Apache CloudStack Evolution Proposal

Alex Huang

Software Architect, Citrix Systems

Design Goals

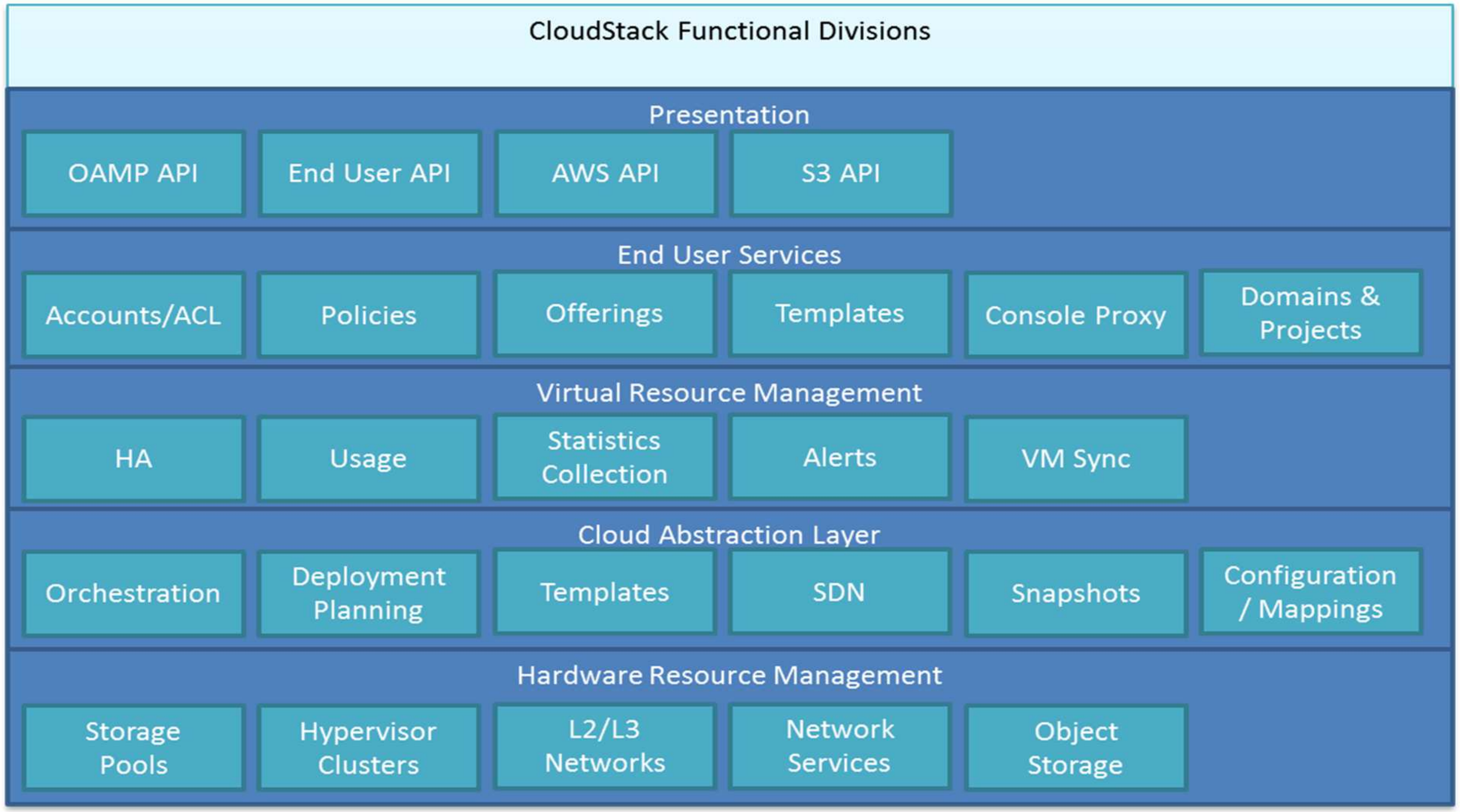
- Make it easier for developers to get started
- Allow developers with different skill sets to work on different parts of CloudStack
- Give operator the choice to deploy only parts of CloudStack that they want to use
- Allow CloudStack components to be written in languages other than Java
- Increase deployment’s availability and maintainability

Action Plan

- Disaggregate CloudStack services
- Clearly differentiate between automation, orchestration, and provisioning
- Switch to using well-known frameworks
- Allow better composition at the resource layer
- Change the deployment model for better resiliency

Disaggregating CloudStack

CloudStack Functional Layers



Pros & Cons

Pros

- Easy for a small team to develop in
- Easy to deploy

Cons

- Interdependency in these layers causes reliability problems.
  - Contracts between layers cannot be enforced since each layer cannot be individually tested.
- Developer skill set must range from API design all the way to system level programming to effectively code in CloudStack
- CloudStack availability and maintainability suffers because layers with different availability and maintainability requirements are deployed in one process.

Action Plan

Service	Purpose
Cloud-Engine	- Presents a data center abstraction layer - Orchestration within the data center abstraction layer - Provisioning of the physical resources - Directory for services and service end points
Cloud-Access	- Account and directory connectors - Authentication - ACL & Governess
Cloud-API	- End User API & UI
Cloud-Management	- Management of physical resources - Data Center automation - Admin UI

CloudStack Service Properties

- Independent life cycle
- Independent scaling
- Independent testing
- RPC through reliable message queue
- Notification through event systems
- Individual database (even further in the future)

Cloud-Engine vs Cloud-API

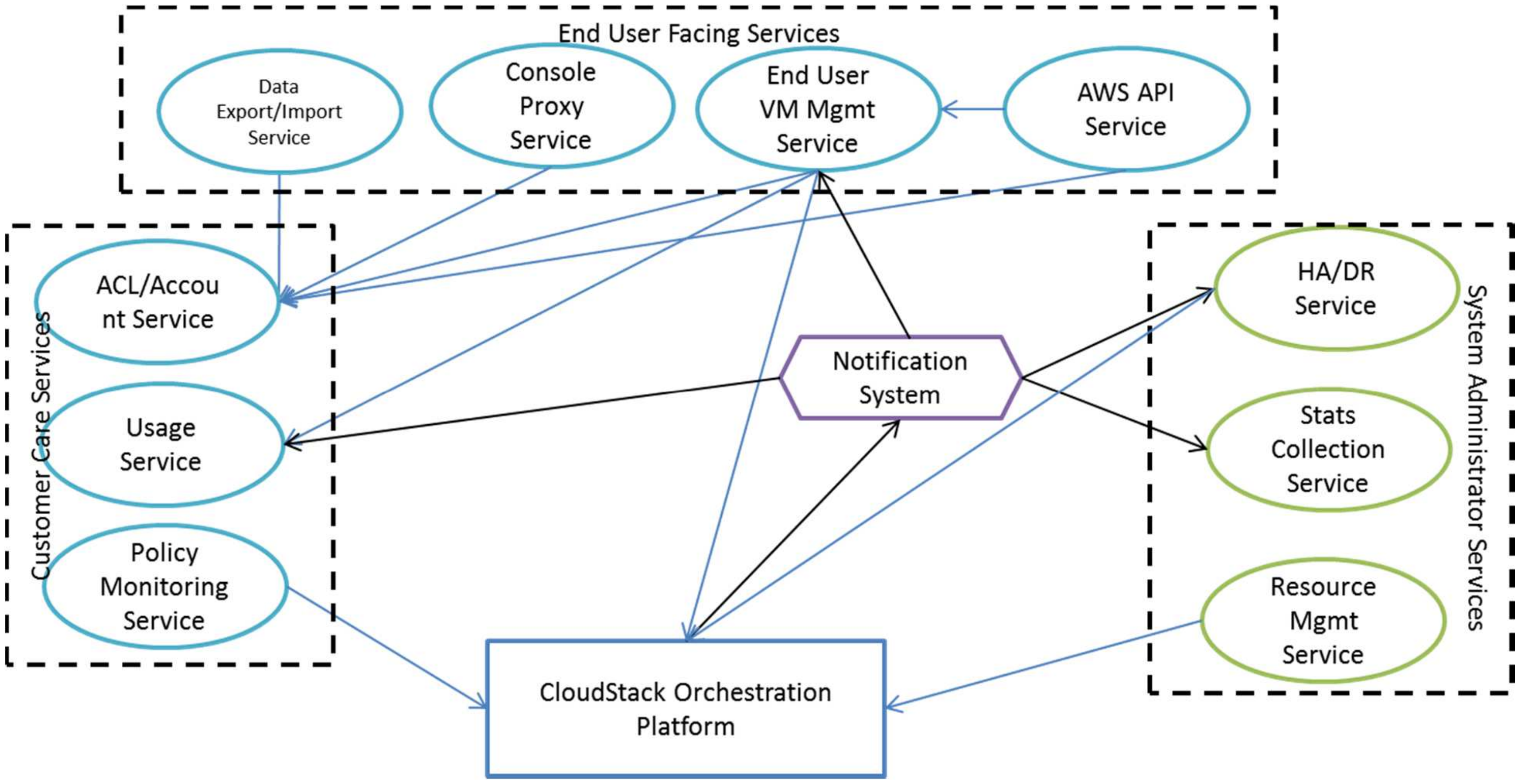
Data Center Abstraction API

- Speaks in virtualization terms (CPU, RAM, etc)
- Callers can specify deployment destination down to the host
- Can be used to deploy service VMs (such as SSVM and VR)
- Contains orchestration logic

Cloud API

- Speaks in service contracts (service offerings, network offerings, disk offerings)
- Callers can only specify deployment destination through resource dedication
- Can only deploy user VMs
- Contains business logic

A Possible Future



Automation, Orchestration, & Provisioning

What is the difference?

- The key is the data center abstraction layer
  - Virtual Machine, Template, Nic, IP Address, Volume, Network, Rules, Snapshot
- Orchestration orchestrates within this abstraction layer
- Provisioning manifests concepts in abstraction layer on physical resources
- Automation orchestrates above the data center abstraction layer to bring greater functionality

Examples

- Orchestration
  - VM deploy, Volume creation, Network Creation, Network rules propagation
- Provisioning
  - Starting a VM on a hypervisor
  - The actual movement of a volume from one storage pool to another
- Automation
  - HA Process
  - Putting a resource into maintenance mode
  - Uploading and downloading templates
  - DR process

Why is this important?

- Cloud-Engine is still too big.
- Plugin partners need to clearly see the division in functionality between Cloud-Engine and their plugin.
- Disaggregating CloudStack Services allow developers to quickly add services utilizing Cloud-Engine
- Disaggregating Cloud-Engine allows partners to add more infrastructure to be utilized in the cloud.

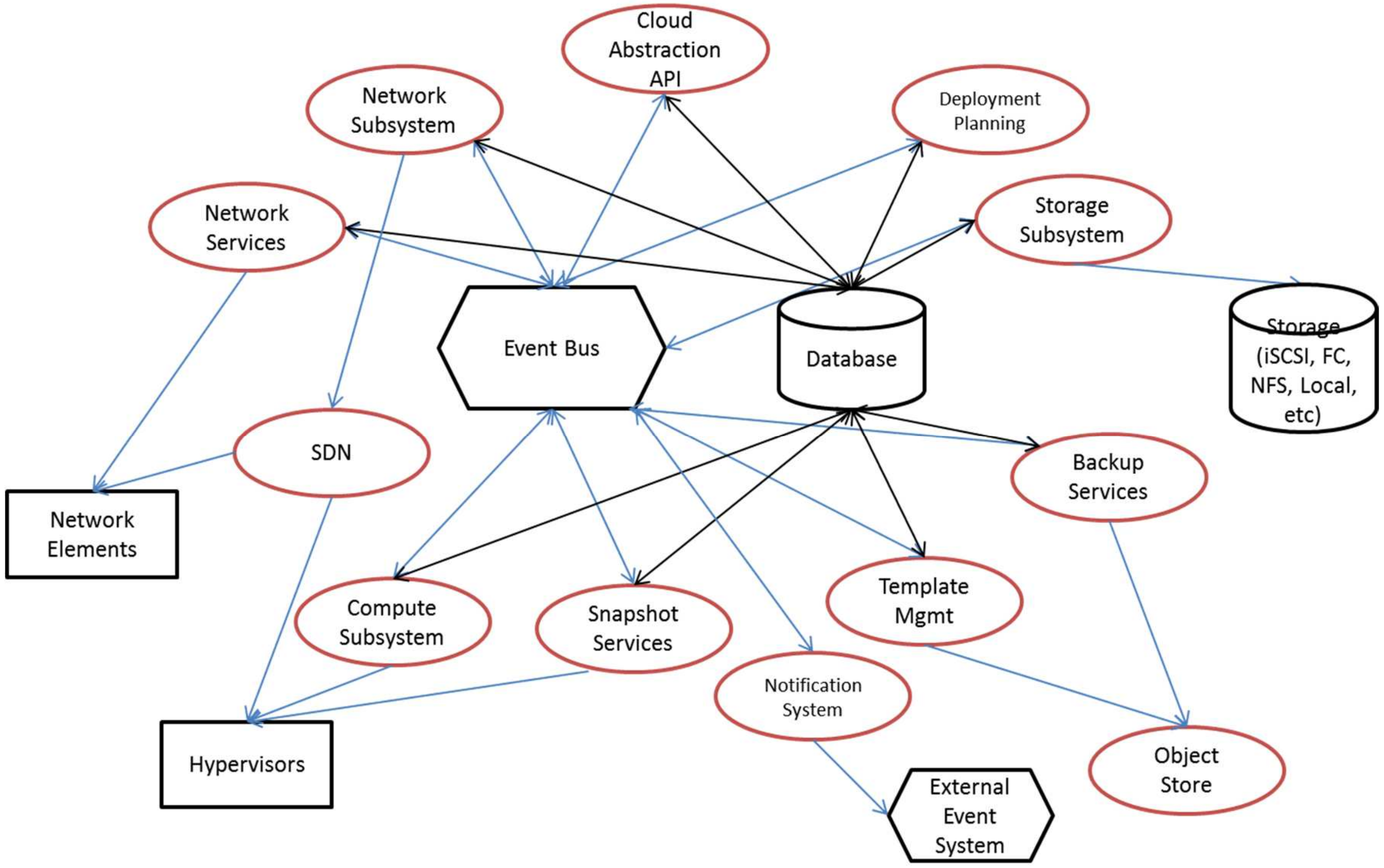
Cloud-Engine Components

Component	Purpose
Orchestration	- Orchestration of the Data Center Abstraction Layer
DeploymentPlanner	- Plans the deployment destination for virtual machine and volumes
Compute	- Provisioning of the hypervisor
NetworkGuru	- Provides mapping of Network to physical network
NetworkElement	- Provides various network services
PrimaryDataStore	- Provisioning of storage
ImageStore	- Provisioning of templates
BackingStore	- Provisioning of backup storage
SnapshotService	- Provides volume snapshots
MotionService	- Provides data movements between various storage technologies

Cloud-Engine Component Properties

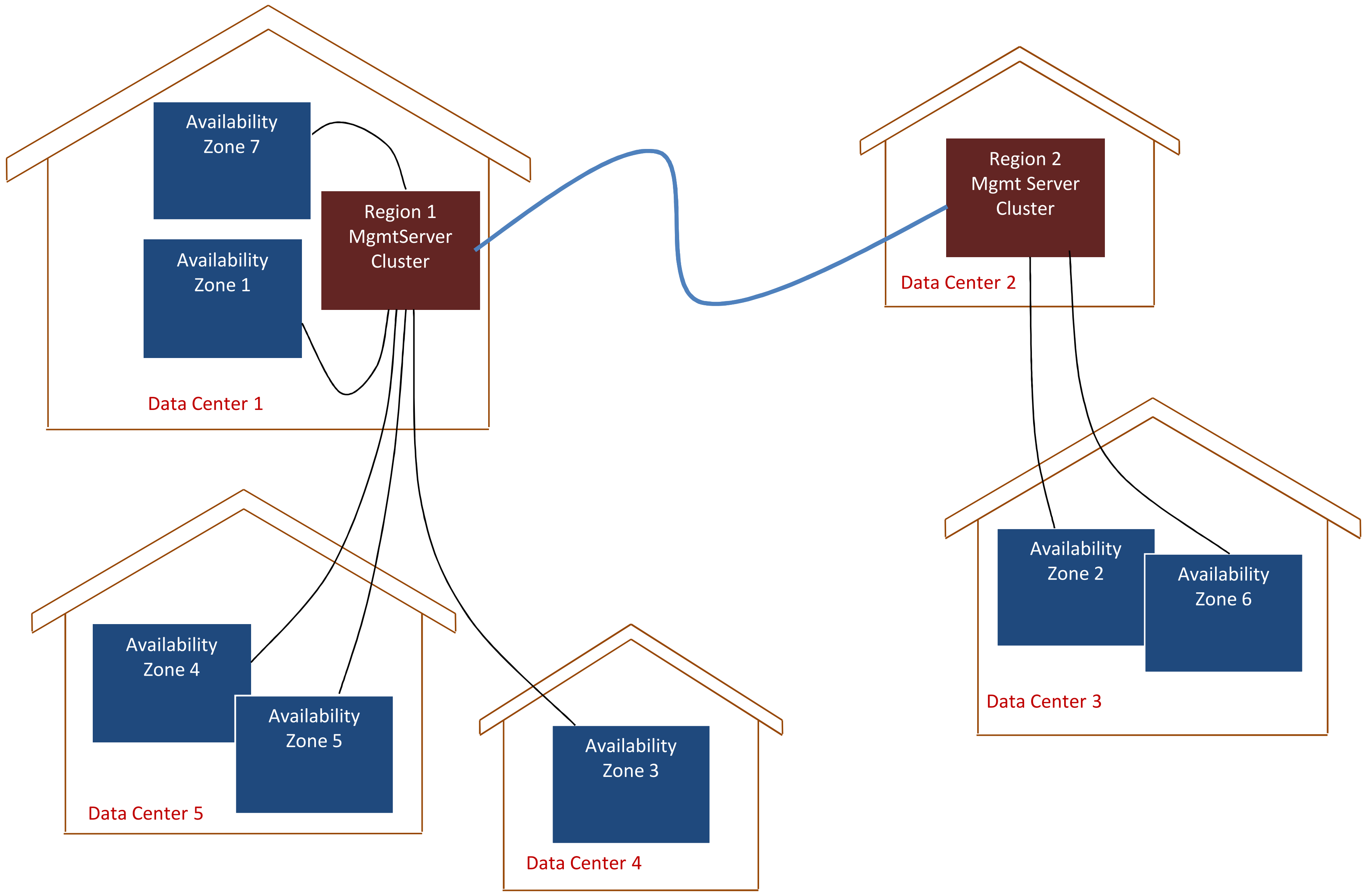
- Recommended to have independent life cycles, databases, scaling, and testing.
- Utilize CloudStack’s plugins to bridge provisioning needed by Cloud-Engine and functionality provided by the component.
- All APIs must be asynchronous.
- Operations are idempotent.

Cloud-Engine Components



Changing CloudStack’s Deployment Model

CloudStack 4.0



Pros & Cons

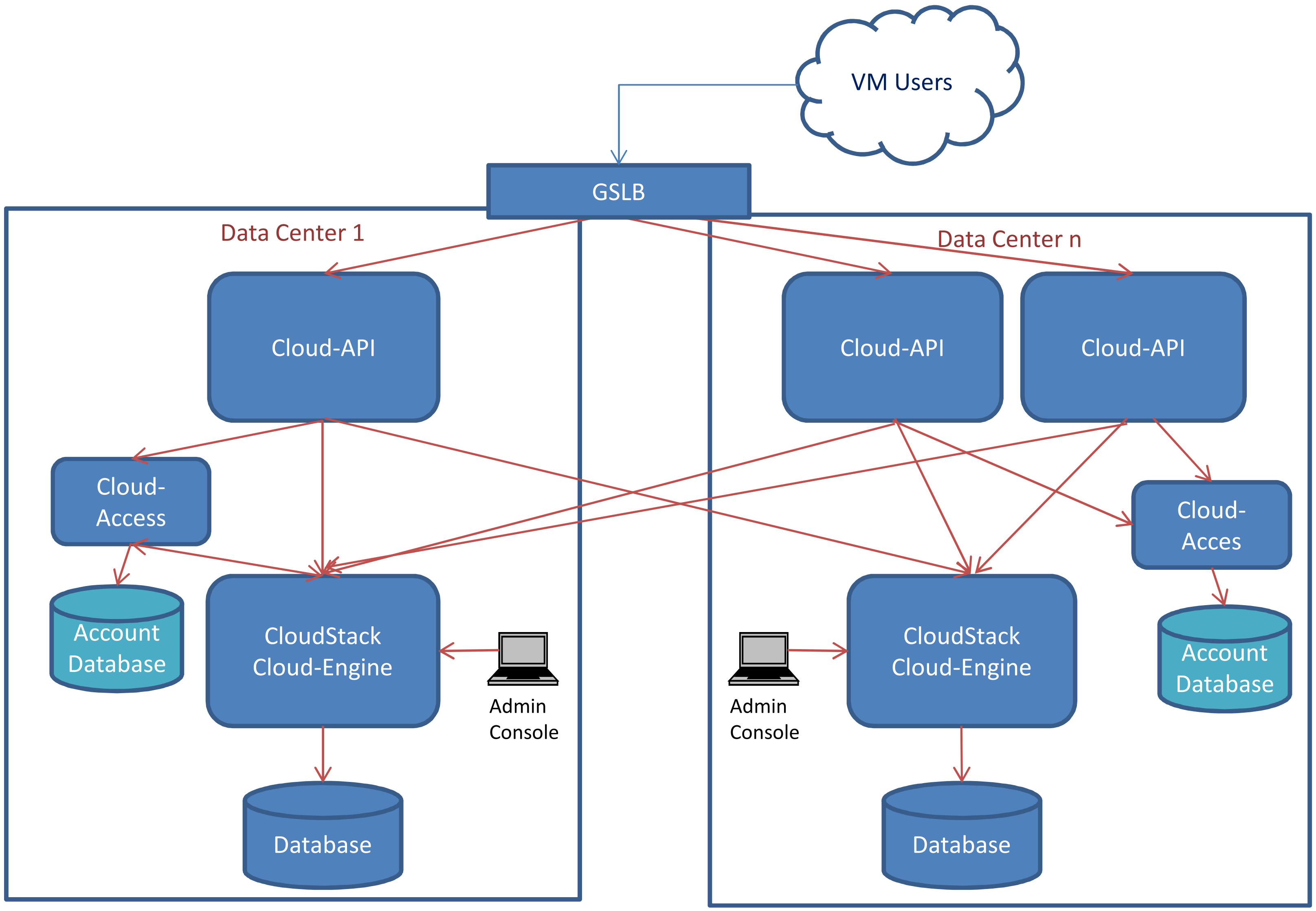
Pros

- Simple deployment model
- Easy to track jobs

Cons

- Management plane goes down, the entire cloud is not operable.
- No fault containment to the availability zone
- Unable to do a zone by zone upgrade of CloudStack
- Cannot guarantee zero downtime upgrades

New Deployment Model



Scalability

- Cloud-API nodes can be brought up and added to cluster to handle more requests
- Cloud-Engine cluster and Cloud-API cluster are scaled independently
  - Cloud-Engine cluster to hardware resources
  - Cloud-API cluster to incoming requests

Availability

- Cloud-API Servers can be deployed in geographically remote locations because they don’t share databases
- One Cloud-API Server going down only impacts the tasks it is executing
- Any number of Cloud-API Servers can be brought up
- Cloud-Engine cluster going down means only one zone is down. Not the whole cloud.
- Even if the entire Cloud-API cluster is down, admins can still manage VMs by directly connecting to the Cloud-Engine cluster.

Maintainability

- Zones can be individually upgraded
- Only the zone being upgraded cannot be provisioned
- Cloud-API Servers can be brought up with new versions and then the old ones shutdown