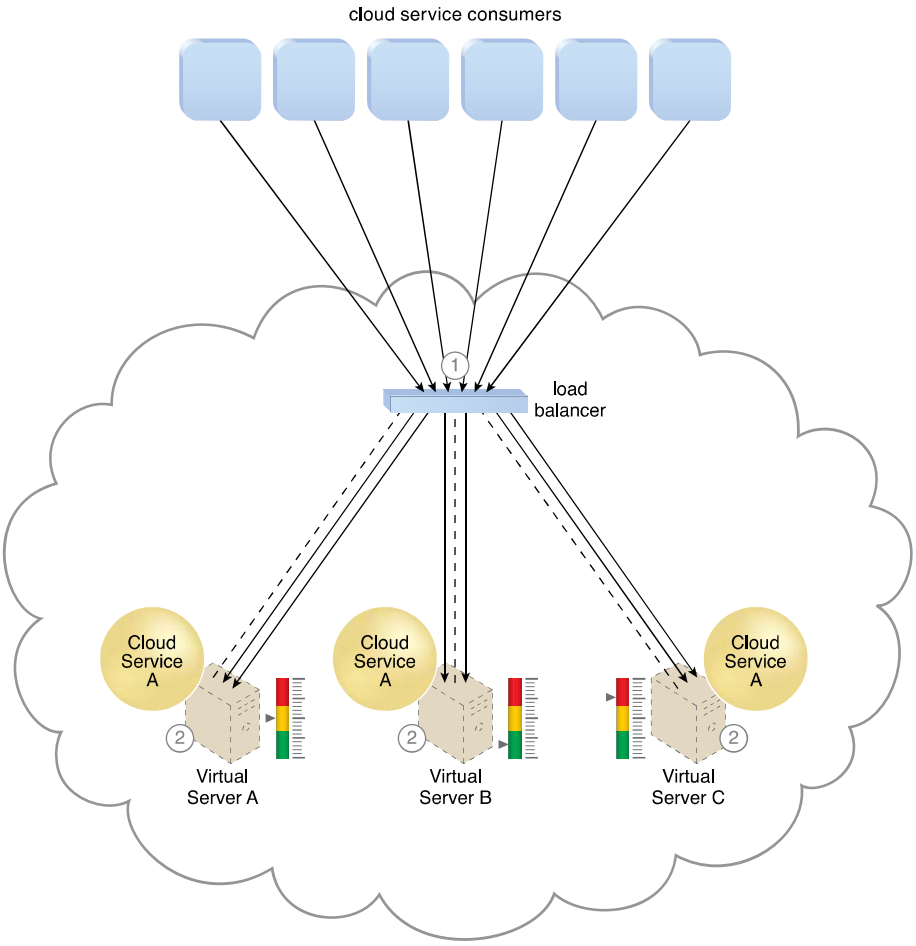


Figure 3.12
The load balancing agent intercepts messages sent by cloud service consumers (1) and forwards the messages at runtime to the virtual servers so that the workload processing is horizontally scaled (2).

Application

Depending on the anticipated workload and the processing capacity of hosting server environments, multiple instances of each cloud service implementation may be generated in order to establish pools of cloud services that can more efficiently respond to high volumes of concurrent requests.

The load balancer may be positioned independently from the cloud services and their hosting servers, as shown in Figure 3.12, or it may be 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, as shown in Figure 3.13.

For this pattern to be applied, a server group needs to be created and configured, so that server group members can be associated with the load balancer. The paths of cloud service consumer requests to be sent through the load balancer need to be set and the load balancer needs to be configured to evaluate each cloud service implementation's capacity on a regular basis.

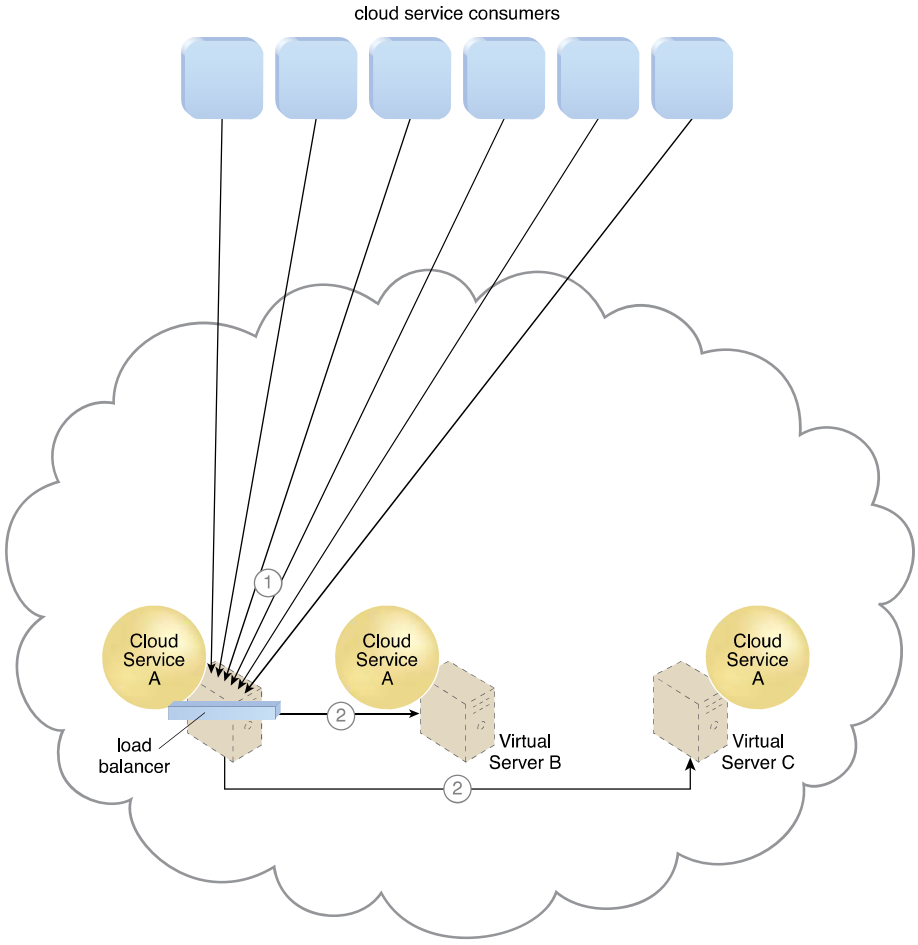


Figure 3.13

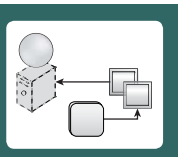
Cloud consumer requests are sent to Cloud Service A on Virtual Server A (1). The cloud service implementation includes built-in load balancing logic that is capable of distributing requests to the neighboring Cloud Service A implementations on Virtual Servers B and C (2).

Mechanisms

- *Cloud Usage Monitor* – In addition to performing various runtime monitoring and usage data collection tasks, cloud usage monitors may be involved with monitoring cloud service instances and their respective IT resource consumption levels.
- *Load Balancer* – This represents the fundamental mechanism used to apply this pattern in order to establish the necessary horizontal scaling functionality.
- *Resource Cluster* – Active-active cluster groups may be incorporated in a service load balancing architecture to help balance workloads across different members of the cluster.
- *Resource Replication* – Resource replication is utilized to keep cloud service implementations synchronized.

Elastic Resource Capacity

How can the processing capacity of virtual servers be dynamically scaled in response to fluctuating IT resource usage requirements?



Problem	When IT resources hosted by a virtual server impose processing requirements that exceed the virtual server's capacity, the performance and reliability of the hosted IT resources and the virtual server itself may be compromised.
Solution	An elastic provisioning system is established to dynamically allocate and reclaim CPUs and RAM for a virtual server in response to the fluctuating processing requirements of its hosted IT resources.
Application	Resource pools are utilized by scaling technology that interacts with the hypervisor and/or VIM to retrieve and return CPU and RAM resources at runtime, as per necessary processing capacity.
Mechanisms	Automated Scaling Listener, Cloud Usage Monitor, Hypervisor, Live VM Migration, Pay-Per-Use Monitor, Resource Replication, Virtual CPU, Virtual Infrastructure Manager (VIM), Virtual RAM, Virtual Server

Problem

When the processing capacity of a virtual server is reached at runtime, it becomes unavailable, resulting in scalability limitations and inhibiting the performance and reliability of its hosted IT resources (Figure 3.14).

Solution

Pools of CPUs and RAM are established for shared allocation. The runtime processing of a virtual server is monitored so that prior to capacity thresholds being met, additional processing power from the resource pool can be leveraged via dynamic allocation to the virtual server. This vertically scales the virtual server and, consequently, its hosted applications and IT resources as well.

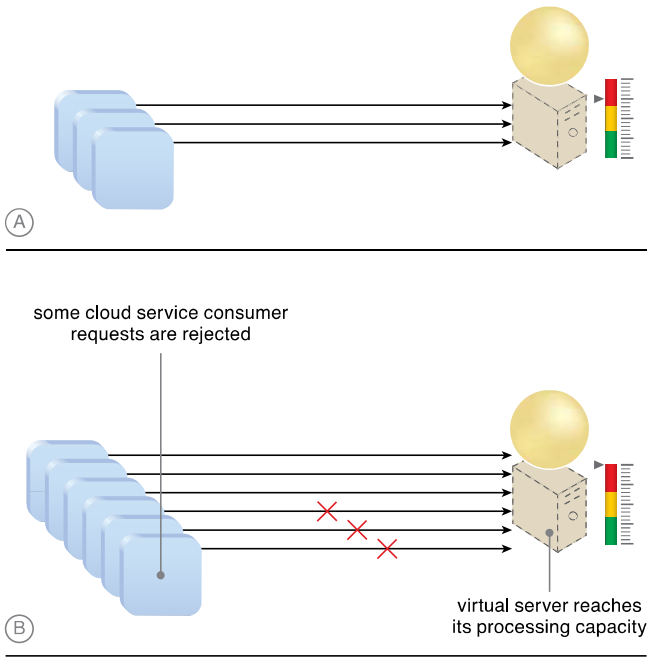


Figure 3.14
After the virtual server hosting the cloud service reaches its processing limit, subsequent cloud service consumer requests cannot be fulfilled.

Application

Resource Pooling (99) is applied to provision the necessary IT resource pools, and Dynamic Scalability (25) is applied to establish the automated scaling listener mechanism as an intermediary between cloud service consumers and any IT resources hosted by the virtual server that need to be accessed. Automated Administration (310) is further applied because intelligent automation engine scripts are needed to signal scaling requirements to the resource pool.

The following steps are shown in Figures 3.15 and 3.16:

1. Resource pools providing CPUs and RAM memory have been implemented and configured.
2. Cloud service consumers are actively sending requests.
3. The automated scaling listener is monitoring the requests.

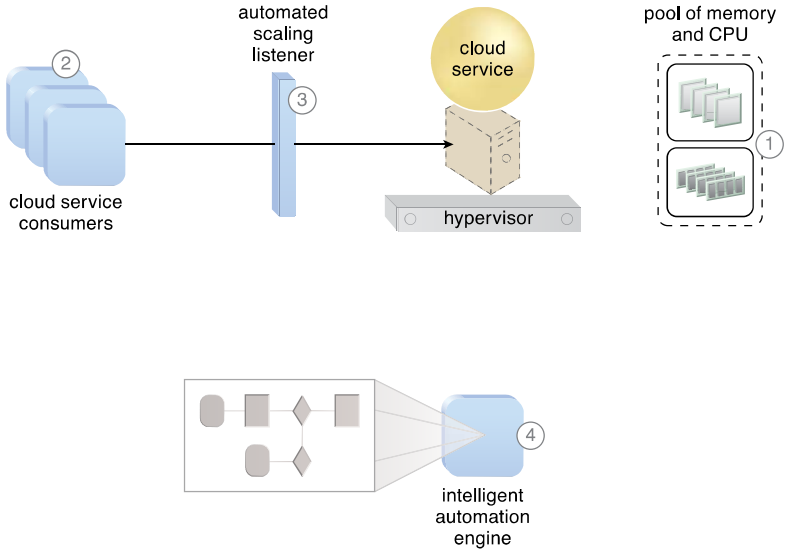


Figure 3.15
The application of the Elastic Resource Capacity pattern on a sample cloud architecture (Part I).

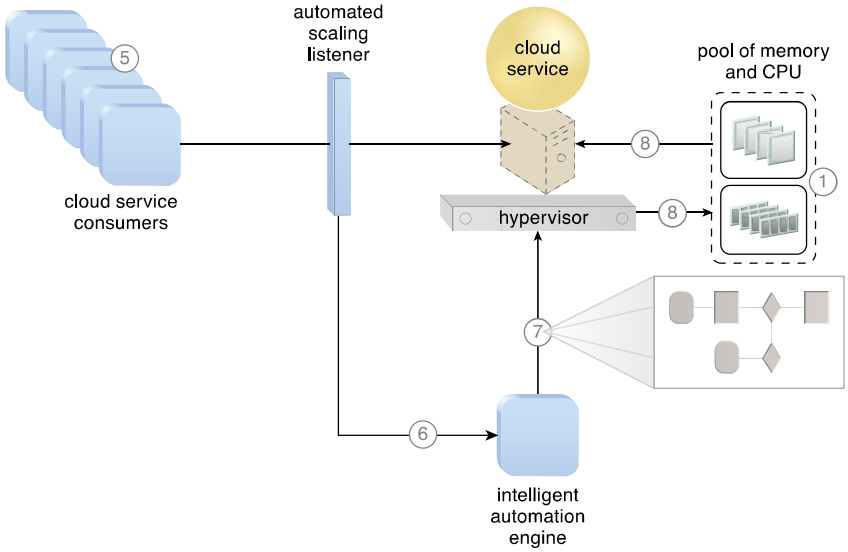


Figure 3.16
The application of the Elastic Resource Capacity pattern on a sample cloud architecture (Part II).

4. An intelligent automation engine script is deployed with workflow logic capable of notifying the resource pool using allocation requests.
5. Cloud service consumer requests increase.
6. The automated scaling listener signals the intelligent automation engine to execute the script.
7. The script runs the workflow logic that signals the hypervisor to allocate more IT resources from the resource pools.
8. The hypervisor allocates additional CPU and RAM to the virtual server, enabling it to handle the increased workload.

This type of cloud architecture may also be designed so that the intelligent automation engine script sends its scaling request via the VIM instead of directly to the hypervisor. Furthermore, in order to support dynamic resource allocation, virtual servers participating in elastic resource allocation systems may need to be rebooted for the allocation to take effect.

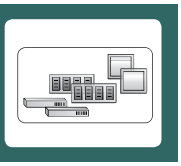
Mechanisms

- *Automated Scaling Listener* – The Elastic Resource Capacity pattern is reliant on the automated scaling listener mechanism for monitoring the workload and initiating the scaling process by indicating the type of scaling that is required.
- *Cloud Usage Monitor* – This mechanism is associated with the Elastic Resource Capacity pattern in how the cloud usage monitor collects the resource usage information of the allocated and released IT resources before, during, and after scaling, to help determine the future processing capacity thresholds of the virtual servers.
- *Hypervisor* – The hypervisor is responsible for hosting the virtual servers that house the resource pools that undergo dynamic reallocation. This mechanism allocates computing capacity to virtual servers according to demand, in alignment with the configurations and policies defined via the virtual infrastructure manager (VIM) by the cloud provider or resource administrator.
- *Live VM Migration* – If a virtual server requires additional capacity that cannot be accommodated by the current hypervisor, this mechanism is used to migrate the virtual server to another hypervisor that can offer the required capacity before adding more computing capacity at the destination.

- *Pay-Per-Use Monitor* – The pay-per-use monitor is related to this pattern in how the monitor is responsible for collecting all of the resource usage cost information, in parallel with the cloud usage monitor.
- *Resource Replication* – Resource replication is associated with this pattern in how this mechanism is used to instantiate new instances of the service or application, virtual server, or both.
- *Virtual CPU* – Processing power is added to virtual servers in units of gigahertz or megahertz, via the use of virtual CPU. This mechanism is used for allocating CPU according to the schedule and processing cycle of the virtual servers.
- *Virtual Infrastructure Manager (VIM)* – Virtual CPU and memory configurations are performed via this mechanism and forwarded to the hypervisors.
- *Virtual RAM* – Virtual servers are allocated the required memory via the use of this mechanism, which allows resource administrators to virtualize the physical memory installed on physical servers and share the virtualized memory between virtual servers. This mechanism also allows resource administrators to allocate memory in quantities greater than the amount of physical memory installed on the physical servers.
- *Virtual Server* – The Elastic Resource Capacity pattern relates to this mechanism in how virtual servers host the services and applications that are consumed by cloud consumers, and experience workload distribution when processing capacities have been reached.

Resource Pooling

How can IT resources be organized to support dynamic sharing?



Problem	When sharing identical IT resources for scalability purposes, it can be error-prone and burdensome to keep them fully synchronized on an on-going basis.
Solution	An automated synchronization system is provided to group identical IT resources into pools and to maintain their synchronicity.
Application	Resource pools can be created at different sizes and further organized into hierarchies to provide parent and child pools.
Mechanisms	Audit Monitor, Cloud Storage Device, Cloud Usage Monitor, Hypervisor, Logical Network Perimeter, Pay-Per-Use Monitor, Remote Administration System, Resource Management System, Resource Replication, Virtual CPU, Virtual Infrastructure Manager (VIM), Virtual RAM, Virtual Server

Problem

When assembling identical IT resources for sharing and scalability purposes (such as when applying Shared Resources (17) and Dynamic Scalability (25)), the IT resources need to carefully be kept synchronized so that no one IT resource differs from another.

Manually establishing and maintaining the level of required synchronicity across collections of shared IT resources is challenging, effort-intensive and, most importantly, error-prone. Variances or disparity between shared IT resources can lead to inconsistent runtime behavior and cause numerous types of runtime exceptions.

Solution

Identical IT resources are grouped into resource pools and maintained by a system that automatically ensures they remain synchronized (Figure 4.1). The following items are commonly pooled:

- physical servers
- virtual servers
- cloud storage devices

- internetwork and networking devices
- CPUs
- memory (RAM)

Dedicated pools can be created for each of these items, or respective pools can be further grouped into a larger pool (in which case each individual pool becomes a sub-pool).

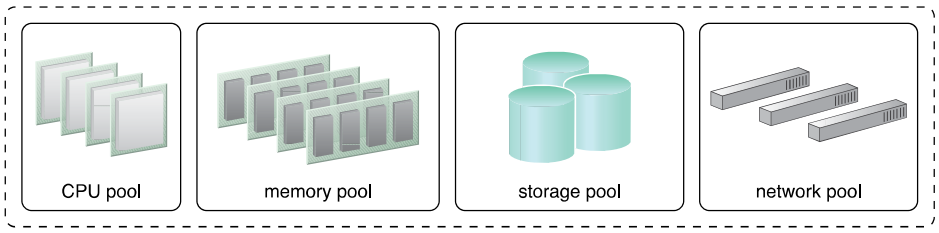
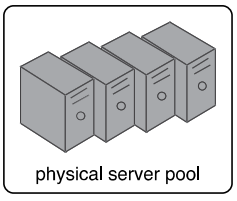


Figure 4.1
A sample resource pool comprised of four sub-pools of CPUs, memory, cloud storage devices, and virtual network devices.

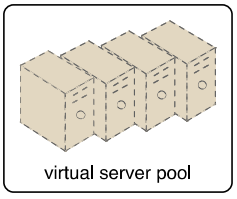
Application

As stated previously, this pattern is primarily applied in support of Shared Resources (17) and Dynamic Scalability (25) in order to establish a reliable system of shared IT resource synchronization. The Resource Pooling pattern itself can be further supported by the application of Resource Reservation (106).

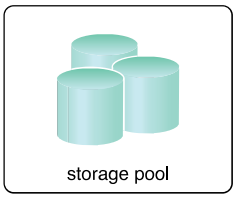
Provided here are common examples of resource pools:



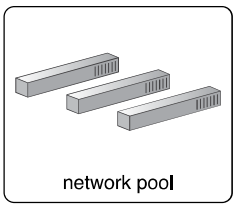
Physical server pools composed of ready-to-go, networked servers installed with operating systems and any other necessary programs or applications.



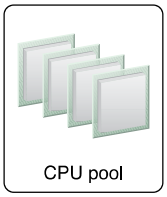
Virtual server pools are usually configured using templates that cloud consumers can choose from, such as a pool of mid-tier Windows servers with 4 GBs of RAM or a pool of low-tier Ubuntu servers with 2 GBs of RAM.



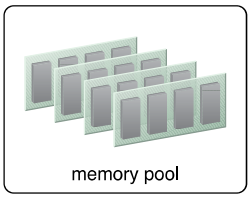
Storage pools (or cloud storage device pools) that consist of file-based or block-based storage structures. Storage pools can contain empty or filled cloud storage devices. Often storage resource pools will take advantage of LUNs.



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. These are often broken down into individual processing cores (as opposed to pooling entire CPUs).



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

Resource pools can grow to become complex, with multiple pools created for specific cloud consumers or applications. To help with the organization of diverse resource pools, a hierarchical structure can be established to create parent, sibling, and nested pools.

Sibling resource pools are normally drawn from the same collection of physical IT resources (as opposed to IT resources spread out over different data centers) and are isolated from one another so that each cloud consumer is only provided access to its respective pool (Figure 4.2).

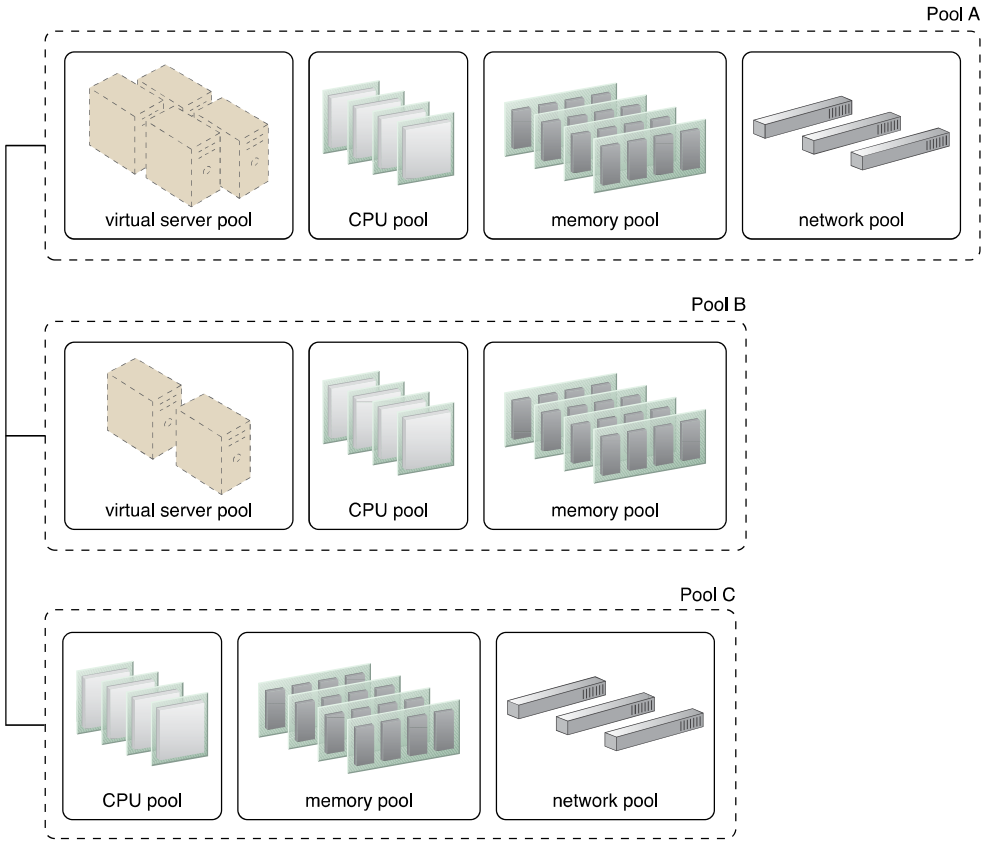


Figure 4.2
Pools B and C are sibling pools taken from the larger Pool A that has been allocated to a cloud consumer. This is an alternative to taking the IT resources for Pool B and Pool C from a general reserve of IT resources that is shared throughout the cloud.

In the nested pool model, larger pools are divided into smaller pools of the same kind (Figure 4.3). Nested pools can be used to assign resource pools to different departments or groups within the same cloud consumer organization.

After resources pools have been defined, multiple instances of IT resources from each pool can be created to provide an in-memory pool of "live" IT resources.

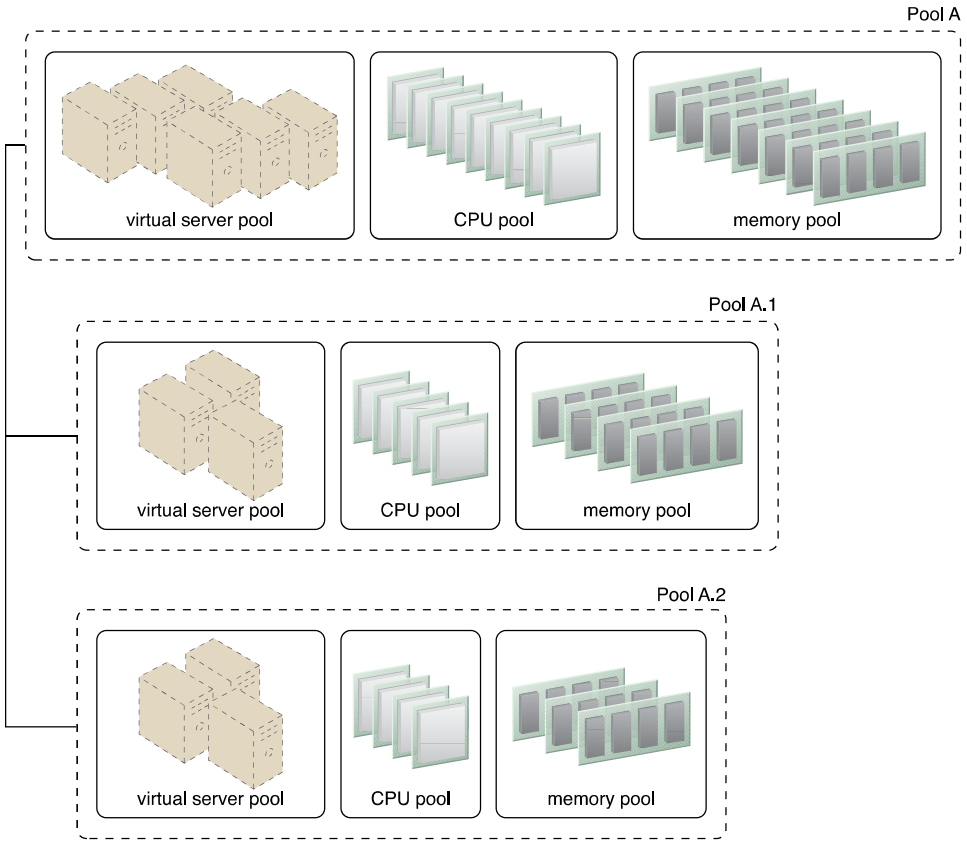


Figure 4.3

Nested Pools A.1 and A.2 are comprised of the same IT resources as Pool A, but in different quantities. Nested pools are generally used to provision cloud services that are rapidly instantiated using the same kind of IT resources with the same configuration settings.

Mechanisms

- *Audit Monitor* – This mechanism monitors resource pool usage to ensure compliance with privacy and regulation requirements, especially when pools include cloud storage devices or data loaded into memory.
- *Cloud Storage Device* – Cloud storage devices are commonly pooled as a result of the application of this pattern.

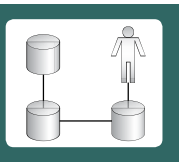
- *Cloud Usage Monitor* – Various cloud usage monitors can be involved with the run-time tracking and synchronization required by IT resources within pools and by the systems managing the resource pools themselves.
- *Hypervisor* – The hypervisor mechanism is responsible for providing virtual servers with access to resource pools, and hosting virtual servers and sometimes the resource pools themselves. Hypervisors further can distribute physical computing capacity between the virtual servers based on each virtual server's configuration and priority.
- *Logical Network Perimeter* – The logical network perimeter can be used to logically organize and isolate the resource pools.
- *Pay-Per-Use Monitor* – The pay-per-use monitor collects usage and billing information in relation to how individual cloud consumers use and are allocated IT resources from various pools.
- *Remote Administration System* – This mechanism is commonly used to interface with backend systems and programs in order to provide resource pool administration features via a front-end portal.
- *Resource Management System* – The resource management system mechanism supplies cloud consumers with the tools and permission management options to administer resource pools.
- *Resource Replication* – This mechanism can be used to generate new instances of IT resources for a given resource pool.
- *Virtual CPU* – This mechanism is used to allocate CPU to virtual servers, and also helps to determine whether a hypervisor's physical CPU is being over-utilized or a virtual server requires more CPU capacity. When a system has more than one CPU or when hypervisors belong to the same cluster, their total CPU capacity can be aggregated into a pool and leveraged by virtual servers.
- *Virtual Infrastructure Manager (VIM)* – This mechanism enables pools of resources to be created on individual hypervisors, and can also aggregate the capacity of multiple hypervisors into a pool from where virtual CPU and memory resources can be assigned to virtual servers.
- *Virtual RAM* – This mechanism is used to allocate memory to virtual servers, and to measure the memory utilization of hypervisors and virtual servers. When more than one hypervisor is present, a pool encompassing the combined memory

capacity of the hypervisors can be created. This mechanism is also used to identify whether more memory should be added to a virtual server.

- *Virtual Server* – This mechanism is associated with the Resource Pooling pattern in how virtual server hosted IT resources are provisioned and consumed by resource pools that are assigned to cloud consumers. Virtual servers themselves may also be pooled.

Redundant Storage

How can the reliability and availability of cloud storage devices survive failure conditions?



Problem	When cloud storage devices fail or become inaccessible, cloud consumers are unable to access data and cloud services relying on access to the device may also fail.
Solution	A failsafe system comprised of redundant cloud storage devices is established so that when the primary device fails, the redundant secondary device takes its place.
Application	Data is replicated from the primary storage to the secondary storage device. A storage service gateway is used to redirect data access requests to the secondary storage device, when necessary.
Mechanisms	Cloud Storage Device, Failover System, Resource Replication

Problem

Cloud storage devices are subject to failure and disruption due to a variety of causes, including network connectivity issues, controller failures, general hardware failure, and security breaches. When the reliability of a cloud storage device is compromised, it can have a ripple effect, causing impact failure across any cloud services, cloud-based applications, and cloud infrastructure program and components that rely on its presence and availability.

The following steps are shown in Figure 4.13:

1. The cloud storage device is installed and configured.
2. Four LUNs are created, one for each cloud consumer.
3. Each cloud consumer sends a request to access its own LUN.
4. The cloud storage device receives the requests and forwards them to the respective LUN.
5. The cloud storage device fails and cloud consumers lose access to their LUNs. This may be due to the loss of the device controller (5.1) or loss of connectivity (5.2).

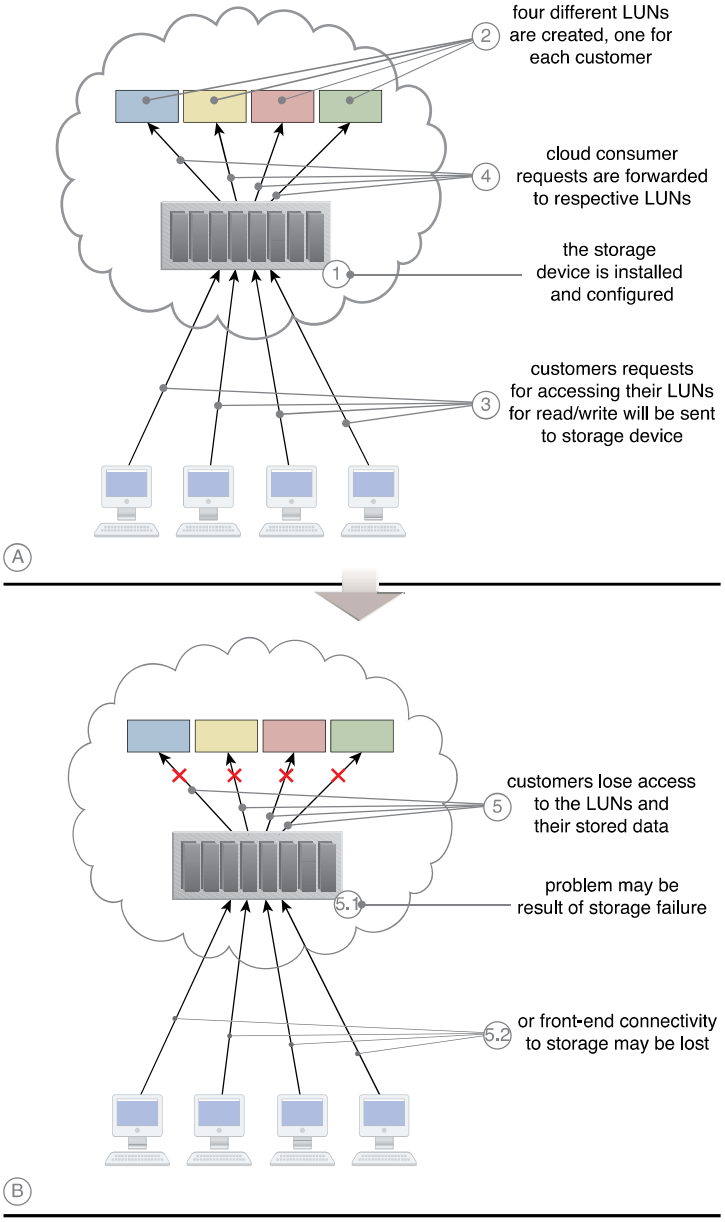


Figure 4.13

A sample scenario that demonstrates the effects of a failed cloud storage device.

Solution

A secondary redundant cloud storage device is incorporated into a system that synchronizes its data with the data in the primary cloud storage device. When the primary device fails, a storage service gateway diverts requests to the secondary device.

The following steps are shown in Figure 4.14:

1. The primary cloud storage device is replicated to the secondary cloud storage device on a regular basis.
2. The primary storage becomes unavailable and the storage service gateway forwards the cloud consumer requests to the secondary storage device.
3. The secondary storage forwards the requests to the LUNs, allowing cloud consumers to continue to access to their data.

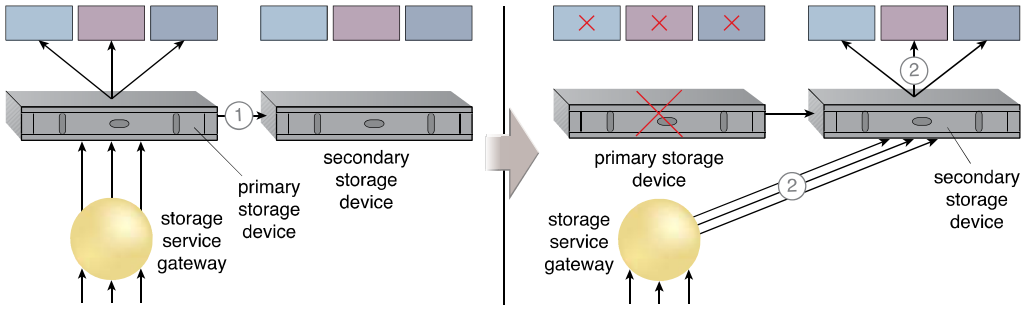


Figure 4.14
A simple scenario demonstrating the failover of redundant storage.

Application

This pattern fundamentally relies on the resource replication mechanism to keep the primary cloud storage device synchronized with any additional duplicate secondary cloud storage devices that comprise the failover system (Figure 4.15).

Cloud providers may locate secondary cloud storage devices in a different geographical region than the primary cloud storage device, usually for economic reasons. For some types of data, this may introduce legal concerns. The location of the secondary cloud storage device can dictate the protocol and method used for synchronization because some replication transport protocols have distance restrictions.

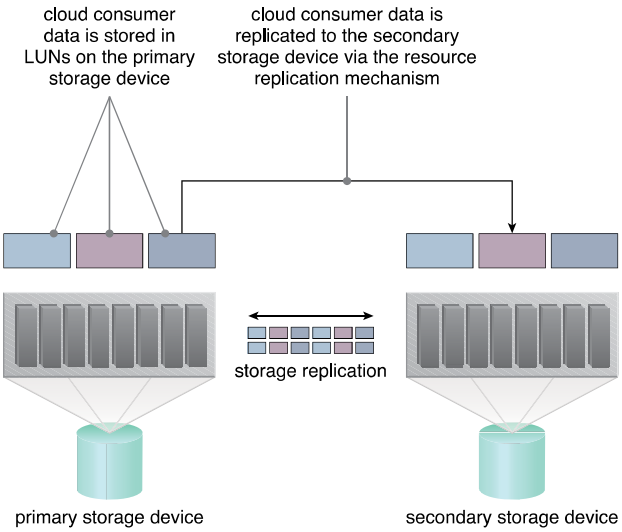


Figure 4.15
Storage replication is used to keep the redundant storage device synchronized.

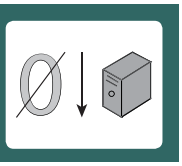
Some cloud providers use storage devices with dual array and storage controllers to improve device redundancy. They may place the secondary storage device in a different physical location for cloud balancing and disaster recovery purposes. In this case, cloud providers may need to lease a network connection via a third-party cloud provider, to establish replication between two devices.

Mechanisms

- *Cloud Storage Device* – This is the mechanism to which the pattern is primarily applied.
- *Failover System* – The application of the Redundant Storage pattern results in a specialized failover system based on the use of duplicate storage devices and a storage service gateway.
- *Resource Replication* – The failover system created by the application of this pattern relies on this mechanism to keep cloud storage devices synchronized.

Zero Downtime

How can downtime of virtual servers be avoided or eliminated?



Problem	It is challenging to provide zero downtime guarantees when a physical host acts as a single point of failure for virtual servers.
Solution	A fault tolerance system is established so that when a physical server fails, virtual servers are migrated to another physical server.
Application	A combination of virtual server fault tolerance, replication, clustering, and load balancing are applied and all virtual servers are stored in a shared volume allowing different physical hosts to access their files.
Mechanisms	Audit Monitor, Cloud Storage Device, Cloud Usage Monitor, Failover System, Hypervisor, Live VM Migration, Logical Network Perimeter, Physical Uplink, Resource Cluster, Resource Replication, Virtual CPU, Virtual Disk, Virtual Infrastructure Manager (VIM), Virtual Network, Virtual RAM, Virtual Server, Virtual Switch, Virtualization Agent, Virtualization Monitor

Problem

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.

Solution

A failover system is established so that virtual servers are dynamically moved to different physical server hosts in the event that their original physical server host fails. For example, in Figure 4.30, Virtual Server A is dynamically moved to another physical server host.

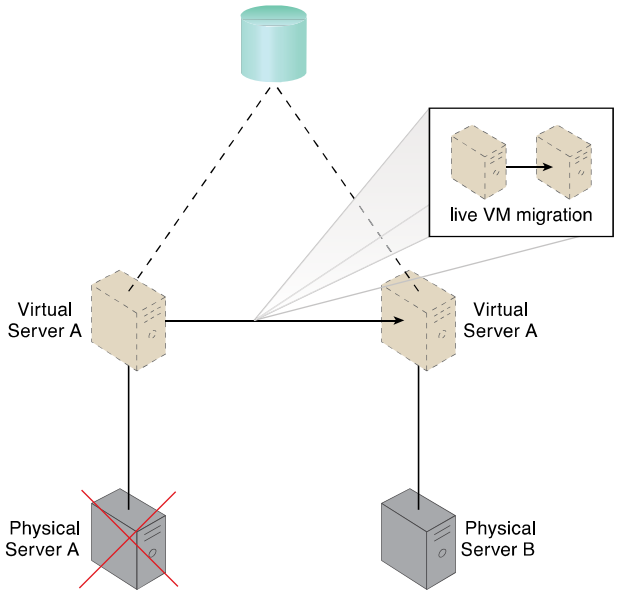


Figure 4.30
Physical Server A fails, triggering the live VM migration program to dynamically move Virtual Server A to Physical Server B.

Application

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. Resource cluster and live VM migration components are commonly part of this form of high availability cloud architecture.

The resulting fault tolerance assures that, in case of physical server failure, hosted virtual servers will be migrated to a secondary physical server. All virtual servers are stored on a shared volume (as per Persistent Virtual Network Configuration (227)) so that other physical server hosts in the same group can access their files.

Live storage replication can further be utilized to guarantee that virtual server files and hard disks remain available via secondary storage devices.

Mechanisms

- *Audit Monitor* – This mechanism may be required to ensure that the relocation of virtual servers does not relocate hosted data to prohibited locations.

- *Cloud Storage Device* – A cloud storage device is used to store virtual server network configuration data shared by the physical servers. It stores virtual servers and virtual disks in a central repository so that other available hypervisors can access the files and power on the failed virtual servers in case one of the hypervisors fails.
- *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.
- *Failover System* – The failover system can be used to switch from a failed primary physical server to a secondary physical server.
- *Hypervisor* – The hypervisor of each affected physical server hosts the affected virtual servers.
- *Live VM Migration* – When multiple instances of the same service or virtual server are provisioned for the purpose of redundancy and availability, this mechanism is used to seamlessly distribute different instances of the same service between different hypervisors to make sure one hypervisor will not become a single point of failure.
- *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.
- *Physical Uplink* – Physical uplinks are used and deployed in a redundant model, so that the virtual servers and services will not lose their connectivity to the cloud service consumers if a physical uplink fails or becomes disconnected.
- *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 new virtual server and cloud service instances upon primary virtual server failure.
- *Virtual CPU* – The virtual CPU mechanism is used to provide CPU cycling, scheduling, and processing capabilities to the virtual servers.
- *Virtual Disk* – This mechanism is used to allocate local storage space to the hosted virtual servers.

- *Virtual Infrastructure Manager (VIM)* – This mechanism is used to control the availability and redundancy of the virtual servers and services, and initiates proper command when rebalancing the environment or recreating a new instance of a service or virtual server is required.
- *Virtual Network* – This mechanism is used to connect virtual servers and the services hosted on top of them.
- *Virtual RAM* – This mechanism is used to establish access for the virtual servers and applications to the physical memory installed on the physical server.
- *Virtual Server* – This is the mechanism to which this pattern primarily applied.
- *Virtual Switch* – This mechanism is used to connect hosted virtual servers to the physical network and external cloud service consumers using physical uplinks.
- *Virtualization Agent* – Virtual servers use this mechanism to send regular heartbeat messages to the hypervisor. A recovery process is initiated if the hypervisor does not receive heartbeats after an extended period of time.
- *Virtualization Monitor* – This mechanism is used to monitor the virtual servers' availability and operational status.