

Exercise 1: Install Puppet Master and Agents and run basic commands

- Install Puppet Master & Agent on AWS
- **Launch And Configure EC2-Instance in console**

Resources

You are using the following Amazon EC2 resources in the Asia Pacific (Mumbai) region:

0 Running Instances	0 Elastic IPs
2 Volumes	0 Snapshots
1 Key Pairs	0 Load Balancers
0 Placement Groups	2 Security Groups

Build and run distributed, fault-tolerant applications in the cloud with [Amazon Simple Workflow Service](#).

Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

[Launch Instance](#)

Note: Your instances will launch in the Asia Pacific (Mumbai) region

Service Health [Scheduled Events](#)

- **Select Ubuntu OS 14.04**

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Tag Instance 6. Configure Security Group 7. Review

Step 1: Choose an Amazon Machine Image (AMI)

Cancel and Exit





My AMIs

AWS Marketplace

Community AMIs

Operating system

- ☐ Amazon Linux
- ☐ Cent OS
- ☐ Debian
- ☐ Fedora
- ☐ Gentoo
- ☐ OpenSUSE
- ☐ Other Linux
- ☐ Red Hat
- ☐ SUSE Linux
- ☒ Ubuntu
- ☐ Windows

	ubuntu/images/hvm-instance/ubuntu-trusty-14.04-amd64-server-20160516 - ami-e6701a89	Select
	Root device type: instance-store Virtualization type: hvm	64-bit
	ubuntu/images/hvm-io1/ubuntu-precise-12.04-amd64-server-20160509 - ami-1372187c	Select
	Canonical, Ubuntu, 12.04 LTS, amd64 precise image build on 2016-05-09	64-bit
	Root device type: ebs Virtualization type: hvm	
	ubuntu/images/hvm-io1/ubuntu-trusty-14.04-amd64-server-20151008 - ami-19721876	Select
	Canonical, Ubuntu, 14.04 LTS, amd64 trusty image build on 2015-10-08	64-bit
	Root device type: ebs Virtualization type: hvm	
	ubuntu/images/hvm-io1/ubuntu-trusty-14.04-amd64-server-20150810 - ami-1b721874	Select
	Canonical, Ubuntu, 14.04 LTS, amd64 trusty image build on 2015-08-10	64-bit
	Root device type: ebs Virtualization type: hvm	

- **Choose the server type**

<input checked="" type="checkbox"/>	General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.small	1	2	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.medium	2	4	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	t2.large	2	8	EBS only	-	Low to Moderate
<input type="checkbox"/>	General purpose	m4.large	2	8	EBS only	Yes	Moderate

[Cancel](#)
[Previous](#)
[Review and Launch](#)
[Next: Configure Instance Details](#)

- Enter the number of servers we need to create :-I enter 2 server it will automatically create Add the Disk

[Launch into Auto Scaling Group](#)

You may want to consider launching these instances into an Auto Scaling Group in the future. [Learn how Auto Scaling can help your application stay healthy](#)

- Storage memory as per requirement**

Root	/dev/sda1	snap-78f54a91	8	Provisioned IOPS SSD (IO1)	200	N/A	<input checked="" type="checkbox"/>	Not Encrypted
------	-----------	---------------	---	----------------------------	-----	-----	-------------------------------------	---------------

[Add New Volume](#)

General Purpose (SSD) volumes provide the ability to burst to 3000 IOPS per volume, independent of volume size, to meet the performance needs of most applications and also deliver a consistent baseline of 3 IOPS/GiB. [Set my root volume to General Purpose \(SSD\).](#)

- Configure the Security Group to enable the HTTP,TCP**

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group: ☒ Create a new security group
☐ Select an existing security group

Security group name:
 Description:

Type	Protocol	Port Range	Source
SSH	TCP	22	Anywhere 0.0.0.0/0
Custom TCP Rule	TCP	8140	Anywhere 0.0.0.0/0
All traffic	All	0 - 65535	Anywhere 0.0.0.0/0
HTTPS	TCP	443	Anywhere 0.0.0.0/0

[Add Rule](#)

- Launch instances and connect using Putty
- Install Puppetmaster and Agent**

```

Sudo -i

Hostname

hostname puppetmaster.test.org

exit

sudo -i
  
```

- Check ifconfig of agent machines and copy the IP address for agent. Run the following command on master machine

```
echo 172.31.26.143 puppetagent1.test.org >> /etc/hosts

ping puppetagent1.test.org

apt-get update

apt-get install puppetmaster

puppet --version

service puppetmaster status

puppet cert list -all
```

- On Agent node, open terminal

```
Sudo -i

hostname

hostname puppetagent1.test.org

exit

sudo -i
```

- On master node, open terminal and run ifconfig to check the IP address

```
Ifconfig
```

- On agent node, open terminal and run following commands

```
echo 172.31.22.177 puppetmaster.test.org >> /etc/hosts

ping puppetmaster.test.org

apt-get update

apt-get install puppet

Service puppet status

vi /etc/puppet/puppet.conf

server=puppetmaster.test.org

Puppet agent -t

Puppet agent --enable

Puppet agent --server puppetmaster.test.org

Puppet agent -t
```

- Run the following commands in agent terminal window

```
puppet agent -t

facter
```

- Run the following commands in master terminal window

```
puppet apply -e 'notify { "Hello World" : }'

puppet apply -e 'service {"sshd" : ensure =>"stopped",
enable => false,}'

service ssh restart
```

- Run the following commands in master terminal window

```
cd /etc/puppet
ls
cd manifests
```

- Create a file by typing nano site.pp and edit the following code. Save and exit

```
node default {
    notify{"The default puppet configuration": }
}
```

- Go to Agent window and run the following command

```
puppet agent -t
```

- Again edit the site.pp by typing nano site.pp. Add the following code, save and exit.

```
node puppetagent1 {
    notify{"We have matched nodes in Puppet": }
}
```

- Go to Agent window and run the following command

```
puppet agent -t
```

- Again edit the site.pp by typing nano site.pp. Update the previous code with the following code, save and exit. This will match all the nodes that have the words "puppet" in their name.

```
node /^puppet/ {
    notify{"We have matched nodes in Puppet": }
}
```

- Go to Agent window and run the following command

```
puppet agent -t
```

- Go to puppetagent1 client terminal and run the following commands

```
cd /var/lib/puppet
ls -l
cd client_data/catalog
ls -l
more puppetagent1.test.org.json
```