Understanding CloudStack Internals

Rohit Yadav

Software Architect, ShapeBlue

Apache CloudStack Committer/PMC Member

rohit.yadav@shapeblue.com

@_bhaisaab | bhaisaab.org

Shape Blue The Cloud Specialists

apachecloudstack open source cloud computing

About Me

- Software Architect at ShapeBlue
- Apache CloudStack Committer since 2012, recent PMC Member
- Author of CloudMonkey, SAML2 plugin, API Discovery Service; Maintainer of various subsystems including API/Auth/DB layers, systemvms, VRs, KVM, build system, packaging, upgrade paths and overall codebase
- 3rd party integrations and feature development, firefight ACS Clouds, Customer PoCs, Production deployment and upgrades including upgrades from CCP to ACS, ACS Patching and Packages hosting

About ShapeBlue

"ShapeBlue are expert builders of public & private clouds. They are the leading global Apache CloudStack integrator & consultancy"

Shape Blue The Cloud Specialists


sky
centrica
orange
CISCO
colt
SUNGARD
Availability Services
eSkyCity
Virtela
An NTT Communications Company
interoute
from the ground to the cloud
BrokerBin.com
The B2B Hardware Exchange
Ascenty
EVRY
Trader
Media Group
Paddy Power
CITRIX
BBC
TOMTOM
Slovak
Telekom

Understanding CloudStack Internals

Agenda

- Getting started as a user or a developer
- Joining the community and becoming a contributor
- A guided tour of the ACS architecture codebase
- Getting started with CloudStack Development
- General development and maintenance patterns
- Deep dive: SystemVMs and Virtual Routers, Networking Implementation, Plugins

Getting started as a User

- http://cloudstack.apache.org
- Joining the Community: Meetups, Confs
  - IRC: #cloudstack, #cloudstack-dev on irc.freenode.net
  - Users ML: users-subscribe@cloudstack.apache.org
  - Dev ML: dev-subscribe@cloudstack.apache.org
- Docs: Release Notes, Install, Admin, API docs, Wiki
- Downloads: Source releases, Packages and Repositories
- Play with APIs and CloudMonkey

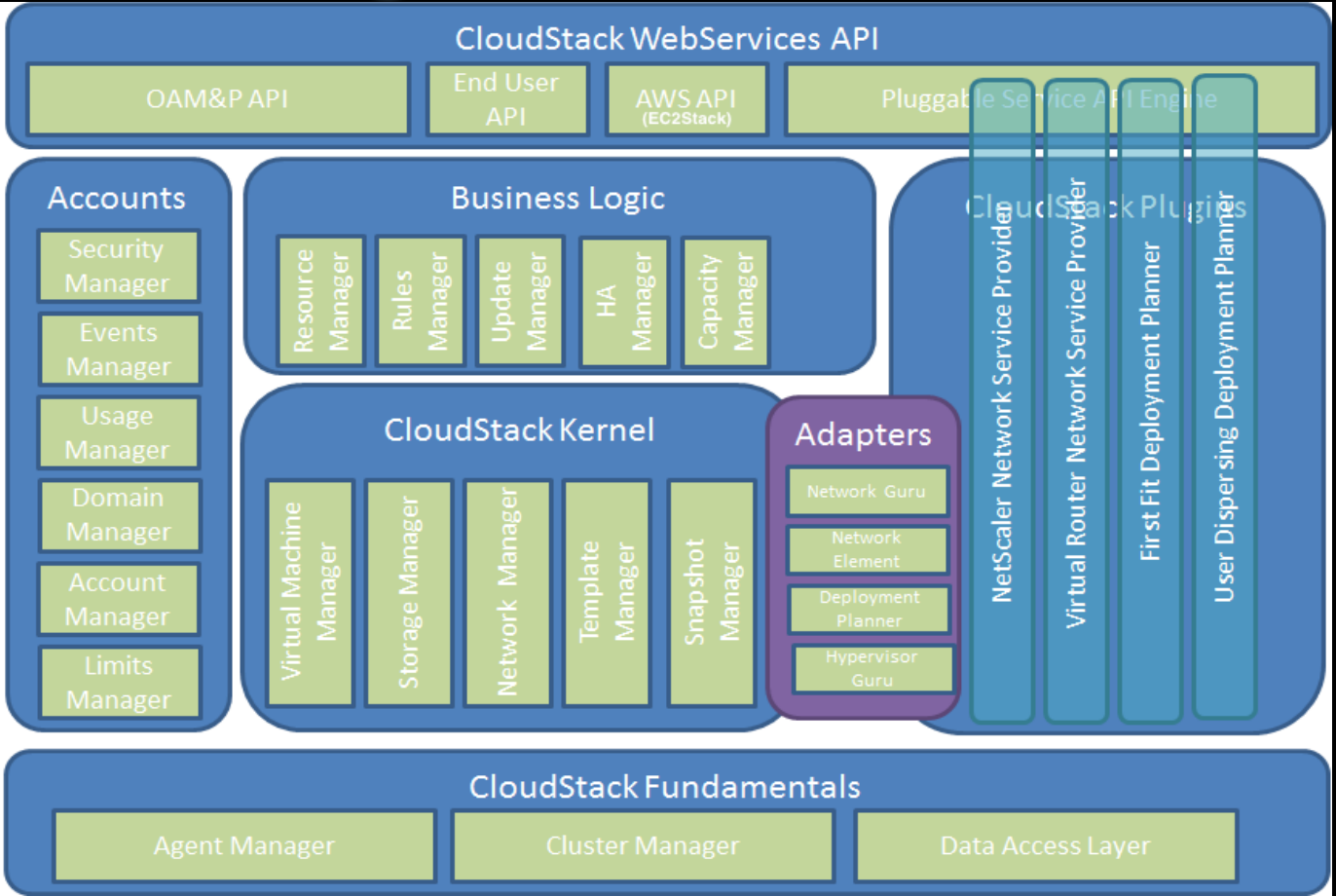
How to Contribute

- Discuss issues on MLs and JIRA:
  https://issues.apache.org/jira/browse/CLOUDSTACK
- Github Pull Requests:
  https://github.com/apache/cloudstack
  https://github.com/apache/cloudstack-cloudmonkey
  https://github.com/apache/cloudstack-ec2stack
  https://github.com/apache/cloudstack-gcestack
  https://github.com/apache/cloudstack-www
  https://github.com/apache/cloudstack-docs
  https://github.com/apache/cloudstack-docs-admin
  https://github.com/apache/cloudstack-docs-install
  https://github.com/apache/cloudstack-docs-rn

Getting started as a Developer

- Cwiki: CloudStack 101
- Tools: Git, Maven, Java, Python, MySQL server, NFS, Text Editor or IDE (IntelliJ/Eclipse)
- How to build/deploy/run; CloudStack, Database etc.
- How to test and automate: CloudMonkey, Marvin
- Reviews and QA : TravisCI, Jenkins (jenkins.buildacloud.org)
- CloudStack Developer Kit: Langur (coming soon)

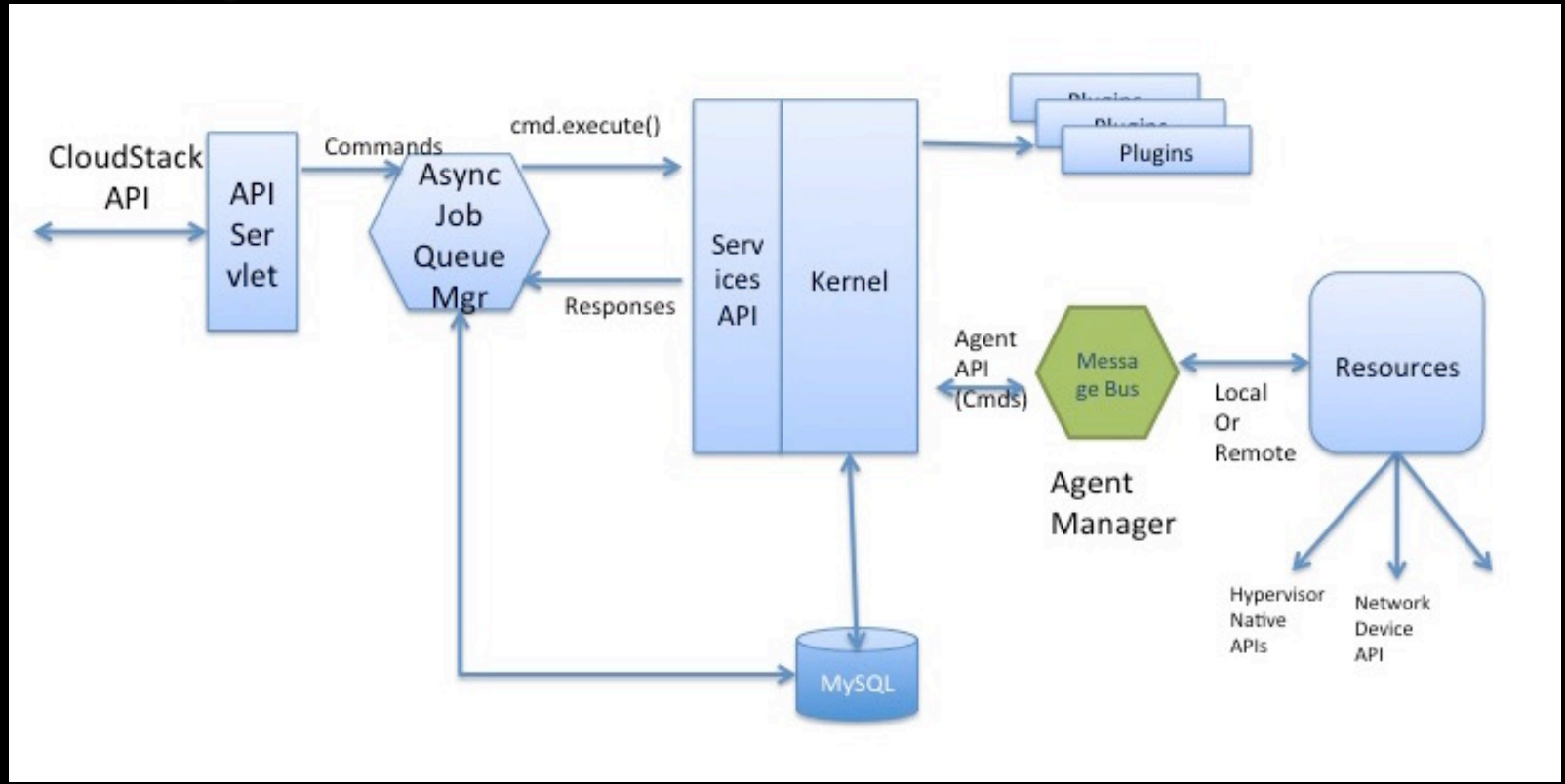
High Level Layers



Understanding Codebase

- Guided tour of the codebase
- Maven projects and project dependencies
- Maven Build Profiles, options
- Packaging

Typical CloudStack Interaction



Common Development Patterns

- Core features
- Extending an Interface to implement a plugin: Hypervisor, Storage, Network, Security, Authentication etc.
- Asynchronous Event Consumers
- Using RPC based approach: Thrift
- Extending common interfaces; or refactoring codebase to create new interfaces to implement plugins on

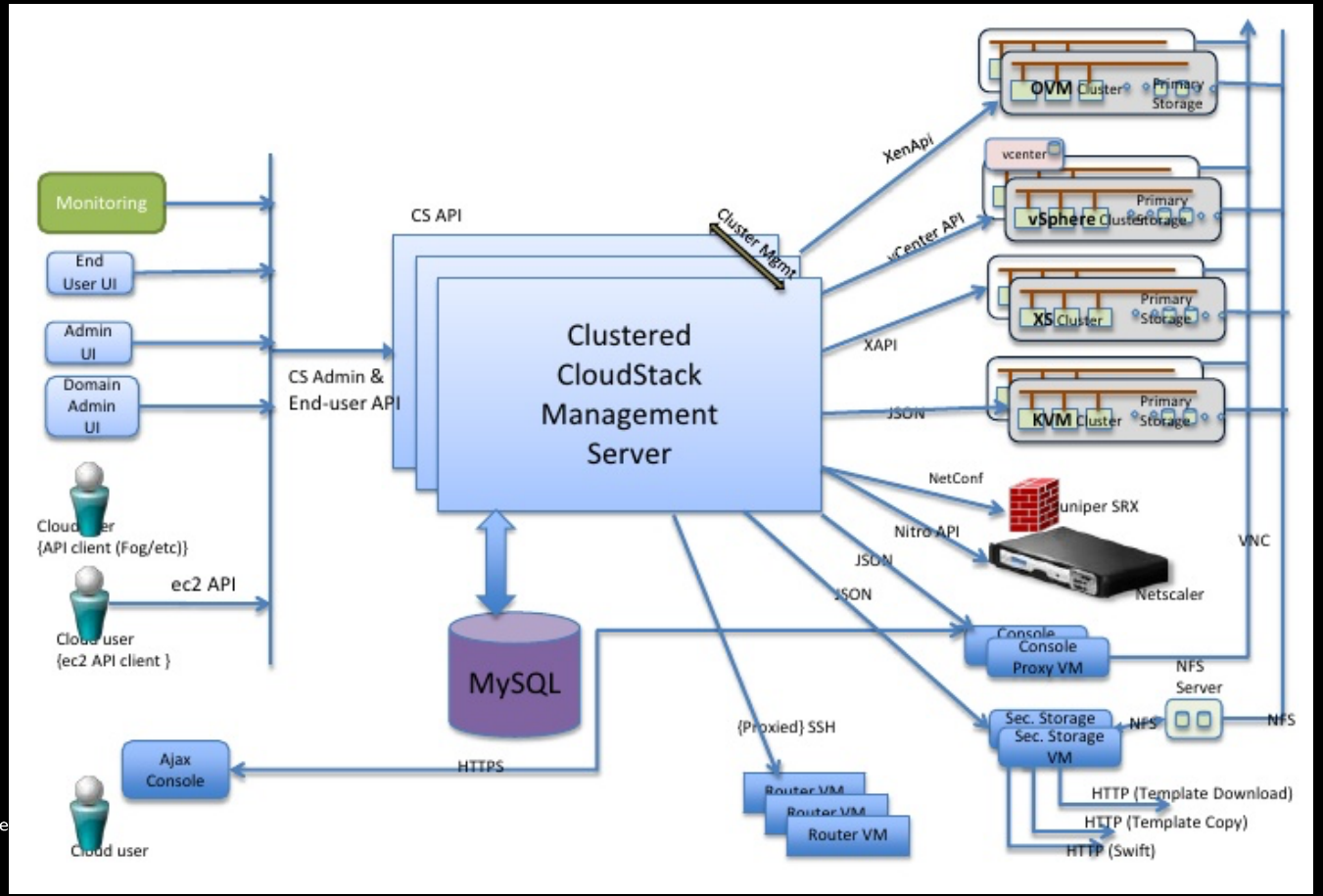
Components

- Manager: Singleton, controls a process. Example: VirtualMachineManager
- Adapter: Different ways to implement same functionality
- DAO: Data Access Object, DB operations per table
- VO: View Object, row in a table
- Service: Platform API. Example: UserVmService
- PluggableService: Interface to define platform APIs
- SystemIntegrityChecker: Interface to define system integrity checkers
- ComponentLibrary: Collection of Managers, Adapter, DAOs
- Interceptor: Aspect Oriented programming patterns
- ServerResource: ACS translation between CloudStack operations and how to perform operations on a physical resource

Common Plugin Interfaces

- NetworkGuru: Network isolation and ip address technologies
- NetworkElement: Network services to support a VM (DNS, DHCP, VPN, LB, PF etc)
- DeploymentPlanner: Algorithms to place VMs and Volumes
- Investigator: What went wrong, host was down
- Fencer: For fencing off a VM
- UserAuthenticator: Authenticating users
- SecurityChecker: ACL Access
- HostAllocator and StorageAllocator: Algorithm to allocate host and storage for a VM
- APIAuthenticator: Inteface to build your own authentication plugin

CloudStack Interactions



Internals

- API and Auth layers, Cmd Pattern, APIs and AgentShell
- Plugins: User Authenticators, Allocators
- Schema, Upgrades
- SystemVMs and Virtual Router
- Hypervisors: KVM, XenServer, VMWare etc.
- Networking: SG, Isolated Networks and VPCs
- Storage: NFS, local storage etc
- Scheduling, Jobs, Spring DI

Internals: API/Auth Layer

- Heavy use of the Command pattern in APIServer and AgentShell
- Heavy use of reflections and annotations
- APIs: BaseCmd (Sync), BaseAsync*Cmd (Async)
- Agents: AgentShell and Command/Answer
- Own serializing and deserializing methods: Request.java , Response.java (json based)
- Agents: Connected Agent (8250), Direct Agent, Forwarding Agent
- AgentManagerImpl implements magic for connected/direct agents
- ClusteredAgentManagerImpl extends AgentManagerImpl b/w mgmt servers (9090). ClusterManager checks mshost up/down state
- AgentManager registers events on mgmt server etc

Plugins

- User Authenticators: PBKDF2, SHA256, MD5 etc.
- Security Layer
- Implementation of custom authenticators and authentication plugin
- SAML example and auth plugin architecture

Internals: Schema Upgrades

- All upgrade paths implement DbUpgrade interface
- DatabaseUpgradeChecker implements SystemIntegrityChecker, responsible for upgrading on startup
- Static upgrade paths, gets source version from cloud.version table
- Upgrade process: Stop all mgmt server, upgrade one server and start all mgmt servers

CloudStack Agents

- Connected Agents are handled by AgentManagerImpl.java, AgentHandler class embedded within AgentManagerImpl.java, and ConnectedAgentAttache.java. The tcp connection itself is handled by a NioConnection class.
- Direct Agents are handled by AgentManagerImpl.java, DirectAgentAttache.java
- Forwarding Agents are handled by ClusteredAgentManagerImpl.java, ClusteredAgentAttache.java

Internals: SystemVMs and VRs

- SystemVMs: Owned by SYSTEM user, for ACS operations
- Secondary Storage VM: Template/ISO, Snapshots etc
- Console Proxy VM: VM Console
- Virtual Routers: CloudStack's own SDN appliance
- Patched by systemvm.iso, ssh port 3922
- Common Debian7 template/image
- Logs, partitions, recent changes

VR/Networking Implementation

- Shared network implementation
- SG implementation is based on network bridges on hypervisor host
- Isolated and VPC network implementation uses VR
- Building blocks: bridges, ipsets, iptables rules and ebtable rules and additional services
- Why SG is not available on VMWare but on KVM/XenServer

Storage Implementation

- Storage Subsystem and Driver interface
- Still depends on hypervisor implementation
- Local storage and shared storage
- Image and Volume storage split
- SSVM agent and interactions

QA

Thank You!

Please join the CloudStack Community!

users-subscribe@cloudstack.apache.org
dev-subscribe@cloudstack.apache.org