**KubeNow**

**What is KubeNow?**

It is a collection/stack of tools that is developed to help provision Kubernetes on Cloud Platforms. It was developed as a part of the PhenoMeNal Project (Phenome and Metabolome aNalysis) – Large Scale Computing for Medical Metabolomics. It was developed at Uppsala University, Sweden.

**Running KubeNow on AWS**

There are 3 tools that you need to install on your local machine, in order to provision Kubernetes with KubeNow:

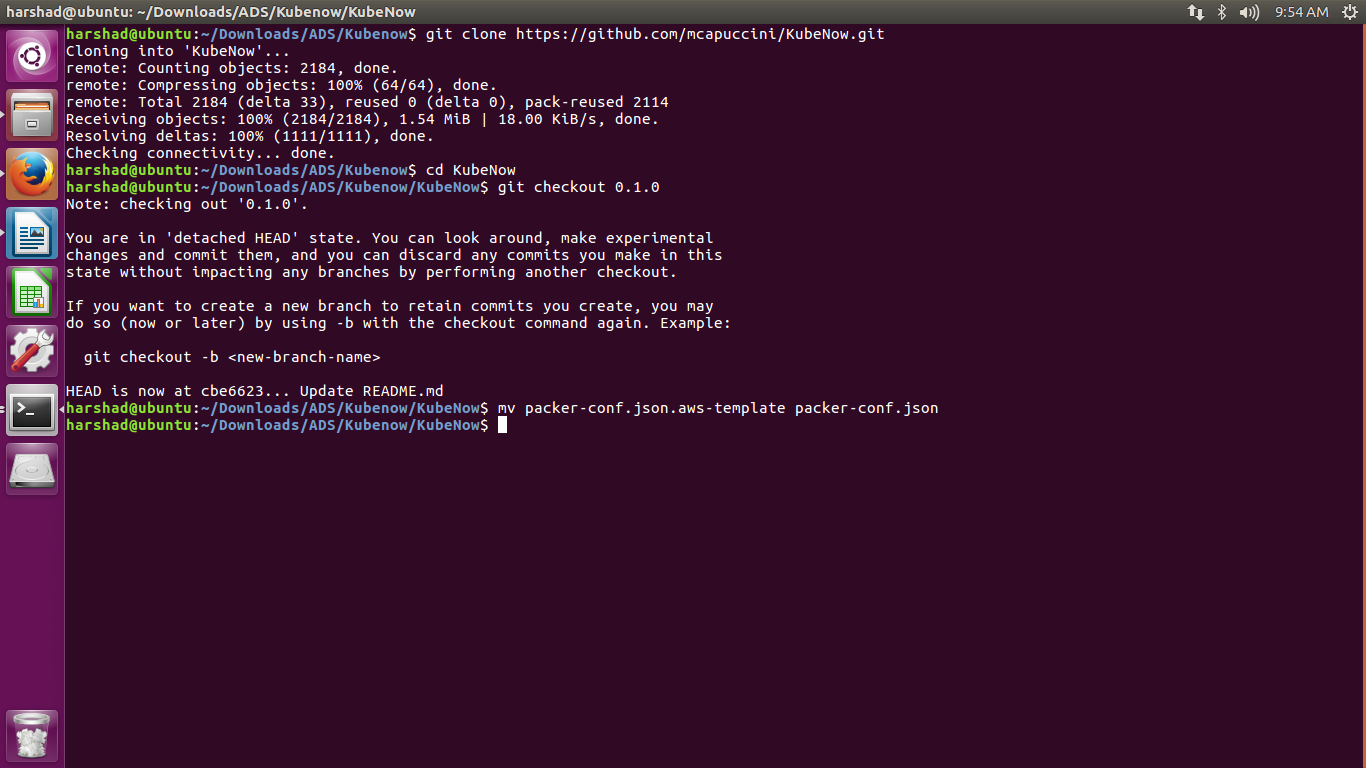
1. Packer (0.10.2 or higher) to build a KubeNow cloud image on the host cloud
2. Terraform (0.7.8 or higher) to fire-up the virtual infrastructure on the host cloud
3. Ansible (2.1.2.0 or higher) to provision the VMs (e.g. install and configure networking, reverse proxy etc.)

**Clone the pre-configured KubeNow from git:**

git clone <https://github.com/mcapuccini/KubeNow.git>

cd KubeNow

git checkout 0.1.0



**Prerequisites: Installing Provisioning tools**

Install **packer:** <https://www.packer.io/downloads.html>

Install **terraform**: <https://www.terraform.io/downloads.html>

Install **ansible:** <https://www.ansible.com/tower-trial>

**Build the KubeNow image**

The first time you are going to deploy KubeNow, you’ll have to create its cloud image. This considerably speeds up the following bootstraps, as all of the required software will already be installed on the instances. To do this we need to configure packer using packer-conf.json.

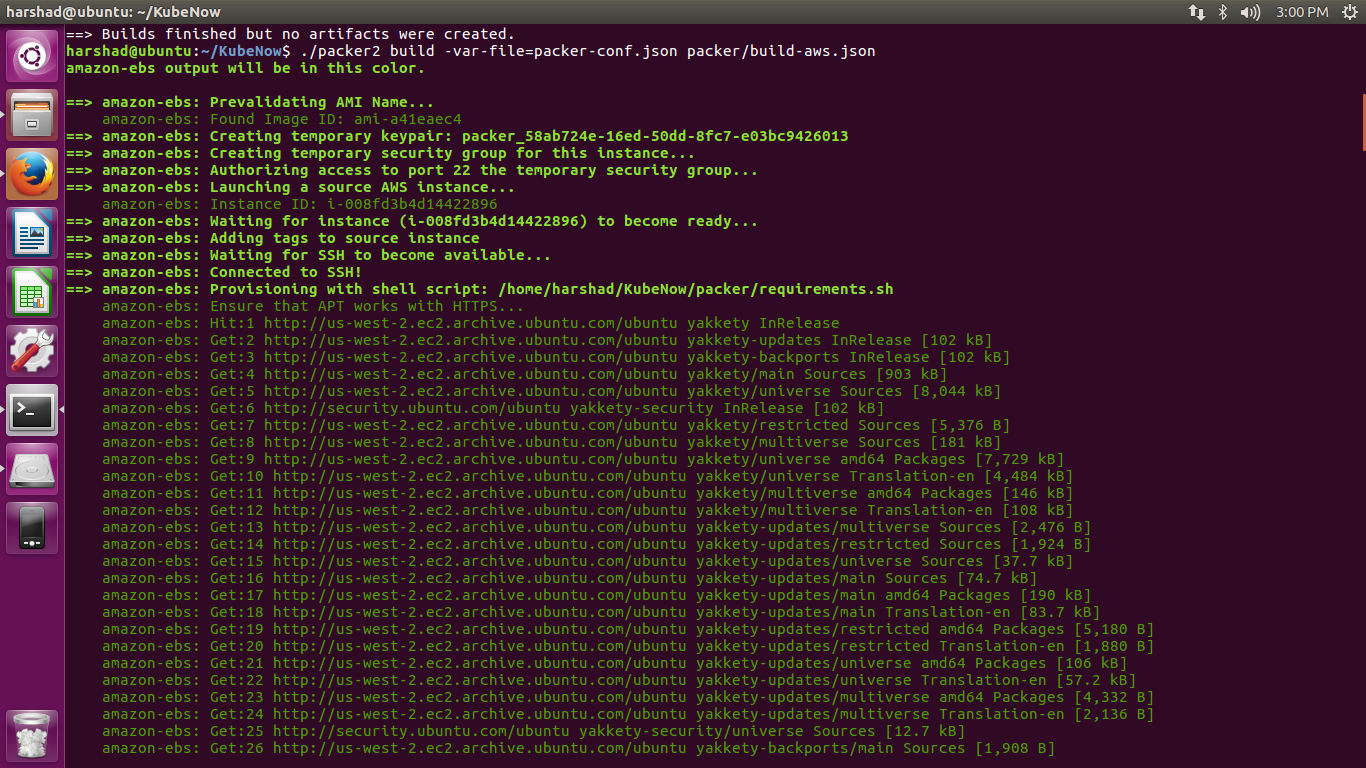
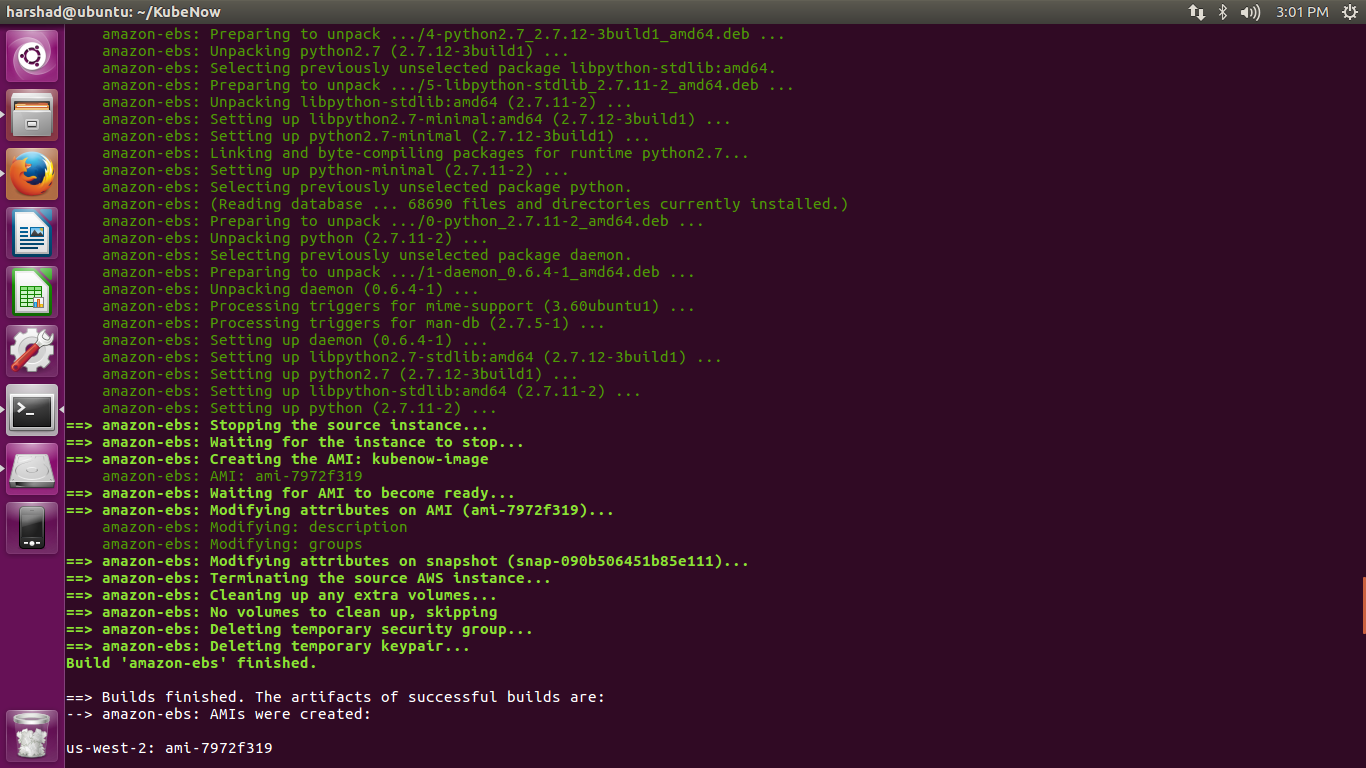
You can use the already provided template

mv packer-conf.json.aws-template packer-conf.json

and set the following parameters:

* image\_name: the name of the image that will be created after the build (e.g. “kubenow-image”). The image\_name must be unique in AWS, otherwise it will fail creating the new image
* source\_image\_id: an Ubuntu Xenial AMI ID. To figure out an Ubuntu Xenial AMI ID that works with your preferred region, you can use the [Amazon EC2 AMI Locator](https://cloud-images.ubuntu.com/locator/ec2/) only hvm:ebs-ssd AMIs are supported (other AMIs might work anyway)
* aws\_access\_key\_id: your access key id
* aws\_secret\_access\_key: your secret access key
* region: the region to use in order to create the image

Once you are done with your settings you are ready to build KubeNow using Packer:

packer build -var-file=packer-conf.json packer/build-aws.json

If everything goes well, something like the following will be printed out:

**==> Builds finished. The artifacts of successful builds are:**

**--> amazon-ebs: AMIs were created:**

**eu-central-1: ami-XXXX**

To check the image is uploaded you can check the aws console or run the following command:

aws ec2 describe-images --owners self

**Bootstrap Kubernetes**

Now we are going to provision the required virtual infrastructure in AWS (Amazon Web Services) using Terraform. This procedure will inject enough information in each instance, to independently provision itself. To do this we need to configure terraform using terraform.tfvars.

You can use the already provided template

mv terraform.tfvars.aws-template terraform.tfvars

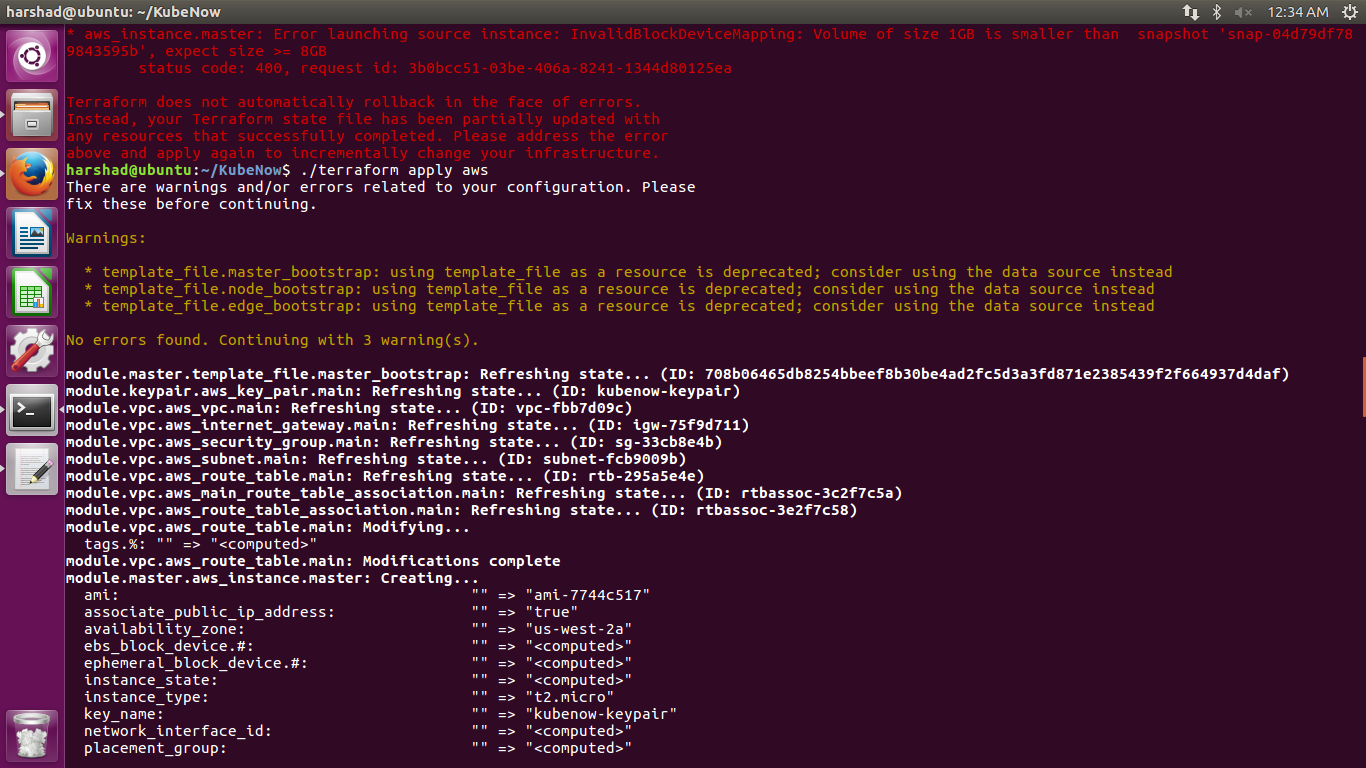
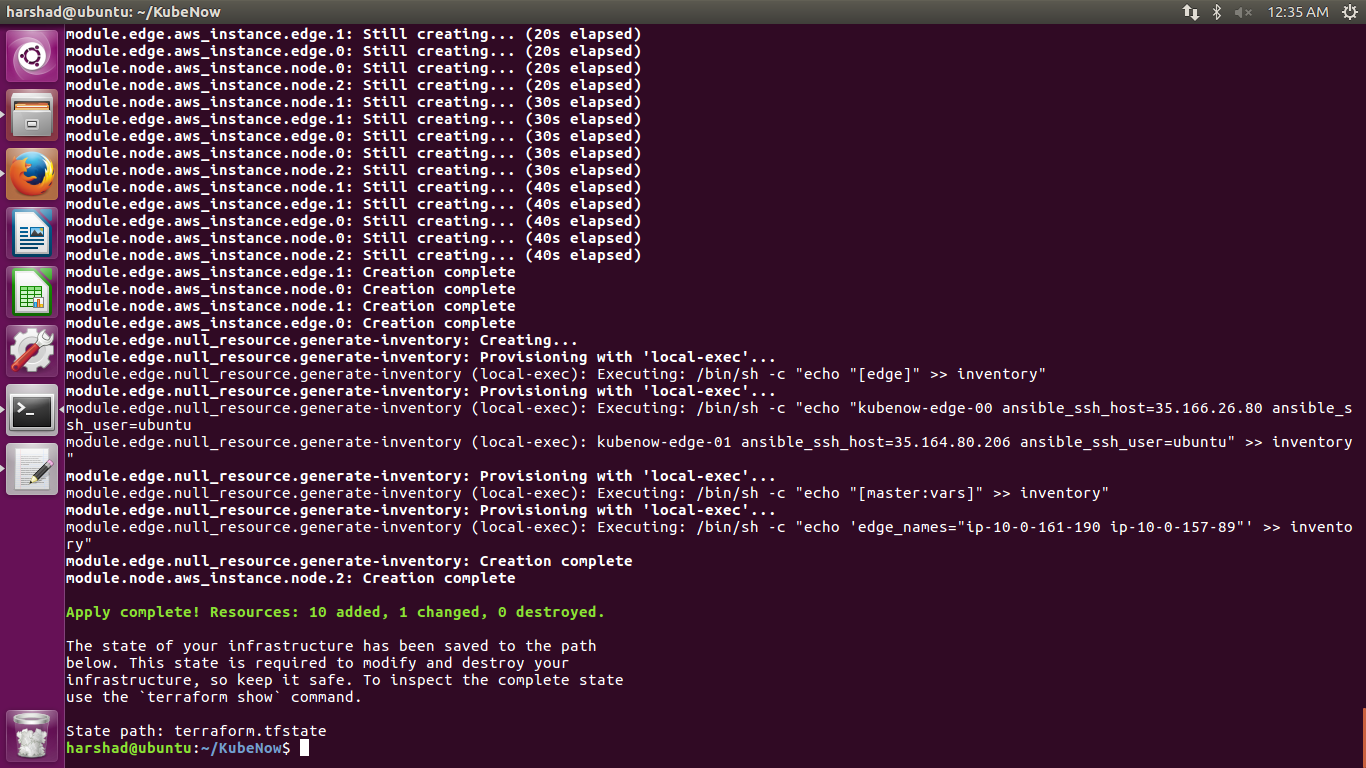
and set the following parameters:

* **Cluser configuration**
  + **cluster\_prefix**: every resource in your tenancy will be named with this prefix
  + **kubenow\_image\_id**: ID of the AMI that you previously created using packer
  + **kubeadm\_token**: a token that will be used by kubeadm, to bootstrap Kubernetes. You can run *generate\_kubetoken.sh* to create a valid one.
  + **ssh\_key**: path to your public ssh-key to be used for ssh node access (e.g. ~/.ssh/id\_rsa.pub)
  + **aws\_region**: the region where your cluster will be bootstrapped (e.g. eu-west-1)
  + **availability\_zone**: an availability zone for your cluster (e.g. eu-west-1a)
* **Credentials**
  + **aws\_access\_key\_id**: your access key id
  + **aws\_secret\_access\_key**: your secret access key
* **Master configuration**
  + **master\_instance\_type**: an instance type for the master (e.g. t2.micro)
  + **master\_disk\_size**: edges disk size in GB
* **Node configuration**
  + **node\_count**: number of Kubernetes nodes to be created
  + **node\_instance\_type**: an instance type for the Kubernetes nodes (e.g. t2.micro)
  + **node\_disk\_size**: edges disk size in GB
* **Edge configuration**
  + **edge\_count**: number of egde nodes to be created
  + **edge\_instance\_type**: an instance type for the edge nodes (e.g. t2.micro)
  + **edge\_disk\_size**: edges disk size in GB

Once done, bootstrap the cluster using terraform

terraform get aws

terraform apply aws



If everything goes well, something like the following message will be printed:

**Apply complete! Resources: X added, Y changed, Z destroyed.**

This will create and launch all the instance of nodes, edges and masters on AWS.

References:

<http://kubenow.readthedocs.io/en/stable>

<https://github.com/kubenow/KubeNow>