



IaaS Quick Build Guide

for CloudStack-based environments



— We Are the CloudStack Company —



Table of Contents

1	Introduction	3
2	Concepts and architecture	4
2.1	Overall CloudStack Concept.....	4
2.1.1	Architecture Overview	4
2.1.2	Zones.....	5
2.1.3	Pods	5
2.1.4	Clusters	5
2.1.5	Primary Storage.....	5
2.1.6	Secondary Storage.....	5
2.1.7	Instances	6
3	Installation and Setup.....	6
3.1	Overview	6
3.2	Prerequisites.....	6
3.3	Environment.....	7
3.3.1	Operating System.....	7
3.3.2	Configuring the network	8
3.3.3	Hostname	11
3.3.4	SELinux in EL8.....	11
3.3.5	Chrony	12
3.3.6	Configuring the CloudStack Package Repository	12
3.3.7	Setup Storage.....	13
3.4	Management Server Installation.....	14
3.4.1	Database Installation and Configuration	14
3.4.2	Installation	15



3.5 KVM Setup and Installation	16
3.5.1 Prerequisites.....	16
3.5.2 Installation	16
3.5.3 KVM configuration complete	17
3.6 Configuration	18
3.6.1 UI Access	18
4 Setting Up a Zone	19
4.1 Setup Zone.....	19
4.2 Setup Network	20
4.3 Add Resources.....	23
5 Conclusion.....	28



1 Introduction

As more and more companies build internal private clouds or enter the service provider market with public clouds, the more they will need the right set of tools to launch, manage and scale easily the Infrastructure as a Service (IaaS) platform. However – choosing the right technology stack can be a difficult decision. Starting from hardware, hypervisor, storage and the cloud management platform, there is a long list of choices to make, to ensure the platform will be reliable and will serve in the best possible way long-term. There are several aspects that should be considered, such as planning for future growth and demand, team size, budget, project timeframe, previous experience, available hardware and the underlying infrastructure already in place.

When it comes to cloud management (or cloud orchestration) platforms, the first thing we need to clarify is what we mean by this. A typical cloud management platform allows one to take existing datacentre infrastructure and wrap around it a common API, CLI and user interface, which allows an organization to benefit from the basic concepts of cloud computing in terms of elasticity, metered usage, self-service and resource pooling.

For some, the first choice to make is whether to go for an open-source or a proprietary/vendor solution. This may not come as a surprise to anyone, but at ShapeBlue, we passionately believe the solution should be 100% open-source, and following years of experience, feedback from users and developers, testing most other solutions and working with the community, we also passionately believe the solution should be Apache CloudStack. Apache CloudStack is simple to deploy, has a low cost of ownership, is production proven and highly scalable.

This ShapeBlue IaaS Quick Build Guide provides a straightforward set of instructions to get you up and running with a CloudStack-based Infrastructure-as-a-Service (IaaS) cloud, on a single server.

We have chosen a quick, proven way to easily build an IaaS platform. CloudStack supports a broad range of hypervisor, network, storage and compute technologies, but here we have chosen a simple configuration using KVM on EL8 or Ubuntu, NFS storage and layer-2 isolation using VLANs. Obviously, normal production environments usually have more complex configurations and are at a much larger scale.



CloudStack is an open-source IaaS cloud orchestrator that powers some of the world's largest clouds in hands of various ISPs, Telcos and general cloud providers. Its monolithic architecture allows it to be easily implemented, without a need to do hundreds of hours of development in order to provide a bare workable installation. Its simplicity, performance, scalability but also manageability, makes it a default go-to cloud orchestrator in, virtually, all cloud scenarios. It supports various hypervisor types, various storage solutions and can provide some very complex tenant networking models.

2 Concepts and architecture

2.1 Overall CloudStack Concept

Apache CloudStack is open-source software designed to deploy and manage large networks of virtual machines, as a highly available, highly scalable Infrastructure as a Service (IaaS) cloud computing platform.

2.1.1 Architecture Overview

CloudStack is a Cloud Orchestration platform that pools computing resources to build public, private and hybrid Infrastructure as a Service (IaaS) clouds. CloudStack manages the network, storage, and compute nodes that make up a cloud infrastructure.

A CloudStack cloud has a hierarchical structure which enables it to scale to manage tens of thousands of physical servers, all from a single management interface.

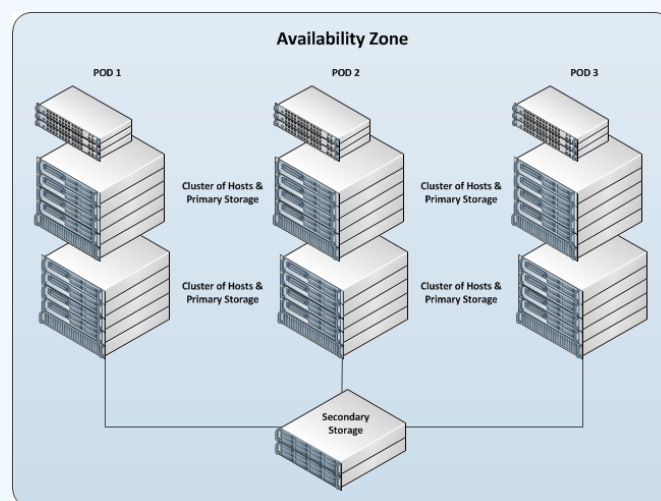


Figure 1: Example CloudStack Availability Zone



2.1.2 Zones

A CloudStack Zone (often called Availability Zone) is the largest organisational unit within a CloudStack deployment. Typically, a datacentre (DC) implementation will contain a single Zone, but there are no hard and fast rules, and a DC can contain multiple Zones. By structuring CloudStack into geographical Zones, virtual instances and data storage can be placed in specific locations to comply with an organisation's data storage policies etc.

A Zone consists of at least one Pod, and Secondary Storage which is shared by all Pods in the Zone.

Zones are visible to end users, who can then choose which Zone they wish to create their virtual Instances in.

2.1.3 Pods

Typically, a Pod relates to a discrete rack in a datacentre so that from CloudStack a whole rack/pod of hosts can be taken offline for maintenance as a group. Pods contain one or more Clusters, and a Layer 2 switch architecture which is shared by all Clusters in that Pod. End users are not aware of and have no visibility of Pods.

2.1.4 Clusters

A Cluster is a group of identical Hosts running a common Hypervisor. For example, a Cluster could be a XenServer Pool, a group of KVM Servers or a VMware cluster pre-configured in vCenter. Each Cluster has a dedicated Primary Storage array which is where the virtual machine instances are hosted.

2.1.5 Primary Storage

Primary Storage is usually unique to each Cluster (although it could also be used Zone-wide) and is used to host Instances. CloudStack is designed to work with all standards-compliant iSCSI and NFS Servers supported by the underlying Hypervisor. Special storage solutions are also supported (such as CEPH, ScaleIO, SolidFire, etc). Primary Storage is a critical component and should be built on high-performance hardware with multiple high-speed disks.

2.1.6 Secondary Storage

Secondary Storage is used to store Instance Templates, ISO images and Volume Snapshots. The storage is available to all PODs in a Zone. Secondary Storage uses the Network File System (NFS) as this ensures it can be accessed by any Host in the Zone.



2.1.7 Instances

An Instance is a virtual machine that can be created by the end-user of CloudStack. Instances are based on Instance Templates and Service Offerings which specify the size (vCPU & RAM).

3 Installation and Setup

3.1 Overview

Infrastructure-as-a-Service (IaaS) can be a complex thing to build, and by definition, it has a plethora of options, which often lead to confusion for even experienced admins who are newcomers to building cloud platforms. The goal of this guide is to provide a straightforward set of instructions to get you up and running with Apache CloudStack v4.18 with a minimum amount of trouble.

This guide can be used to build a small CloudStack test/demo cloud only, as certain networking choices have been made to get you up and running with a minimal amount of time. This guide should not be used for production setup. Its purpose is to help you get a better understanding of Apache CloudStack and learn how to use it and manage a cloud with it, before launching CloudStack for your production environment.

In this guide we will be using Rocky Linux 8.7 (as EL8 variant) and Ubuntu 22.04 LTS (as Ubuntu variant).

KVM, or Kernel-based Virtual Machine is a virtualization technology for the Linux kernel. KVM supports native virtualization atop processors with hardware virtualization extensions.

3.2 Prerequisites

To complete this guide, you'll need the following items:

- At least one computer (physical or a VM) which supports and has enabled hardware virtualization with 2 CPU, 4GB RAM.
 - If using a VM, you must ensure that Nested Virtualisation is available inside the VM
 - This server will be used as both CloudStack management server, KVM host and NFS server (for primary and secondary storage) – everything inside a single box



- Bootable media (or ISO) for EL8/Ubuntu
- A /24 network with the gateway being at (e.g.) xxx.xxx.xxx.1, no DHCP is needed on this network and none of the computers running CloudStack will have a dynamic address. Again, this is done for the sake of simplicity. If you are using a different network range, that's no problem – the idea is that we will use a small part of this network for the POD/private IPs and a small portion of the same network to serve as a “Public” network in CloudStack. No need to turn off DHCP on this network, but we will set static IPs where needed.

3.3 Environment

Before you begin, you need to prepare the environment before you install CloudStack. We will go over the steps to prepare now.

3.3.1 Operating System

In EL8:

Using the [Rocky Linux](#) 8.7 minimal x86_64 install ISO, you'll need to install the operating system on your hardware (have at least 4GB RAM, preferably 8GB or more). The defaults will generally be acceptable for this installation - but make sure to configure IP address/parameters so that you can later install needed packages from the internet (DHCP is fine at the time of installation). Later, we will change the network configuration as needed.

Once this installation is complete, you'll want to gain access to your server - through SSH.

It is always wise to update the system before starting:

```
# dnf -y update
```

To disable issues down the road, make sure to stop and disable firewall:

```
# systemctl stop firewalld; systemctl disable firewalld
```


Ubuntu 20:

First install [Ubuntu 22.04 LTS](#) on your x86_64 system that has at least 4GB RAM (preferably 8GB or more. Ensure that the “universe” repository is enabled in /etc/apt/sources.list.

Install basic packages:

```
# apt update; apt install openntpd openssh-server sudo vim htop tar net-tools
```

Allow the “root” user for SSH access using password, by editing /etc/ssh/sshd_config and ensuring there is a line “PermitRootLogin yes”. Restart ssh daemon. Change and remember the root password, test remote access using SSH for the “root” user:



```
# passwd root
```

To disable issues down the road, make sure to stop and disable firewall:

```
# systemctl stop ufw; systemctl disable ufw
```

3.3.2 Configuring the network

Throughout this document, we are assuming that you will have a /24 network for your CloudStack implementation. This can be any RFC 1918 network. However, we are assuming that you will match the server address that we are using. Thus, we may use server IP 192.168.1.2 in this guide, and because you might be on an e.g. 172.16.10.0/24 network - you would use e.g. 172.16.10.2 or similar IP.

NOTE:

If you use desktop virtualisation e.g. VirtualBox on your laptop/computer, you could have your local home network on e.g. 192.168.1.0/24 network - in this case, you can use a single free IP address from your home range for the VM. A few notes for VirtualBox:

- VirtualBox NIC for this VM should be in bridged mode for correct functioning.
- Keep in mind that there might be issues with VirtualBox networking and it might work on one's computer/laptop/OS but not on another – we've seen this during the testing of this guide – specifically, we've seen the setup works fine when VirtualBox was running on Linux laptop, with a VM's NIC bridged to a wired LAN adapter, but when testing the same on the Windows 11 OS with VM's NIC bridged to a wireless adapter, there was no network traffic happening between the host (Windows 11 laptop) and the nested VMs deployed by CloudStack (SSVM or CPVM) – rendering the setup eventually unfunctional as SSVM/CPVM could not reach Internet.
- We've also seen cases when certain CPU types were causing the nested VMs deployed by CloudStack (SSVM or CPVM) to not boot completely.
- Preferably – use a physical machine (EL8 or Ubuntu 22.04 on laptop/computer)

In EL8:

Connecting via the console or SSH, you should log in as root. We will start by creating the bridge that Cloudstack will use for networking. Create and open /etc/sysconfig/network-scripts/ifcfg-cloudbr0 and add the following settings:

```
STP=yes
```

```
TYPE=Bridge
```

```
BOOTPROTO=None
```

```
IPADDR=192.168.1.2
```



```
GATEWAY=192.168.1.1 #(this would be your physical/home router)
NETMASK=255.255.255.0
DNS1=8.8.8.8
DEFROUTE=yes
IPV6INIT=no
NAME=cloudbr0
DEVICE=cloudbr0
ONBOOT=yes
```

Save the configuration and exit. We will then edit the NIC so that it makes use of this bridge.

Open the configuration file of your NIC (e.g. /etc/sysconfig/network-scripts/ifcfg-eth0) and edit it as follows:

```
TYPE=Ethernet
BOOTPROTO=none
DEFROUTE=yes
IPV6INIT=no
NAME=eth0
DEVICE=eth0
ONBOOT=yes
BRIDGE=cloudbr0
```

Note:

Interface name (**eth0**) is used as an example only. Replace “eth0” with your default ethernet interface name.

If your physical nic has already been set up with static IP configuration before following this guide, make sure that there is no duplication between the IP configuration of cloudbr0 and eth0, which will cause a failure that would prevent the network from starting. Basically, IP configuration of eth0 should be moved to the bridge and eth0 will be added to the bridge (eth0 will have no IP configuration)



Now that we have the configuration files properly set up, we need to restart, to start up the network:

```
# reboot
```

In Ubuntu:

The same concept applies here for Ubuntu as for the EL8. Install bridge utilities:

```
# apt install bridge-utils
```

Manually create a file at `/etc/netplan/01-netcfg.yaml` applying your network-specific changes. Please note that the name of the interface in our example is “ens3” – if your interface has a different name, then make sure to reference it correctly in the configuration below:

```
network:
  version: 2
  renderer: networkd
  ethernets:
    ens3:
      dhcp4: false
      dhcp6: false
      optional: true
  bridges:
    cloudbri0:
      addresses: [192.168.1.2/24]
      gateway4: 192.168.1.1 #(this would be your physical/home router)
      nameservers:
        addresses: [8.8.8.8]
      interfaces: [ens3]
      dhcp4: false
      dhcp6: false
      parameters:
        stp: false
        forward-delay: 0
```

Note:

Interface name (**ens3**) is used as an example only. Replace “ens3” with your default ethernet interface name.



If your physical nic has already been set up with static IP configuration before following this guide, make sure that there is no duplication between the IP configuration of cloudbr0 and ens3, which will cause a failure that would prevent the network from starting. Basically, IP configuration of ens3 should be moved to the bridge and ens3 will be added to the bridge (ens3 will have no IP configuration).

Save the file and apply network config, finally reboot:

```
# netplan generate
# netplan apply
# reboot
```

3.3.3 Hostname

CloudStack requires that the hostname is properly set. If you used the default options in the installation, then your hostname is currently set to localhost.localdomain. To test this, we will run:

```
# hostname --fqdn
```

At this point it will likely return:

```
localhost
```

To rectify this situation - we'll set the hostname (we are using "server.local" in our example)

```
hostnamectl set-hostname server.local --static
```

After you've modified that file restart:

```
# reboot
```

Now recheck with the

```
# hostname --fqdn
```

and ensure that it returns a FQDN response.

3.3.4 SELinux in EL8

At the moment, for CloudStack to work properly, SELinux must be set to permissive or disabled. We want to both configure this for future boots and modify it in the current running system.

To configure SELinux to be permissive in the running system we need to run the following command:

```
# setenforce 0
```

To ensure that it remains in that state we need to configure the file /etc/selinux/config to have the following line:

```
SELINUX=permissive
```



3.3.5 Chrony

Turn on NTP for time synchronization.

Note: An NTP daemon is required to synchronize the clocks of the servers in your cloud.

Install chrony.

In EL8:

```
# dnf install chrony
```

In Ubuntu:

```
# apt install chrony
```

3.3.6 Configuring the CloudStack Package Repository

We need to configure the machine to use a CloudStack package repository.

Note: This section is using the CloudStack version 4.18, can replace the version number accordingly.

In EL8:

Import the gpg release key: (Key ID 584DF93F, Key fingerprint = 7203 0CA1 18C1 A275 68B1 37C4 BDF0 E176 584D F93F)

```
# rpm --import http://packages.shapeblue.com/release.asc
```

Add the following to your /etc/yum.repos.d/cloudstack.repo:

```
[cloudstack-4.18]
name=cloudstack
baseurl=http://packages.shapeblue.com/cloudstack/upstream/el8/4.18
enabled=1
gpgcheck=1
gpgkey=http://packages.shapeblue.com/release.asc
```

In Ubuntu:

Import the gpg release key: (Key ID 584DF93F, Key fingerprint = 7203 0CA1 18C1 A275 68B1 37C4 BDF0 E176 584D F93F)

```
# apt-key adv --keyserver keyserver.ubuntu.com --recv-keys
BDF0E176584DF93F
```

Add the following to your /etc/apt/sources.list.d/cloudstack.list file, to enable the repository:

```
deb http://packages.shapeblue.com/cloudstack/upstream/debian/4.18 /
```



3.3.7 Setup Storage

Our configuration is going to use NFS for both Primary and Secondary storage. We are going to set up two NFS shares for those purposes.

In EL8:

Install NFS server:

```
# dnf -y install nfs-utils
```

Create exports:

```
# echo "/export/primary *(rw,async,no_root_squash,no_subtree_check)" >> /etc/exports
```

```
# echo "/export/secondary *(rw,async,no_root_squash,no_subtree_check)" >> /etc/exports
```

```
# mkdir -p /export/primary /export/secondary
```

NFSv4 requires that domain setting matches on all clients. In our case, the domain is “local”, so ensure that the domain setting in /etc/idmapd.conf is uncommented and set as follows:

```
Domain = local
```

We now need to enable and start the NDS service:

```
# systemctl enable rpcbind; systemctl enable nfs-server
```

```
# systemctl start rpcbind; systemctl start nfs-server
```

In Ubuntu:

Install NFS server:

```
# apt install nfs-kernel-server quota
```

Create exports:

```
# echo "/export/primary *(rw,async,no_root_squash,no_subtree_check)" >> /etc/exports
```

```
# echo "/export/secondary *(rw,async,no_root_squash,no_subtree_check)" >> /etc/exports
```

```
# mkdir -p /export/primary /export/secondary
```

```
# exportfs -a
```



Configure and restart NFS server:

```
# sed -i -e 's/^RPCMOUNTDOPTS="--manage-gids"/RPCMOUNTDOPTS="-p 892 --manage-gids"/g' /etc/default/nfs-kernel-server

# sed -i -e 's/^STATDOPTS=/STATDOPTS="--port 662 --outgoing-port 2020"/g' /etc/default/nfs-common

# echo "NEED_STATD=yes" >> /etc/default/nfs-common

# sed -i -e 's/^RPCRQUOTADOPTS=/RPCRQUOTADOPTS="-p 875"/g' /etc/default/quota

# service nfs-kernel-server restart
```

3.4 Management Server Installation

We're going to install the CloudStack management server and surrounding tools.

3.4.1 Database Installation and Configuration

We'll start with installing MySQL and configuring some options to ensure it runs well with CloudStack.

In EL8:

```
dnf -y install mysql-server
```

With MySQL now installed we need to make a few configuration changes to `/etc/my.cnf.d/mysql-server.cnf`. Specifically, we need to add the following options to the `[mysqld]` section:

```
innodb_rollback_on_timeout=1
innodb_lock_wait_timeout=600
max_connections=350
log-bin=mysql-bin
binlog-format = 'ROW'
```

Enable and start MYSQL server daemon:

```
systemctl enable mysqld; systemctl start mysqld
```

Optionally, you can secure your MySQL installation with:

```
# mysql_secure_installation
```

In Ubuntu:

```
# apt update -y
# apt install mysql-server
```



Make a note of the MySQL server's root user password.

Configure InnoDB settings in mysql server's `/etc/mysql/mysql.conf.d/mysqld.cnf`:

```
[mysqld]
innodb_rollback_on_timeout=1
innodb_lock_wait_timeout=600
max_connections=350
log-bin=mysql-bin
binlog-format = 'ROW'
```

Restart MySQL server:

```
# systemctl restart mysql
```

3.4.2 Installation

We are now going to install the management server. We do that by executing the following command:

```
# dnf -y install cloudstack-management #(In EL8)
# apt -y install cloudstack-management #(In Ubuntu)
```

CloudStack 4.18 requires Java 11 JRE. Installing the management server will automatically install Java 11, but it's good to explicitly confirm that Java 11 is the selected/active one (in case you had a previous Java version already installed):

```
# update-alternatives --config java
```

Make sure that Java 11 is the chosen one.

With the application itself installed we can now setup the database, we'll do that with the following command and options:

```
# cloudstack-setup-databases cloud:password@localhost --deploy-
as=root:<root password, default blank> -i <cloudbr0 IP here>
```

When this process is finished, you should see a message like "CloudStack has successfully initialized database...".

Now that the database has been created, we can take the final step in setting up the management server by issuing the following command:

```
# cloudstack-setup-management
```

That concludes our setup of the management server. We still need to configure CloudStack, but we will do that after we get our hypervisor set up.



3.5 KVM Setup and Installation

3.5.1 Prerequisites

We are using the management server as a compute node as well, which means that we have already performed many of the prerequisite steps when setting up the management server, but we will list them here for clarity. Those steps are:

- Configuring the network
- Hostname
- SELinux
- Chrony
- Configuring the CloudStack Package Repository

(You don't need to do that now as we've already done that).

3.5.2 Installation

In EL8:

Installation of the KVM agent is trivial with just a single command, but afterwards, we'll need to configure a few things. We need to install the EPEL repository also.

```
# dnf -y install epel-release
# dnf -y install cloudstack-agent
```

We have two different parts of KVM to configure - libvirt and QEMU.

Enable VNC for console proxy:

```
# sed -i -e 's/\#vnc_listen.*$/vnc_listen = "0.0.0.0"/g'
/etc/libvirt/qemu.conf
```

CloudStack uses libvirt for managing virtual machines. Therefore, libvirt must be configured correctly. Libvirt is a dependency of cloud-agent and should already be installed.

Even though we are using a single host, the following steps are recommended to get familiar with the general requirements. To have live migration working libvirt has to listen for unsecured TCP connections. We also need to turn off libvirt's attempt to use Multicast DNS advertising. Both of these settings are in `/etc/libvirt/libvirtd.conf`:

```
# echo 'listen_tls=0' >> /etc/libvirt/libvirtd.conf
# echo 'listen_tcp=1' >> /etc/libvirt/libvirtd.conf
# echo 'tcp_port = "16509"' >> /etc/libvirt/libvirtd.conf
# echo 'mdns_adv = 0' >> /etc/libvirt/libvirtd.conf
# echo 'auth_tcp = "none"' >> /etc/libvirt/libvirtd.conf
```



```
# systemctl restart libvirtd
```

In Ubuntu:

Install KVM and CloudStack agent, configure libvirt:

```
# apt install qemu-kvm cloudstack-agent
```

Enable VNC for console proxy:

```
# sed -i -e 's/\#vnc_listen.*$/vnc_listen = "0.0.0.0"/g'
/etc/libvirt/qemu.conf
```

Configure default libvirtd config:

```
# echo 'listen_tls=0' >> /etc/libvirt/libvirtd.conf
# echo 'listen_tcp=1' >> /etc/libvirt/libvirtd.conf
# echo 'tcp_port = "16509"' >> /etc/libvirt/libvirtd.conf
# echo 'mdns_adv = 0' >> /etc/libvirt/libvirtd.conf
# echo 'auth_tcp = "none"' >> /etc/libvirt/libvirtd.conf
# systemctl restart libvirtd
```

Disable apparmor on libvirtd

```
# ln -s /etc/apparmor.d/usr.sbin.libvirtd /etc/apparmor.d/disable/
# ln -s /etc/apparmor.d/usr.lib.libvirt.virt-aa-helper
/etc/apparmor.d/disable/
# apparmor_parser -R /etc/apparmor.d/usr.sbin.libvirtd
# apparmor_parser -R /etc/apparmor.d/usr.lib.libvirt.virt-aa-helper
```

3.5.3 KVM configuration complete

For the sake of completeness, you should check if KVM is running OK on your machine (you should see kvm_intel or kvm_amd modules shown as loaded):

```
# lsmod | grep kvm
kvm_intel      55496 0
kvm            337772 1 kvm_intel
kvm_amd # if you are in AMD cpu
```




That concludes our installation and configuration of KVM, and we'll now move to using the CloudStack UI for the actual configuration of our cloud.

3.6 Configuration

3.6.1 UI Access


To get access to CloudStack's web interface, point your browser to the IP address of your machine e.g. `http://192.168.1.2:8080/client`


The default username is 'admin', and the default password is 'password'



apachecloudstackTM
open source cloud computing

☒ Portal login ☐ Single sign-on





4 Setting Up a Zone

The following is an example of how you can setup an advanced zone in your local network - in our case, it's 192.168.1.0/24 network.

4.1 Setup Zone

Go to Infrastructure > Zone and click on add zone button, select "Core", click on 'Next', select Advanced zone and provide the following configuration (leaving all other defaults):

Name – Zone1

IPv4 DNS 1 - 8.8.8.8

Internal DNS 1 – 8.8.8.8

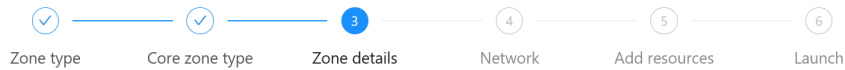
Hypervisor – KVM

The screenshot shows the 'Add zone' wizard in ShapeBlue. The wizard has six steps: Zone type, Core zone type, Zone details, Network, Add resources, and Launch. The 'Core zone type' step is currently active. Under this step, the 'Advanced' option is selected with a radio button. A description for 'Advanced' states: 'This is recommended and allows more sophisticated network topologies. This network model provides the most flexibility in defining guest networks and providing custom network offerings such as firewall, VPN, or load balancer support.' Below this, there is a toggle for 'Security groups' which is currently turned off, with a description: 'Choose this if you wish to use security groups to provide guest VM isolation.' The 'Basic' option is also visible but not selected, with a description: 'Provide a single network where each VM instance is assigned an IP directly from the network. Guest isolation can be provided through layer-3 means such as security groups (IP address source filtering).' At the bottom of the wizard, there are 'Previous' and 'Next' buttons.



Add zone ?

X



A zone is the largest organizational unit in CloudStack, and it typically corresponds to a single datacenter. Zones provide physical isolation and redundancy. A zone consists of one or more pods (each of which contains hosts and primary storage servers) and a secondary storage server which is shared by all pods in the zone.

* Name: Zone1 ✓

* IPv4 DNS1: 8.8.8.8 ✓

IPv4 DNS2:

IPv6 DNS1:

IPv6 DNS2:

* Internal DNS 1: 8.8.8.8 ✓

Internal DNS 2:

* Hypervisor: KVM ✓

Network domain:

Previous

Next

4.2 Setup Network

On the next screen (Physical network details), use the defaults – we will use the VLAN isolation method on a single physical nic (on the host) that will carry all traffic types (management, public, guest) and click “Next”.

Add zone ?

X

Zone type Core zone type Zone details **Network** Add resources Launch

Physical network Public traffic Pod Guest traffic

When adding a zone, you need to set up one or more physical networks. Each network corresponds to a NIC on the hypervisor. Each physical network can carry one or more types of traffic, with certain restrictions on how they may be combined. Add or remove one or more traffic types onto each physical network.

Network name	Isolation method	Traffic types
Physical Network 1	VLAN	<div>GUEST ✓</div> <div>MANAGEMENT ✓</div> <div>PUBLIC ✓</div> <div>+ Add traffic</div>

Add physical network

Previous Next



Public traffic configuration:

Gateway - 192.168.1.1

Netmask - 255.255.255.0

VLAN/VNI - leave blank

Start IP - 192.168.1.20

End IP - 192.168.1.50

Click on the **“Add”** button, click on **“Next”**

Add zone ?

X

✓

✓

✓

4

5

6

Zone type

Core zone type

Zone details

Network

Add resources

Launch

Physical network

Public traffic

Pod

Guest traffic

Public traffic is generated when VMs in the cloud access the internet. Publicly-accessible IPs must be allocated for this purpose. End users can use the CloudStack UI to acquire these IPs to implement NAT between their guest network and their public network.

Provide at least one range of IP addresses for internet traffic.

Gateway	Netmask	VLAN/VNI	Start IP	End IP	
<div>No Data</div>					
192.168.1.1	255.255.255.0	VLAN/VNI	192.168.1.20	192.168.1.50	Add

Previous

Next

Pod Configuration:

Name – e.g. Pod1

Reserved system gateway - 192.168.1.1

Reserved system netmask - 255.255.255.0

Start reserved system IP - 192.168.1.51

Start reserved system IP - 192.168.1.80

Click on **“Next”**



Add zone ?

X

✓

✓

✓

4

5

6

Zone type

Core zone type

Zone details

Network

Add resources

Launch

Physical network

Public traffic

Pod

Guest traffic

Each zone must contain one or more pods. We will add the first pod now. A pod contains hosts and primary storage servers, which you will add in a later step. First, configure a range of reserved IP addresses for CloudStack's internal management traffic. The reserved IP range must be unique for each zone in the cloud.

* Pod name:

Pod1

✓

* Reserved system gateway:

192.168.1.1

✓

* Reserved system netmask:

255.255.255.0

✓

* Start reserved system IP:

192.168.1.1

✓

* End reserved system IP:

192.168.1.80

✓

Previous

Next

Guest traffic:

VLAN/VNI range: 700-900

Click on "Next"

Add zone ?

X

✓

✓

✓

4

5

6

Zone type

Core zone type

Zone details

Network

Add resources

Launch

Physical network

Public traffic

Pod

Guest traffic

Guest network traffic is communication between end-user virtual machines. Specify a range of VLAN IDs or VXLAN network identifiers (VNIs) to carry guest traffic for each physical network.

VLAN/VNI range:

700

✓

-

900

✓

Previous

Next



4.3 Add Resources

Create a cluster with the following:

Name – e.g. Cluster1

Click on “Next”

Add your default/first host:

Hostname - 192.168.1.2 (or whatever IP you have set up for this machine)

Username - root

Password - <password for root user>

Click on “Next”

Add zone ? X

Zone type Core zone type Zone details Network Add resources Launch

Cluster IP address Primary storage Secondary storage

Each cluster must contain at least one host (computer) for guest VMs to run on. We will add the first host now. For a host to function in CloudStack, you must install hypervisor software on the host, assign an IP address to the host, and ensure the host is connected to the CloudStack management server.

Give the host's DNS or IP address, the user name (usually root) and password, and any labels you use to categorize hosts.

* Host name: 192.168.1.2 ✓

* Username: root ✓

Authentication Method: Password System SSH Key

* Password: ✓

Tags:

Previous Next

Add primary storage:

Name – e.g. Primary1

Scope - Zone

Protocol - NFS

Server - 192.168.1.2 (or whatever IP you have set up for this machine)

Path - /export/primary

Click on “Next”



Add zone ? X

Zone type Core zone type Zone details Network Add resources Launch

Cluster IP address Primary storage Secondary storage

Each cluster must contain one or more primary storage servers. We will add the first one now. Primary storage contains the disk volumes for all the VMs running on hosts in the cluster. Use any standards-compliant protocol that is supported by the underlying hypervisor.

* Name: Primary1 ✓

Scope: Cluster ▾

* Protocol: nfs ▾ ✓

* Server: 192.168.1.2 ✓

* Path: /export/primary ✓

* Provider: DefaultPrimary ▾

Storage tags:

Previous Next

Add secondary storage:


Provider - NFS

Name – e.g. Secondary1

Server - 192.168.1.2 (or whatever IP you have set up for this machine)

Path - /export/secondary

Click on “Next”



Add zone ? X

Zone type Core zone type Zone details Network Add resources Launch

Cluster IP address Primary storage Secondary storage

Each zone must have at least one NFS or secondary storage server. We will add the first one now. Secondary storage stores VM templates, ISO images, and VM disk volume snapshots. This server must be available to all hosts in the zone.

Provide the IP address and exported path.

Provider: NFS

Name: Secondary1

* Server: 192.168.1.2

* Path: /export/primary

Previous Next

Next, click “Launch zone” which will perform the following actions:

Create Zone

Create Physical networks:

- Add various traffic types to the physical network
- Update and enable the physical network
- Configure, enable and update various network providers and elements such as the virtual network element

Create Pod

Configure public traffic

Configure guest traffic (VLAN range for physical network)

Create Cluster

Add host

Create primary storage (also mount it on the KVM host)

Create secondary storage

Complete zone creation



Add zone ?

×

✓

✓

✓

✓

✓

6

Zone type

Core zone type

Zone details

Network

Add resources

Launch

✓

 Zone is ready to launch; please proceed to the next step.

Previous

Launch zone

Finally, confirm and enable the zone. Wait for the system VMs to come up, and then you can proceed with your IaaS usage. Happy hacking!

Add zone ?

×

✓

✓

✓

✓

✓

6

Zone type

Core zone type

Zone details

Network

Add resources

Launch

Please wait while your zone is being created; this may take a while...

✓

 Configuring public traffic

✓

 Configuring guest traffic

✓

 Creating cluster

✓

 Adding host

✓

 Creating primary storage

✓

 Creating secondary storage

✓

 Zone creation complete.

Enable zone



Accounts

Domains

Infrastructure

Summary

Zones

Pods

Clusters

Hosts

Primary storage

Secondary storage

System VMs

Default view

/ System VMs Refresh

<input type="checkbox"/>	Name	State	Agent state	System VM type
<input type="checkbox"/>	v-1-VM	Running	Up	consoleproxy
<input type="checkbox"/>	s-2-VM	Running	Up	secondarystoragevm

Showing 1-2 of 2 items < 1 > 20 / page



5 Conclusion

CloudStack is a very powerful, all-in-one, solution for IaaS platform that provides a seamless cloud computing experience. It is hypervisor-agnostic and works with different storage types to fit different Private, Public, or Hybrid cloud purposes.

CloudStack installation and upgrade processes are relatively simple, especially when compared to other cloud orchestrators, and it takes no special skills to keep it running smoothly. It's used by some of the largest ISPs across USA, Japan, Europe and Asia, while at the same time it's also used by so many small start-ups. As an extremely versatile, scalable, and performant cloud orchestrator, it can be used in, virtually, any cloud scenario. Being an open-source product assures no vendor lock-in, zero price tag, while at the same time allowing for the possibility to be easily extended and improved as needed.

CloudStack can manage tens of thousands of physical servers installed in geographically distributed datacenters. The management server scales near-linearly eliminating the need for cluster-level management servers. Maintenance or other outages of the management server can occur without affecting the virtual machines running in the cloud.

Apache CloudStack is a full-featured IaaS platform supporting a wide range of integrations. Its global community constantly develops new features and supports new technologies with a clearly defined, evolving roadmap guided by users and the community. There are no different levels of support or versions, and although there are vendor distributions available, no vendor has a dominant influence over the project and most organizations run the freely available, open-source version in production.

Being open-source ensures that CloudStack follows the needs of its user and enables them to build future-proof technology solutions.

About Apache CloudStack

Apache CloudStack is the leading open source cloud orchestration platform, in use by many of the worlds largest public and private clouds. It is a multi-hypervisor, multi-tenant, high-availability Infrastructure as a Service cloud management platform. CloudStack is software that provides a cloud orchestration layer, giving automation of the creation, provisioning and configuration of IaaS components.

CloudStack turns an existing virtual infrastructure into a cloud-based infrastructure as a Service (IaaS) platform. The fact CloudStack leverages existing infrastructure means that the cost and time for an organisation to build a multi-tenant IaaS platform is greatly reduced.



cloudstack.apache.org

About ShapeBlue

ShapeBlue is the largest independent integrator of CloudStack technologies globally and are specialists in the design and implementation of IaaS cloud infrastructures for both private and public cloud implementations. We combine 100's of person-years of experience in designing and building complex network, storage and compute infrastructures with globally leading skills in Apache CloudStack.



www.shapeblue.com | info@shapeblue.com

"Apache", "CloudStack", "Apache CloudStack", the Apache CloudStack logo, the Apache CloudStack Cloud Monkey logo and the Apache feather logos are registered trademarks or trademarks of The Apache Software Foundation.

Need help with Apache CloudStack?

Get In Touch