# Fuel deploy on Ubuntu devnet Sandbox – Brahmaputra release

### What is Sandbox?

Sandbox is a DevNet service providing developers access to hosted Cisco technology labs. With Sandbox, developers can schedule and get access to working Cisco technology labs via an online public facing web interface. These labs are available 24X7 across the globe and available via VPN. As a DevNet member we will able to develop and test new or existing applications with Cisco developer enabled technology. Over the web users will log in, schedule a lab of their choice, specify a length of time, and the rest is automatic.

Lab guide of OPNFV Sandbox reservation can be found here:

<https://developer.cisco.com/site/devnet/sandbox/documents/guide/>

### Introduction

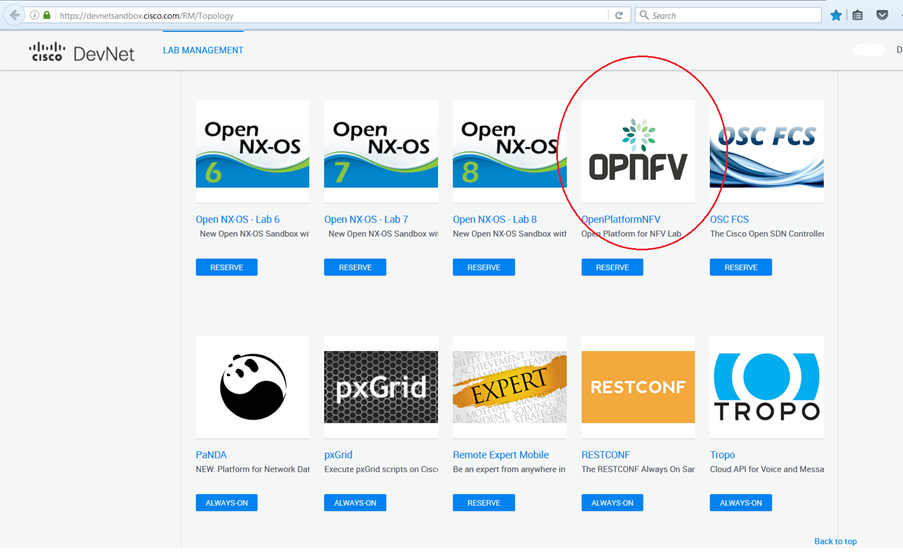
The objective of this setup is to try out OPNFV on virtual environment and basic VNF on different VMs launched in openstack. However the Sandbox can be used to try out more of its features (i.e. O/S, ODL L2, etc.). For the setup we use Fuel as an installer. As the compute nodes are going to be virtual machines they do not support hardware virtualization therefore we are going to use qemu, instead of KVM, emulator.

### Sandbox setup

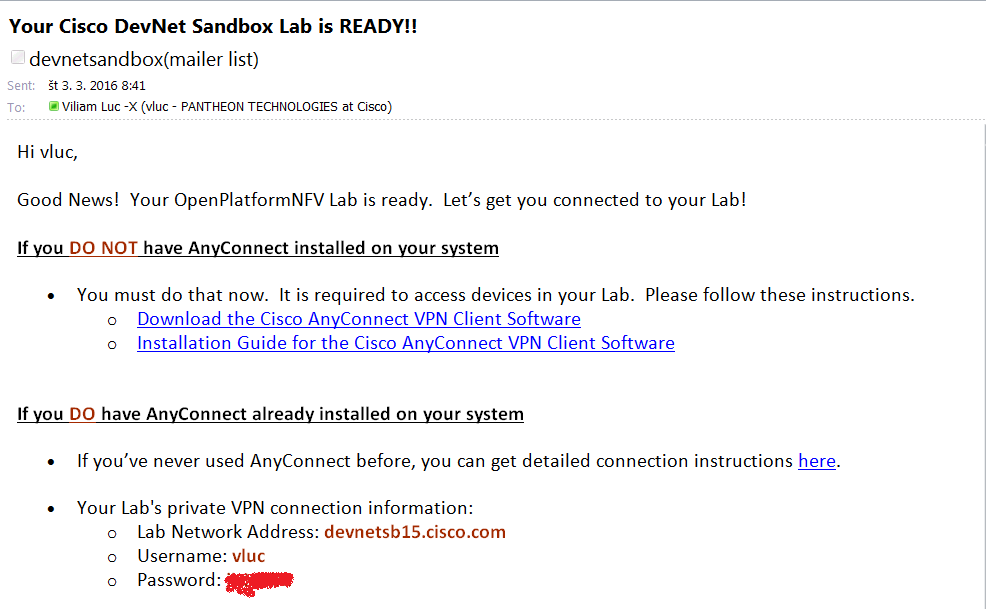
For the access to the OPNFV devnet sandbox contact Eddie Corban ([ecorban@cisco.com](mailto:ecorban@cisco.com)).

At first we need to do the reservation in devnet sandbox: <https://developer.cisco.com/site/devnet/sandbox/>

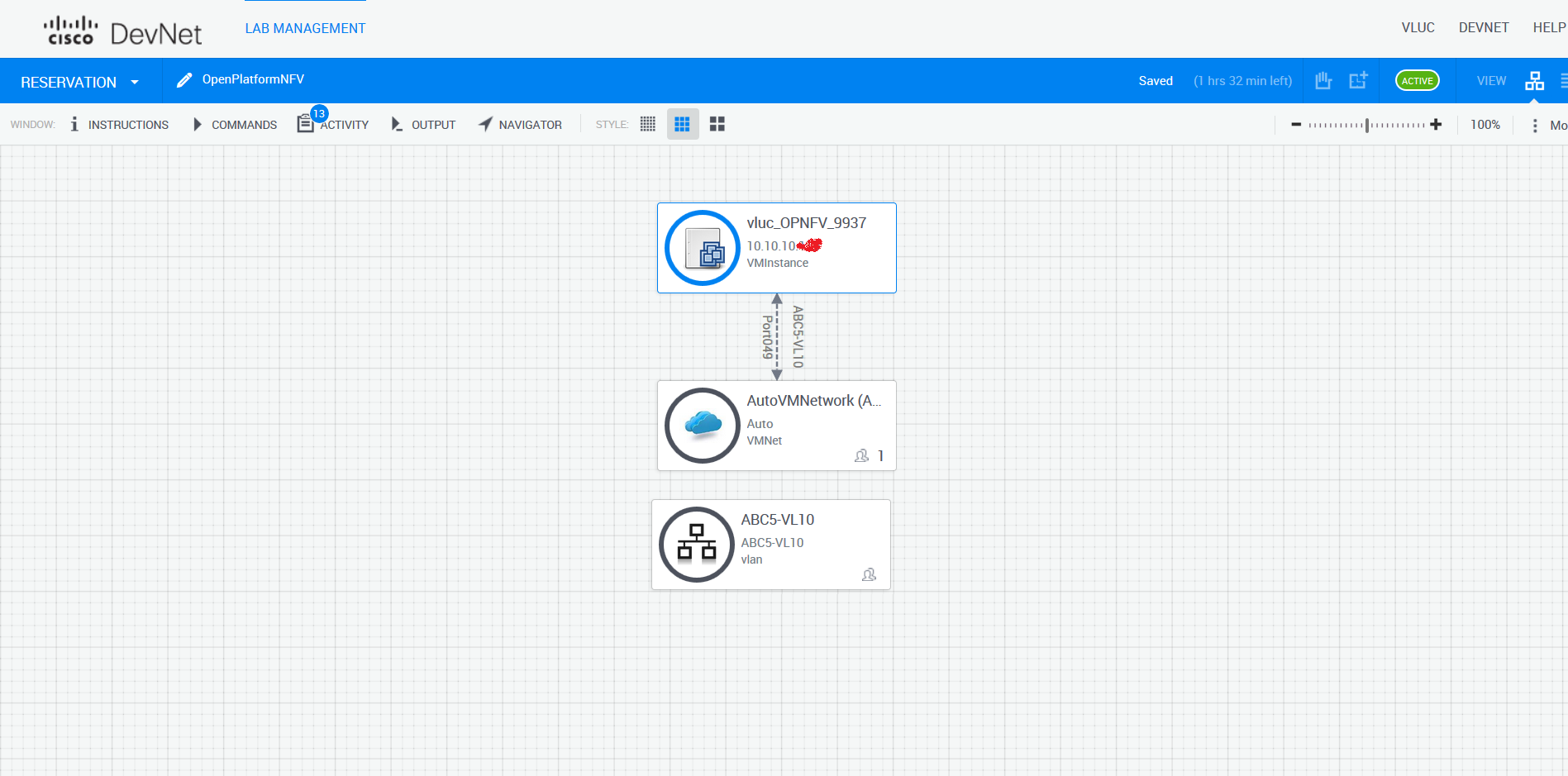
Click on Get Started and when we get in DevNet zone find OpenPlatformNFV (OPNFV) box and click RESERVE button.



After 25-45 minutes it should be ready. We will receive an email notification about starting and ending the setup. In this email we will also receive a VPN credentials to connect to the sandbox. It should look like this:



When the setup is ready e should see the screen similar to the picture below with the IP address of OPNFV instance.

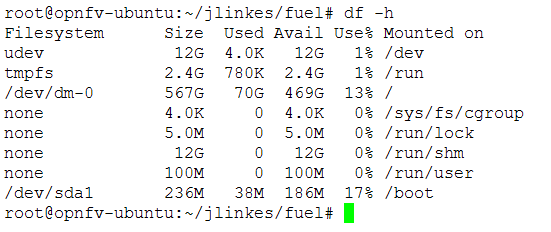


Then we can connect to the VPN and use putty or any other ssh client to connect to the sandbox. To connect to the VPN follow instructions received in the email. When connected to the VPN, we should be able to connect to the VM instance which IP start with 10.10.10.xxx. OPNFV is not deployed on this VM instance so next step is to deploy OPNFV on it.

### OPNFV deploy using Fuel

Now when connected to the 10.10.10.xxx VM we can start deploying OPNFV. Before starting deployment we should check if there is enough space left for fuel, controller and 3 compute nodes. Fuel takes 100GB, controller 100GB, and one compute node 100GB of space. We will need at least 500GB of free disk space. Running command df should show us how much space we got:

*# df*

**

If there is not enough space some VMs might be running from previous deployment, so we can delete them. Use following commands to see if some VMs are running or created (but not running).

*# virsh list*

*# virsh list --all*

If some VMs are running from previous deployment delete them by using following commands. That should also clear space on disk.

*# virsh destroy*

*# virsh undefine*

When the VMs are destroyed and undefined look at ***/var/lib/libvirt/images.*** And see whether there are some images left and delete them (should clear space on disk too)

These dependencies and modules are required before deployment

*# sudo apt-get install -y libvirt-bin qemu-kvm python-pip fuseiso mkisofs genisoimage  
# sudo apt-get install -y python-dev libz-dev libxml2-dev libxslt-dev  
# sudo pip install pyyaml netaddr paramiko lxml scp pycrypto ecdsa*During libvirt install the user is added to the libvirtd group, so you have to  
logout then login back again

Now let’s start deployment. This may take ~2 hours.

Remember that the project is under active development so it is possible that some steps might change during time.

1. I recommend to create our own directory inside root home:

*# sudo su -*

*# mkdir user1  
# cd user1*

1. Clone deploy scripts into that directory.  
   *# git clone https://gerrit.opnfv.org/gerrit/fuel -b stable/brahmaputra*

Deploy script can be found in fuel/deploy directory.

*# cd fuel/deploy/*

1. a) In the /home/opnfvuser directory find *opnfv-2016-03-02\_07-59-55.iso* image file which will be used later for deploy.

b) Or download fuel iso image. The latest ISO image can be found on <http://artifacts.opnfv.org> (search for fuel). The name should something similar to this: [*fuel/opnfv-2016-03-02\_07-59-55.iso*](http://artifacts.opnfv.org/fuel/opnfv-2016-03-02_07-59-55.iso)

*# wget* [*http://artifacts.opnfv.org/fuel/opnfv-2016-03-02\_07-59-55.iso*](http://artifacts.opnfv.org/fuel/opnfv-2016-03-02_07-59-55.iso)

1. a) Now we need to update few config files (dea.yaml, dha.yaml, controller.xml, compute.xml and fuel.xml) to get it working on Brahmaputra release. Steps how to update files are in sections 4.1, 4.2, 4.3, 4.4 and 4.5

b) Or we can use pre-configured files and simply copy-paste them on the specified location. All these files can be pulled from git repository: [*https://github.com/CiscoDevNet/OPNFV.git*](https://github.com/CiscoDevNet/OPNFV.git)

After pulling from repository configuration files can be found in *ubuntu/brahmaputra\_fuel\_conf\_files* directory.

* 1. **Dea.yaml**

*# vim template/virtual\_environment/conf/dea.yaml*

* + 1. Change release to

*wanted\_release: Liberty on Ubuntu 14.04*

* + 1. Change number of controllers and compute nodes. I use 1 controller and 3 compute nodes. Note: As far as I know deployment need at least 3 ceph-osd on compute nodes, so at least 3 compute nodes are required

*- id: 1*

*interfaces: interfaces\_1*

*transformations: transformations\_1*

*role: ceph-osd,controller*

*- id: 2*

*interfaces: interfaces\_1*

*transformations: transformations\_2*

*role: ceph-osd,compute*

*- id: 3*

*interfaces: interfaces\_1*

*transformations: transformations\_2*

*role: ceph-osd,compute*

*- id: 4*

*interfaces: interfaces\_1*

*transformations: transformations\_2*

*role: ceph-osd,compute*

* + 1. Change NTP servers. Set NTP2 to null and NTP1 to 172.16.0.1. Also change external\_ntp value to 10.20.0.2

*...*

*NTP1: 172.16.0.1*

*NTP2: null*

*NTP3: null*

*...*

*external\_ntp:*

*metadata:*

*label: Host OS NTP Servers*

*weight: 100*

*ntp\_list:*

*description: List of upstream NTP servers, separated by comma*

*label: NTP server list*

*regex:*

*error: Invalid NTP server list*

*source: ^\s\*(?:(?:\w+(?:-+\w+)\*\.)+[a-z]+|\d{1,3}(?:\.\d{1,3}){3})\s\*(?:,\s\*(?:(?:\w+(?:-+\w+)\*\.)+[a-z]+|\d{1,3}(\.\d{1,3}){3})\s\*)\*$*

*type: text*

*value: 10.20.0.2*

*...*

* + 1. Change ethernet names from eth to ens. Eg eth0 ->ens3, eth1->ens4, eth2->ens5, eth3->ens6

*transformations\_1:*

*transformations:*

*- action: add-br*

*name: br-fw-admin*

*- action: add-br*

*name: br-mgmt*

*- action: add-br*

*name: br-storage*

*- action: add-br*

*name: br-ex*

*- action: add-br*

*name: br-floating*

*provider: ovs*

*- action: add-patch*

*bridges:*

*- br-floating*

*- br-ex*

*mtu: 65000*

*provider: ovs*

*- action: add-br*

*name: br-mesh*

*- action: add-port*

*bridge: br-fw-admin*

*name: ens3*

*- action: add-port*

*bridge: br-mgmt*

*name: ens3.101*

*- action: add-port*

*bridge: br-storage*

*name: ens4.102*

*- action: add-port*

*bridge: br-mesh*

*name: ens5.103*

*- action: add-port*

*bridge: br-ex*

*name: ens6*

*transformations\_2:*

*transformations:*

*- action: add-br*

*name: br-fw-admin*

*- action: add-br*

*name: br-mgmt*

*- action: add-br*

*name: br-storage*

*- action: add-br*

*name: br-mesh*

*- action: add-port*

*bridge: br-fw-admin*

*name: ens3*

*- action: add-port*

*bridge: br-mgmt*

*name: ens3.101*

*- action: add-port*

*bridge: br-storage*

*name: ens4.102*

*- action: add-port*

*bridge: br-mesh*

*name: ens5.103*

* + 1. Remove mongo in dea.yaml

*...*

*mongo:*

*description: If selected, You can use external Mongo DB as ceilometer backend*

*label: Use external Mongo DB*

*restrictions:*

*- settings:additional\_components.ceilometer.value == false*

*type: checkbox*

*value: false*

*weight: 40*

...

*external\_mongo:*

*hosts\_ip:*

*description: IP Addresses of MongoDB. Use comma to split IPs*

*label: MongoDB hosts IP*

*regex:*

*error: Invalid hosts ip sequence*

*source: ^(((25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?),)\*((25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}(25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)$*

*type: text*

*value: ''*

*weight: 30*

*metadata:*

*label: External MongoDB*

*restrictions:*

*- action: hide*

*condition: settings:additional\_components.mongo.value == false*

*message: Ceilometer and MongoDB are not enabled on the Additional Components*

*section*

*weight: 20*

*mongo\_db\_name:*

*description: Mongo database name*

*label: Database name*

*regex:*

*error: Invalid database name*

*source: ^\w+$*

*type: text*

*value: ceilometer*

*weight: 30*

*mongo\_password:*

*description: Mongo database password*

*label: Password*

*regex:*

*error: Password contains spaces*

*source: ^\S\*$*

*type: password*

*value: ceilometer*

*weight: 30*

*mongo\_replset:*

*description: Name for Mongo replication set*

*label: Replset*

*type: text*

*value: ''*

*weight: 30*

*mongo\_user:*

*description: Mongo database username*

*label: Username*

*regex:*

*error: Empty username*

*source: ^\w+$*

*type: text*

*value: ceilometer*

*weight: 30*

* + 1. Change virtualization type to qemu-kvm

*libvirt\_type:*

*label: Hypervisor type*

*type: radio*

*value: qemu*

* + 1. Set repositories

*- name: ubuntu*

*priority: null*

*section: main universe multiverse*

*suite: trusty*

*type: deb*

*uri: http://10.20.0.2:8080/mirrors/ubuntu/*

*- name: ubuntu-updates*

*priority: null*

*section: main universe multiverse*

*suite: trusty-updates*

*type: deb*

*uri: http://10.20.0.2:8080/mirrors/ubuntu/*

*- name: ubuntu-security*

*priority: null*

*section: main universe multiverse*

*suite: trusty-security*

*type: deb*

*uri: http://10.20.0.2:8080/mirrors/ubuntu/*

*- name: mos*

*priority: 1050*

*section: main restricted*

*suite: mos8.0*

*type: deb*

*uri: http://10.20.0.2:8080/liberty-8.0/ubuntu/x86\_64*

*- name: Auxiliary*

*priority: 1150*

*section: main restricted*

*suite: auxiliary*

*type: deb*

*uri: http://10.20.0.2:8080/liberty-8.0/ubuntu/auxiliary*

* 1. **Dha.yaml**

*# vim template/virtual\_environment/conf/dha.yaml*

* + 1. Change number of controllers and compute nodes as it is in dea.yaml

*nodes:*

*- id: 1*

*libvirtName: controller1*

*libvirtTemplate: templates/virtual\_environment/vms/controller.xml*

*- id: 2*

*libvirtName: compute1*

*libvirtTemplate: templates/virtual\_environment/vms/compute.xml*

*- id: 3*

*libvirtName: compute2*

*libvirtTemplate: templates/virtual\_environment/vms/compute.xml*

*- id: 4*

*libvirtName: compute3*

*libvirtTemplate: templates/virtual\_environment/vms/compute.xml*

*- id: 5*

*libvirtName: fuel-master*

*libvirtTemplate: templates/virtual\_environment/vms/fuel.xml*

*isFuel: yes*

*username: root*

*password: r00tme*

* + 1. Change disk sizes for fuel, controller and compute to 100GB

*disks:*

*fuel: 100G*

*controller: 100G*

*compute: 100G*

* 1. **Controller.xml**

*# vim template/virtual\_environment/vms/controller.xml*

* + 1. Change machine-type

*<os>*

*<type arch='x86\_64' machine='pc'>hvm</type>*

* 1. **Compute.xml**

*# vim template/virtual\_environment/vms/compute.xml*

* + 1. Change machine-type

*<os>*

*<type arch='x86\_64' machine='pc'>hvm</type>*

* 1. **Fuel.xml**

*# vim /root/user1/fuel/deploy/template/virtual\_environment/vms/fuel.xml*

* + 1. Change machine-type

*<os>*

*<type arch='x86\_64' machine='pc'>hvm</type>*

1. Deploy script needs to be started as „root“. Type this command to start deployment.

*# cd /root/user1/fuel/deploy*

*# nohup python deploy.py -iso /home/opnfvuser/opnfv-2016-03-02\_07-59-55.iso -dea templates/virtual\_environment/conf/dea.yaml -dha templates/virtual\_environment/conf/dha.yaml -s /var/lib/libvirt/images &*

**Note1:** There was a known bug for fuel that installation of openstack nodes has not finished correctly. There is a workaround to start deployment again without fuel node installation. If we want only redeploy Openstack without reinstalling fuel node use same command with –nf option (No Fuel).

**Note2** (might be obsolete): We need to specify full path to the iso image /root/user1/fuel/deploy/opnfv-2016-03-02\_07-59-55.iso, otherwise we will receive this error:

*Failed command: isoinfo -d -i opnfv-2016-03-28\_15-10-28.iso | grep -i "Volume id: "*

*Command returned response:*

*Command return code: 1*

**Note 3:** If „could not SSH to fuel VM“ issue is encountered, you can try to specify -dt parameter with value higher than 240 in the deploy script. Then command should look like this:

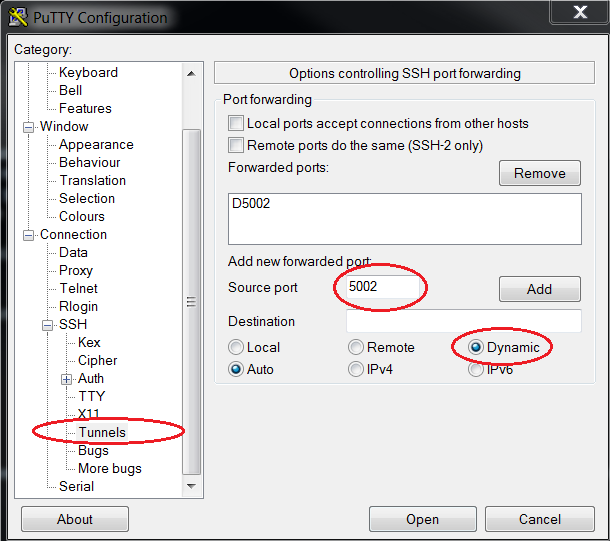
*nohup python deploy.py -iso /home/opnfvuser/opnfv-2016-03-02\_07-59-55.iso -dea templates/virtual\_environment/conf/dea.yaml -dha templates/virtual\_environment/conf/dha.yaml -s /var/lib/libvirt/images -dt 480 &*

### Access GUI of deployed OPNFV environment

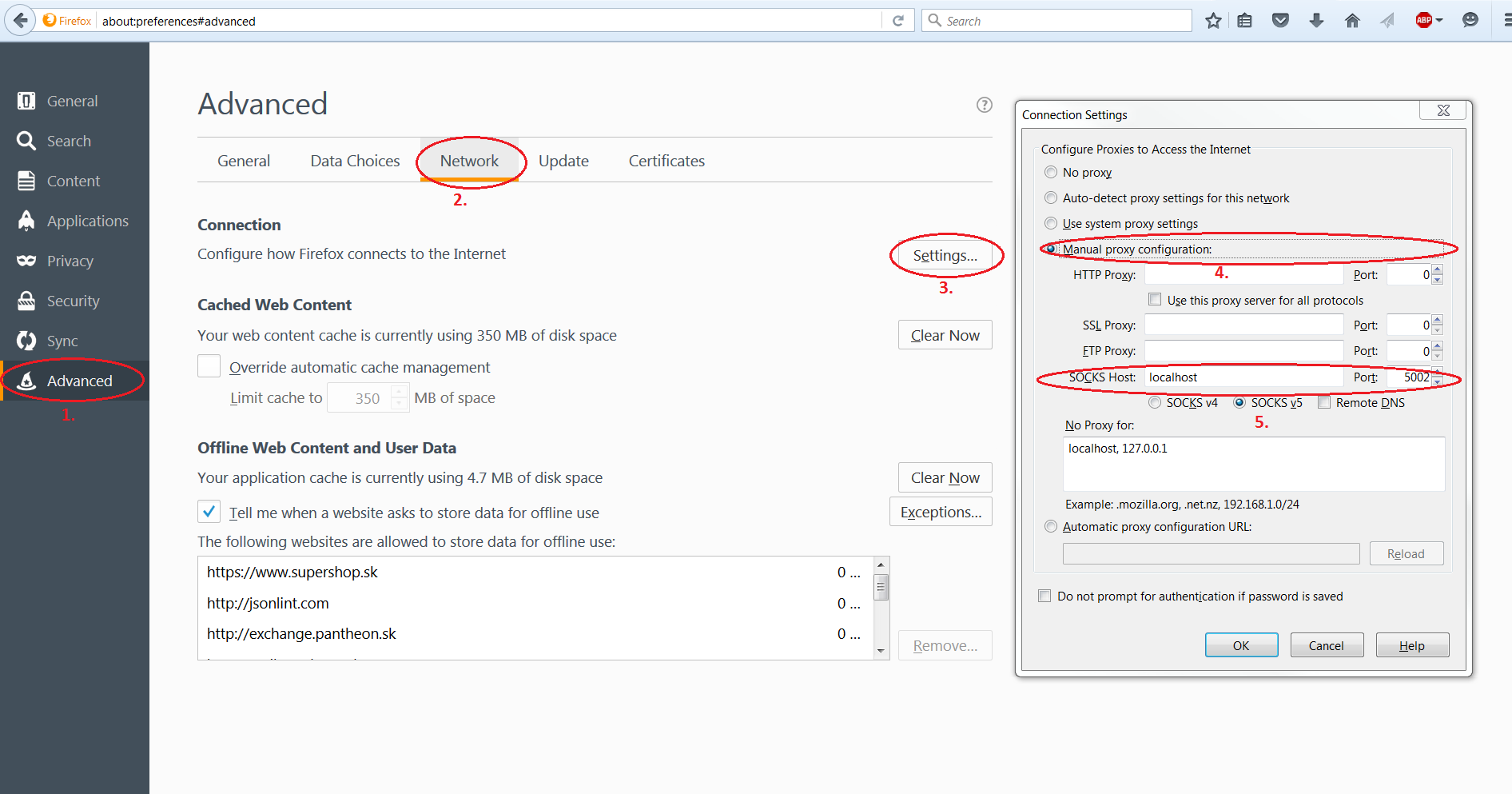
Once the opnfv is deployed we can access fuel GUI, Openstack GUI, ODL GUI using ssh tunnel and SOCKS proxy.

1. a) establish ssh tunnel  
   ssh -D 5002 opnfvuser@10.10.10.xxx –N  
   10.10.10.xxx is the IP address of the sandbox. Type password (opnfv) and press enter.

b) If using putty we need to add dynamic port 5002 under Connection/SSH/Tunnels to the configuration of sandbox VM 10.10.10.xxx



1. confgure SOCKS v5 proxy on our web browser. Below is example for mozzila firefox.  
   localhost, port 5002



1. access IP address of fuel from our browser.  
   Fuel ip 10.20.0.2 directly from our browser (admin/admin)
2. Access Openstack (IP address should be 172.16.0.3) directly from our browser (admin/admin)

**NOTE: Openstack and Fuel GUI are accessible only when we are connected and therefore tunnel created to the OPNFV instance.**

#### See also

Before start it is recommended to look at latest documentation available at:   
<https://www.opnfv.org/software/download>

Other good source of info is opnfv wiki (search for fuel):  
<http://wiki.opnfv.org>