Automation

CHALLENGE 3: WE SHOULD’VE HIRED BETTER PEOPLE!

Katarina Wagnerova
katkaaw@gmail.com
@_KatkaW_

13.07.2016

Table of Contents

1 Mission objectives

It seems that somebody wants Zombies to survive and they do everything they can to stop us from recolonizing the Earth.

Our billionaire has stopped the funding so we can't get any more people to help us with our efforts, therefore we need to put some automation in place.

Somebody has decided that we have to implement two different orchestration tools (not us) that need to be able to manage CentOS and Ubuntu distributions (definitely not us).

We also need to deploy NGINX web servers onto them as well as Hello World application which is hosted on GitHub.

Keeping security in mind, we also need to patch our servers daily.

1.1 Requirements

Following requirements have been derived from the mission objective:

#	Description
R01	Build host platform
R02	Deploy two orchestration tools
R03	Implement provisioning service
R04	Deploy two CentOS servers
R05	Deploy two Ubuntu servers
R06	Create processes for daily patching
R07	Deploy NGINX web server
R08	Create “Hello World” source on GitHub
R09	Deploy “Hello World” onto each server

1.2 Constrains

#	Description
C01	OS types defined as CentOS and Ubuntu
C02	Web server type selected
C04	“Hello World” has to be stored on GitHub

1.3 Assumptions

#	Description
A01	Existing infrastructure will be used
A02	Admins will be able to trigger tasks as needed
A03	Patches are released daily
A04	Servers can be rebooted after patching if required

1.4 Risks

#	Description	Risk mitigation
RI01	Lack of Linux and Automation skills	Train people or give them enough time to learn it ☺
RI02	Buggy patches can be released	Deploy patches in batches
RI03	Orchestration platform can fail	Deploy two independent platforms

2 Logical design decisions

2.1 Infrastructure

Logical Design Decision	LDD 1. Infrastructure planning
Decision options	1. Use existing infrastructure 2. Build new infrastructure
Option selected	1.
Justification	We will utilize existing infrastructure for the resource cluster and deploy a new management cluster.
Associated risks	Earth datacenter is not fully in our hands yet and there might be configuration changes needed.
Implications	Additional components will need to be deployed to support new functions.
Requirement met	R01

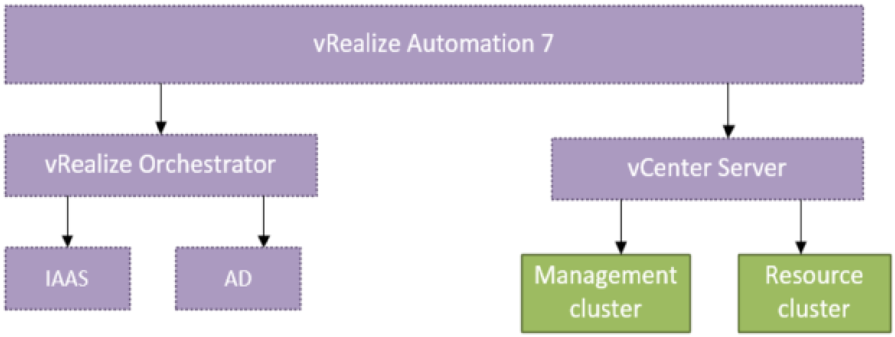
2.2 Provisioning

Logical Design Decision	LDD 2. Provisioning Service
Decision options	1. Deploy VM from templates 2. Use orchestration tools for deployment 3. Use cloud management tools
Option selected	3.
Justification	Provides simple deployment portal and multiple configuration options
Associated risks	Added complexity of the solution
Implications	Additional tools need to be installed
Requirement met	R03

3 Physical design decisions

3.1 Infrastructure

Physical Design Decision	PDD 1. Infrastructure
Decision options	1. Existing VMware environment 2. New deployment
Option selected	1.
Justification	Existing stretched cluster will be used for resource environment. Additional management cluster will be deployed to support vRA management tools.
Associated risks	Increased complexity and risk of failure
Implications	Increased cost and need to purchase new hardware
Requirement met	R01
Logical Design Decision Referenced	LDD 1.



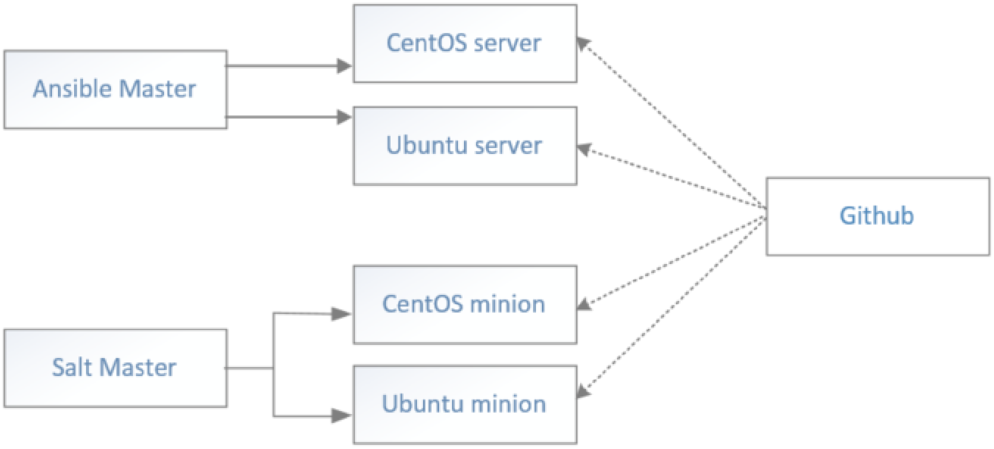
3.2 Provisioning

Physical Design Decision	PDD 2. Provisioning Service
Decision options	1. vRealize Automation 7 2. OpenStack
Option selected	1.
Justification	Some existing knowledge about the solution and the possibility to test it.
Associated risks	Budgetary risks: Increased costs
Implications	Additional tools need to be deployed: ○ vRA appliance ○ IaaS server ○ vRO
Requirement met	R03
Logical Design Decision Referenced	LDD 2.

3.3 Orchestration

Physical Design Decision	PDD 3. Orchestration tools
Decision options	1. Chef 2. Puppet 3. Ansible 4. Salt
Option selected	3. & 4.
Justification	We were looking for push solutions with immediate remote execution. Both Ansible and Salt use Python packages and use similar syntax which eases management. Ansible will be running on a CentOS 7 VM. Salt will be running on Ubuntu 14.04 VM Both tools are able to execute patching remotely. Tools will be also used to deploy application.
Associated risks	Decreased scalability
Implications	Additional CentOS and Ubuntu servers need to be deployed, configured and managed.
Requirement met	R02, R06, R07, R08, R09

Logical Design Decision Referenced	LDD 3.



3.4 Operating Systems

Physical Design Decision	PDD 3. Operating Systems
Decision options	1. CentOS 6 2. CentOS 7 3. Ubuntu 16.04 4. Ubuntu 14.04
Option selected	2. & 4.
Justification	Versions have been selected based on industry recognized standards.
Associated risks	CentOS 7 : newest version, possible bugs waiting to be found Ubuntu 14.04: older version, might require upgrade to support future features
Implications	Additional CentOS and Ubuntu servers need to be deployed, configured and managed.
Requirement met	R04, R05

4 Setup Guide

4.1 Provisioning Infrastructure

Following tools need to be deployed and configured on top of the existing infrastructure:

- Deploy vRealize Automation 7 appliance
- Deploy vRealize Orchestrator (internal)
- Deploy IaaS Server
- Create blueprint for VM creations
- Publish blueprint
- Create Service
- Add blueprint to Service
- Assign Entitled groups to the service

Please refer to Appendix A for details.

4.2 Templates

We have to create two VM templates, one will be used for CentOS and the other one for Ubuntu.

VM configuration will be the same on both:

- 1 vCPU
- 1 GB RAM
- 1 20GB disk
- 1 VXNET3 network adapter

Note: Configuration above has been used in testing environment. We would base resource assignment in real environment on application requirements.

4.2.1 CentOS template

Perform following steps to install and configure CentOS template:

- Create Virtual Machine
- Download CentOS installation iso from https://www.centos.org/download/
- Attach iso file to VM
- Turn on VM
- Install OS by following on-screen instructions
- Login as root
  - Sudo su -
- Configure IPs:
  - vi /etc/sysconfig/network-scripts/ifcfg-eno16777984
- Update system
  - yum update
- Install EPEL repositories
  - yum install epel-release
- Install Python packages
  - yum -y install python-pip
- Install git
  - yum install git
- Install VM Tools
  - yum install open-vm-tools
- Create users
  - Personal (kwagnerova)
  - Ansible
  - Salt
- Turn off VM
- Convert to Template

4.2.2 Ubuntu template

Perform following steps to install and configure Ubuntu template:

- Create Virtual Machine
- Download CentOS installation iso from http://www.ubuntu.com/download
- Attach iso file to VM
- Turn on VM
- Install OS by following on-screen instructions
- Login as root
  - Sudo su -

- Configure IPs:
  - vi /etc/network/interfaces
- Update system
  - apt-get update
  - apt-get dist-upgrade
- Install Python packages
  - apt-get install python-pip
- Install git
  - apt-get install git
- Install VM Tools
  - mount VMTools
  - mkdir /mnt/cdrom
  - mount /dev/cdrom /mnt/cdrom or sudo mount /dev/sr0 /mnt/cdrom
  - ls /mnt/cdrom
  - tar xzvf /mnt/cdrom/VMwareTools-x.x.x-xxxx.tar.gz -C /tmp/
  - cd /tmp/vmware-tools-distrib/
  - ./vmware-install.pl –d
- Create users
  - Personal (kwagnerova)
  - Ansible
  - Salt
- Allow root login
  - vi /etc/ssh/sshd_config

```
# Authentication:
LoginGraceTime 120
#PermitRootLogin without-password
PermitRootLogin yes
StrictModes yes
```

- Turn off VM
- Convert to Template

4.3 HelloWorld

HelloWorld will be a HTML page stored on GitHub, execute following steps to create it:

- Create GitHub.com account and login
- Create a repository
  - Click on the + icon
  - Name repository ‘hello-world’
  - Public
  - Initialize with README
  - Create Repository
- Create HelloWorld
  - Create new file called index.html
  - Add following lines:

```
<HTML>
  <HEAD>
    <TITLE>
      Hello World!
    </TITLE>
  </HEAD>
  <BODY>
          <H1>Hey Zombies!</H1>
```

```
            <P>If you keep eating people you're gonna have a bad time</P>
      </BODY>
    </HTML>
```

- Add your public SSH key
    - Click on account settings
    - SSH and GPG keys
    - New SSH key

4.4 Ansible

4.4.1 Ansible Master

- Deploy CentOS VM
- Login as root
    - Sudo su -
- Configure IPs:
    - vi /etc/sysconfig/network-scripts/ifcfg-eno16777984
- Change hostname
    - vi /etc/hostname
    - vi /etc/hosts
- Install Ansible
    - yum install Ansible
- Generate RSA keys
    - ssh-keygen -t root
- Copy public keys to templates
    - cat ~/.ssh/id_rsa.pub | ssh root@10.5.1.200 "mkdir ~/.ssh; cat >> ~/.ssh/authorized_keys"
    - cat ~/.ssh/id_rsa.pub | ssh root@10.5.1.201 "mkdir ~/.ssh; cat >> ~/.ssh/authorized_keys"
- Add clients to host list
    - vi /etc/ansible/hosts

      ```
      [testcentos]
      10.5.1.203

      [testubuntu]
      10.5.1.204
      ```
- Create folder structure
    - mkdir /etc/Ansible/playbooks
    - mkdir /etc/Ansible/tasks
- Create tasks for nginx installation
    - vi /etc/ansible/tasks/install_nginx_centos.yml

```
---
# File name ./install_nginx_centos.yml

 - name: NGINX | Installing NGINX repo rpm
   yum:
    name: http://nginx.org/packages/centos/7/noarch/RPMS/nginx-release-centos-7-0.el7.ngx.noarch.rpm

 - name: NGINX | Installing NGINX
   yum:
    name: nginx
    state: latest

 - name: NGINX | Starting NGINX
   service:
```

```
    name: nginx
    state: started
```

- vi /etc/ansible/tasks/install_nginx_ubuntu.yml

```
---
# file name = ./deploy-nginx-ubuntu.yml

- name: NGINX | Adding NGINX signing key
  apt_key: url=http://nginx.org/keys/nginx_signing.key state=present

- name: NGINX | Adding sources.list deb url for NGINX
  lineinfile: dest=/etc/apt/sources.list line="deb http://nginx.org/packages/mainline/ubuntu/
trusty nginx"

- name: NGINX Plus | Adding sources.list deb-src url for NGINX
  lineinfile: dest=/etc/apt/sources.list line="deb-src http://nginx.org/packages/mainline/ubuntu/
trusty nginx"

- name: NGINX | Updating apt cache
  apt:
    update_cache: yes

- name: NGINX | Installing NGINX
  apt:
    pkg: nginx
    state: latest
```

- Create task for HelloWorld deployment
  - vi /etc/ansible/tasks/helloworld_centos.yml

```
---
# File name ./helloworldcentos.yml

 - name: HelloWorld | Getting HelloWorld from GitHub
   git:
    repo=https://github.com/katkaaw/hello-world
    dest=/data/www/
```

  - vi /etc/ansible/tasks/helloworld_ubuntu.yml

```
---
# File name ./helloworldcentos.yml

 - name: HelloWorld | Getting HelloWorld from GitHub
   git:
    repo=https://github.com/katkaaw/hello-world
    dest=/data/www/
```

- Create playbook for CentOS
  - vi /etc/ansible/playbooks/deploy-nginx-centos.ym

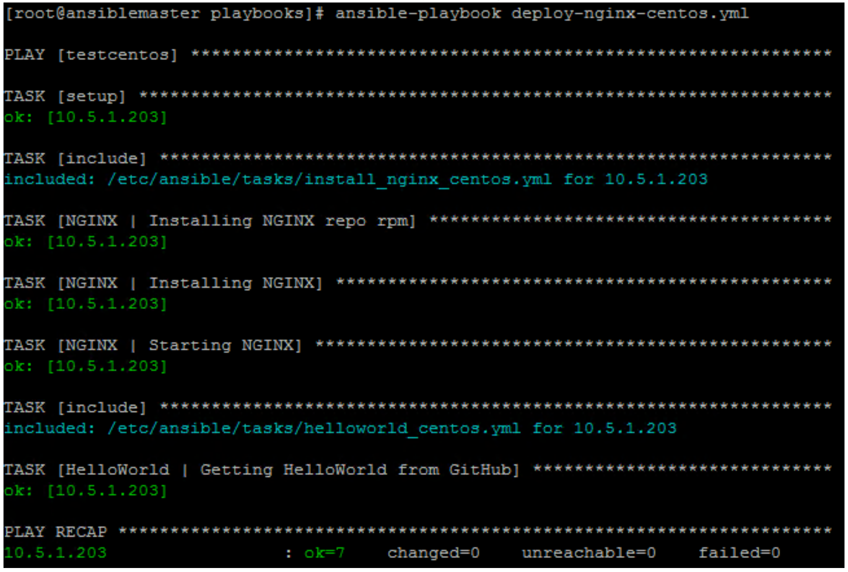
```
---
# file name = ./deploy-nginx-centos.yml

 - hosts: testcentos
   tasks:
    - include: '/etc/ansible/tasks/install_nginx_centos.yml'
    - include: '/etc/ansible/tasks/helloworld_centos.yml'
```

- Create playbook for Ubuntu
  - vi /etc/ansible/playbooks/deploy-nginx-ubuntu.ym

```
---
# file name = ./deploy-nginx-ubuntu.yml

 - hosts: testubuntu
   tasks:
    - include: '/etc/ansible/tasks/install_nginx_ubuntu.yml'
    - include: '/etc/ansible/tasks/helloworld_ubuntu.yml'
```

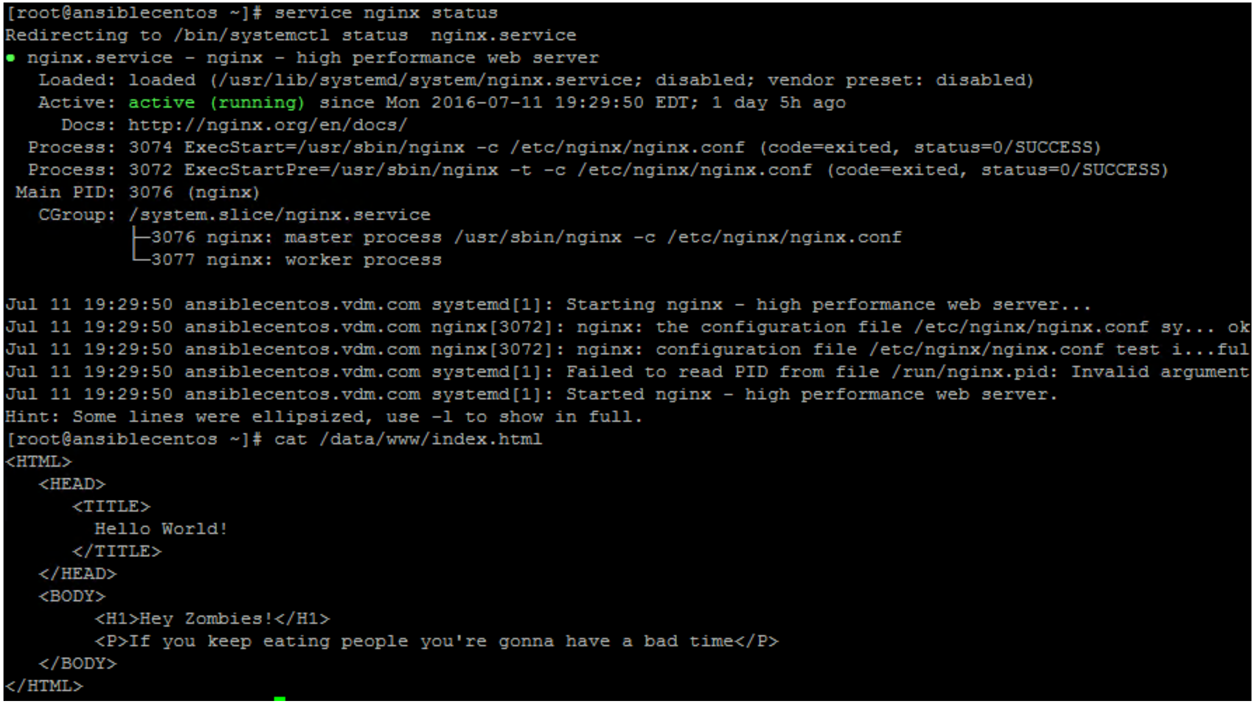
4.4.2 Execution

- For Centos, run ‘ansible-playbook deploy-nginx-centos.yml’



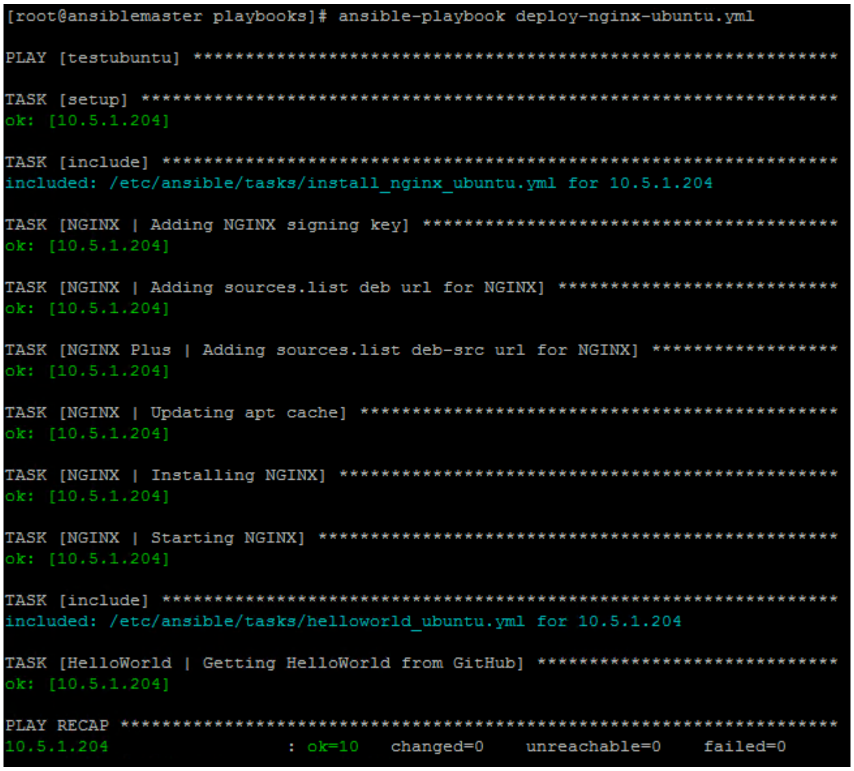
```
[root@ansiblemaster playbooks]# ansible-playbook deploy-nginx-centos.yml

PLAY [testcentos] *************************************************************

TASK [setup] ******************************************************************
ok: [10.5.1.203]

TASK [include] ****************************************************************
included: /etc/ansible/tasks/install_nginx_centos.yml for 10.5.1.203

TASK [NGINX | Installing NGINX repo rpm] **************************************
ok: [10.5.1.203]

TASK [NGINX | Installing NGINX] ***********************************************
ok: [10.5.1.203]

TASK [NGINX | Starting NGINX] *************************************************
ok: [10.5.1.203]

TASK [include] ****************************************************************
included: /etc/ansible/tasks/helloworld_centos.yml for 10.5.1.203

TASK [HelloWorld | Getting HelloWorld from GitHub] ****************************
ok: [10.5.1.203]

PLAY RECAP ********************************************************************
10.5.1.203                 : ok=7    changed=0    unreachable=0    failed=0
```

You can also verify the result on the client side:



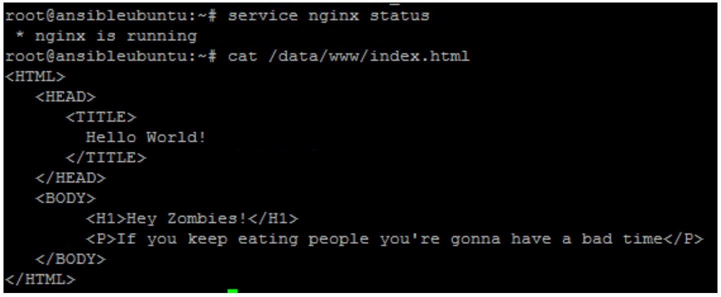
```
[root@ansiblecentos ~]# service nginx status
Redirecting to /bin/systemctl status  nginx.service
● nginx.service - nginx - high performance web server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled; vendor preset: disabled)
   Active: active (running) since Mon 2016-07-11 19:29:50 EDT; 1 day 5h ago
     Docs: http://nginx.org/en/docs/
  Process: 3074 ExecStart=/usr/sbin/nginx -c /etc/nginx/nginx.conf (code=exited, status=0/SUCCESS)
  Process: 3072 ExecStartPre=/usr/sbin/nginx -t -c /etc/nginx/nginx.conf (code=exited, status=0/SUCCESS)
 Main PID: 3076 (nginx)
   CGroup: /system.slice/nginx.service
           ├─3076 nginx: master process /usr/sbin/nginx -c /etc/nginx/nginx.conf
           └─3077 nginx: worker process

Jul 11 19:29:50 ansiblecentos.vdm.com systemd[1]: Starting nginx - high performance web server...
Jul 11 19:29:50 ansiblecentos.vdm.com nginx[3072]: nginx: the configuration file /etc/nginx/nginx.conf sy... ok
Jul 11 19:29:50 ansiblecentos.vdm.com nginx[3072]: nginx: configuration file /etc/nginx/nginx.conf test i...ful
Jul 11 19:29:50 ansiblecentos.vdm.com systemd[1]: Failed to read PID from file /run/nginx.pid: Invalid argument
Jul 11 19:29:50 ansiblecentos.vdm.com systemd[1]: Started nginx - high performance web server.
Hint: Some lines were ellipsized, use -l to show in full.
[root@ansiblecentos ~]# cat /data/www/index.html
<HTML>
   <HEAD>
      <TITLE>
        Hello World!
      </TITLE>
   </HEAD>
   <BODY>
        <H1>Hey Zombies!</H1>
        <P>If you keep eating people you're gonna have a bad time</P>
   </BODY>
</HTML>
```

- For Ubuntu, run ‘ansible-playbook deploy-nginx-ubuntu.yml’



```
[root@ansiblemaster playbooks]# ansible-playbook deploy-nginx-ubuntu.yml

PLAY [testubuntu] **************************************************************

TASK [setup] *******************************************************************
ok: [10.5.1.204]

TASK [include] *****************************************************************
included: /etc/ansible/tasks/install_nginx_ubuntu.yml for 10.5.1.204

TASK [NGINX | Adding NGINX signing key] ****************************************
ok: [10.5.1.204]

TASK [NGINX | Adding sources.list deb url for NGINX] ***************************
ok: [10.5.1.204]

TASK [NGINX Plus | Adding sources.list deb-src url for NGINX] ******************
ok: [10.5.1.204]

TASK [NGINX | Updating apt cache] **********************************************
ok: [10.5.1.204]

TASK [NGINX | Installing NGINX] ************************************************
ok: [10.5.1.204]

TASK [NGINX | Starting NGINX] **************************************************
ok: [10.5.1.204]

TASK [include] *****************************************************************
included: /etc/ansible/tasks/helloworld_ubuntu.yml for 10.5.1.204

TASK [HelloWorld | Getting HelloWorld from GitHub] *****************************
ok: [10.5.1.204]

PLAY RECAP *********************************************************************
10.5.1.204                 : ok=10   changed=0    unreachable=0    failed=0
```

You can also verify the result on the client side:



```
root@ansibleubuntu:~# service nginx status
 * nginx is running
root@ansibleubuntu:~# cat /data/www/index.html
<HTML>
   <HEAD>
      <TITLE>
        Hello World!
      </TITLE>
   </HEAD>
   <BODY>
        <H1>Hey Zombies!</H1>
        <P>If you keep eating people you're gonna have a bad time</P>
   </BODY>
</HTML>
```

Note: In real life we would like to have this running under dedicated user. I have created Ansible user, generated RSA keys and distributed them to the systems and changed configuration on Ansible master. Unfortunately, I was not able to get it running in time.

4.5 Salt

4.5.1 Salt Master

- Deploy Ubuntu VM
- Login as root
  - Sudo su -
- Configure IPs:
  - vi /etc/network/interfaces
- Change hostname
  - vi /etc/hostname
  - vi /etc/hosts
- Install python package
  - apt-get install python-software-properties

- Add Salt repository
  - add-apt-repository ppa:saltstack/salt
- Update packages
  - apt-get update
- Install salt-master
  - apt-get install salt-master
- Configure salt-master
  - vi /etc/salt/master

    # The address of the interface to bind to:
    interface: 10.5.1.220
- Generate RSA keys
  - ssh-keygen -t root
- Copy public keys to minions
  - cat ~/.ssh/id_rsa.pub | ssh root@10.5.1.221 "mkdir ~/.ssh; cat >> ~/.ssh/authorized_keys"
  - cat ~/.ssh/id_rsa.pub | ssh root@10.5.1.222 "mkdir ~/.ssh; cat >> ~/.ssh/authorized_keys"

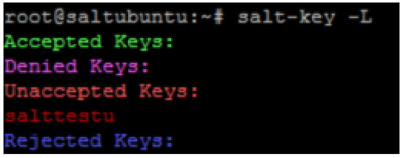
4.5.2 Salt Minions

- Deploy Ubuntu VM
- Login as root
  - Sudo su -
- Configure IPs:
  - vi /etc/network/interfaces
- Change hostname
  - vi /etc/hostname
  - vi /etc/hosts
- Install python package
  - apt-get install python-software-properties
- Add Salt repository
  - add-apt-repository ppa:saltstack/salt
- Update packages
  - apt-get update
- Install salt-minion
  - apt-get install salt-minion
- Configure salt-minion
  - vi /etc/salt/minion

    # The address of the interface to bind to:
    interface: 10.5.1.210
- Restart salt-minion process
  - service salt-minion restart
- Follow the same process for CentOS machine (installation commands may vary, please refer to CentOS template)

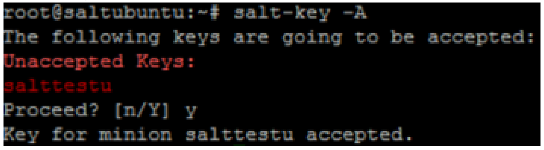
4.5.3 Setting up communication

- List available minion keys on salt master
  - Salt-key -L



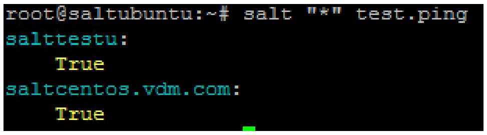
```
root@saltubuntu:~# salt-key -L
Accepted Keys:
Denied Keys:
Unaccepted Keys:
salttestu
Rejected Keys:
```

- Add available minion keys to salt master
    - Salt-key –A



```
root@saltubuntu:~# salt-key -A
The following keys are going to be accepted:
Unaccepted Keys:
salttestu
Proceed? [n/Y] y
Key for minion salttestu accepted.
```

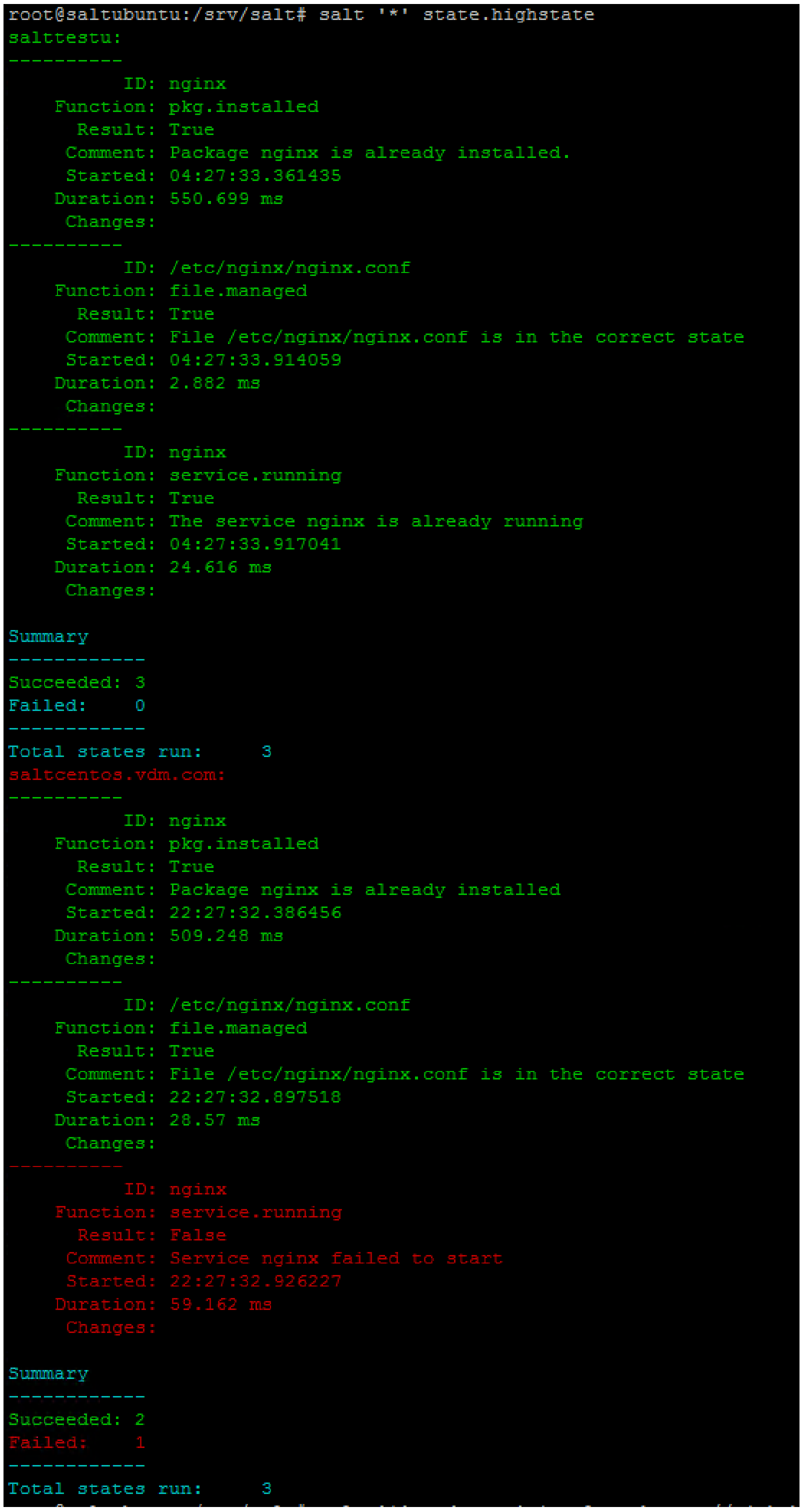
- Verify connection



```
root@saltubuntu:~# salt "*" test.ping
salttestu:
    True
saltcentos.vdm.com:
    True
```

4.5.4 Execution

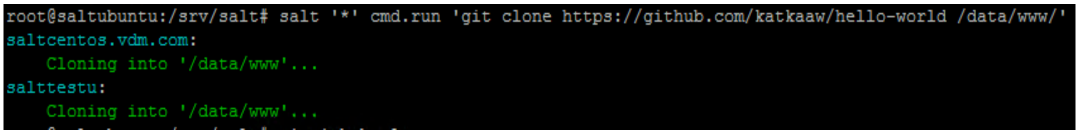
- Install nginx
    - salt '*' state.highstate



```
root@saltubuntu:/srv/salt# salt '*' state.highstate
salttestu:
----------
          ID: nginx
    Function: pkg.installed
      Result: True
     Comment: Package nginx is already installed.
     Started: 04:27:33.361435
    Duration: 550.699 ms
     Changes:
----------
          ID: /etc/nginx/nginx.conf
    Function: file.managed
      Result: True
     Comment: File /etc/nginx/nginx.conf is in the correct state
     Started: 04:27:33.914059
    Duration: 2.882 ms
     Changes:
----------
          ID: nginx
    Function: service.running
      Result: True
     Comment: The service nginx is already running
     Started: 04:27:33.917041
    Duration: 24.616 ms
     Changes:

Summary
------------
Succeeded: 3
Failed:    0
------------
Total states run:     3
saltcentos.vdm.com:
----------
          ID: nginx
    Function: pkg.installed
      Result: True
     Comment: Package nginx is already installed
     Started: 22:27:32.386456
    Duration: 509.248 ms
     Changes:
----------
          ID: /etc/nginx/nginx.conf
    Function: file.managed
      Result: True
     Comment: File /etc/nginx/nginx.conf is in the correct state
     Started: 22:27:32.897518
    Duration: 28.57 ms
     Changes:
----------
          ID: nginx
    Function: service.running
      Result: False
     Comment: Service nginx failed to start
     Started: 22:27:32.926227
    Duration: 59.162 ms
     Changes:

Summary
------------
Succeeded: 2
Failed:    1
------------
Total states run:     3
```

Note: nginx service failed to start on centos machine, I have verified the configuration files but was not able to find the reason why.

- Get HelloWorld
  - salt '*' cmd.run 'git clone https://github.com/katkaaw/hello-world /data/www/'



```
root@saltubuntu:/srv/salt# salt '*' cmd.run 'git clone https://github.com/katkaaw/hello-world /data/www/'
saltcentos.vdm.com:
    Cloning into '/data/www'...
salttestu:
    Cloning into '/data/www'...
```

Note: Again, ideally this would be running under salt account instead of root. But. Time.

4.6 Patching Process

All servers have to be patched daily. In order to verify and test released patches, they will be installed on test servers first. Production systems will be scheduled to run several hours later. This will allow the admins to react in case of a faulty patch distribution.

Patching can be automated using Ansible by performing the following:

- Create update playbooks
    - vi /etc/ansible/playbooks/update_test.yml

      ```
      ---
      # file name = ./update_test.yml

      - hosts: testservers
        tasks:
          - name: update and upgrade using apt
            apt: upgrade=safe update_cache=yes
      ```

    - vi /etc/ansible/playbooks/update_prod.yml

      ```
      ---
      # file name = ./update_prod.yml

      - hosts: prodservers
        tasks:
          - name: update and upgrade using apt
            apt: upgrade=safe update_cache=yes
      ```

- Updates will be scheduled using cron

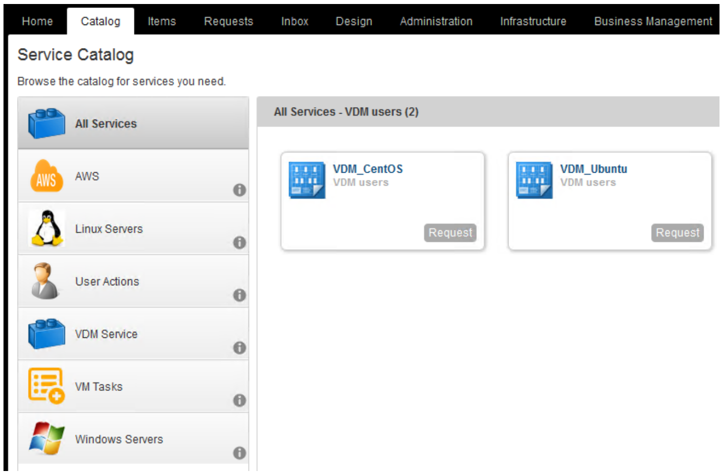
Salt can be used as well, either by creating a grain or scheduling the following:

- salt ‘test’ cmd.run "apt-get -y update"
- salt ‘prod’ cmd.run "apt-get -y update"

4.7 Provisioning Process

In order to deploy a new VM perform following steps:

- Login to the Provisioning portal
- Go to Catalog
- Request deployment of your desired Operating System



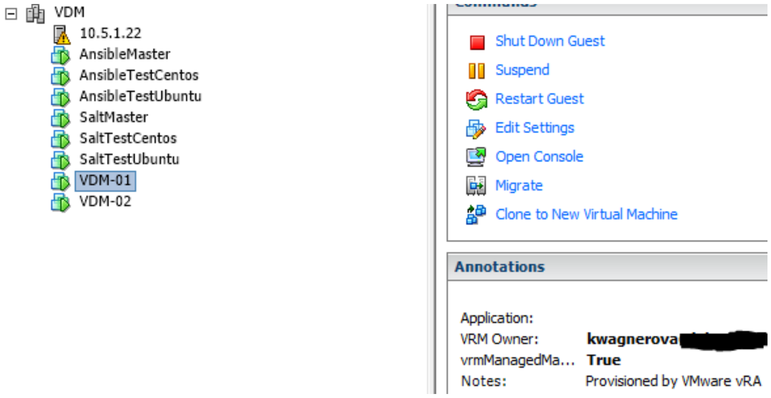
Service Catalog

Browse the catalog for services you need.

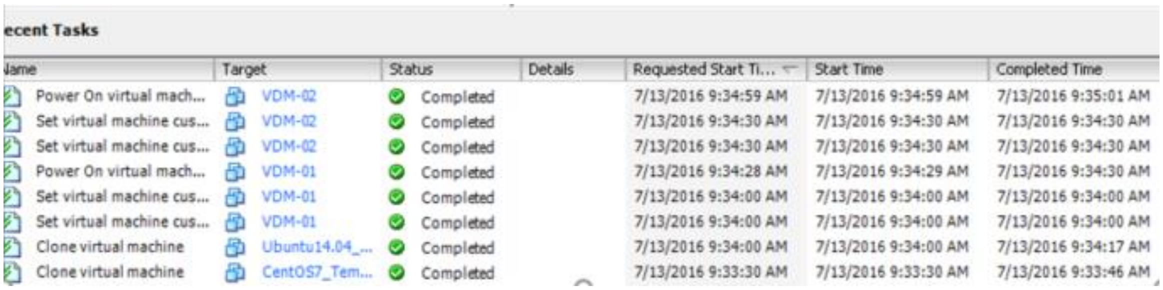
All Services - VDM users (2)

VDM_CentOS VDM users	VDM_Ubuntu VDM users

- VM will be deployed

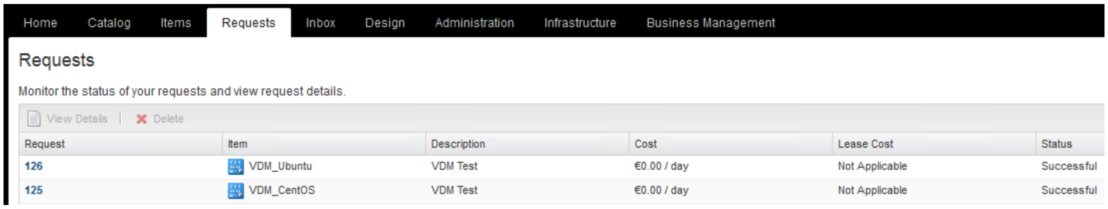


Application:
VRM Owner: kwagnerova[redacted]
vrmManagedMa… True
Notes: Provisioned by VMware vRA


ecent Tasks

lame	Target	Status	Details	Requested Start Ti...	Start Time	Completed Time
Power On virtual mach...	VDM-02	Completed		7/13/2016 9:34:59 AM	7/13/2016 9:34:59 AM	7/13/2016 9:35:01 AM
Set virtual machine cus...	VDM-02	Completed		7/13/2016 9:34:30 AM	7/13/2016 9:34:30 AM	7/13/2016 9:34:30 AM
Set virtual machine cus...	VDM-02	Completed		7/13/2016 9:34:30 AM	7/13/2016 9:34:30 AM	7/13/2016 9:34:30 AM
Power On virtual mach...	VDM-01	Completed		7/13/2016 9:34:28 AM	7/13/2016 9:34:29 AM	7/13/2016 9:34:30 AM
Set virtual machine cus...	VDM-01	Completed		7/13/2016 9:34:00 AM	7/13/2016 9:34:00 AM	7/13/2016 9:34:00 AM
Set virtual machine cus...	VDM-01	Completed		7/13/2016 9:34:00 AM	7/13/2016 9:34:00 AM	7/13/2016 9:34:00 AM
Clone virtual machine	Ubuntu14.04_...	Completed		7/13/2016 9:34:00 AM	7/13/2016 9:34:00 AM	7/13/2016 9:34:17 AM
Clone virtual machine	CentOS7_Tem...	Completed		7/13/2016 9:33:30 AM	7/13/2016 9:33:30 AM	7/13/2016 9:33:46 AM

- Go to Requests to confirm successful deployment



Requests

Monitor the status of your requests and view request details.

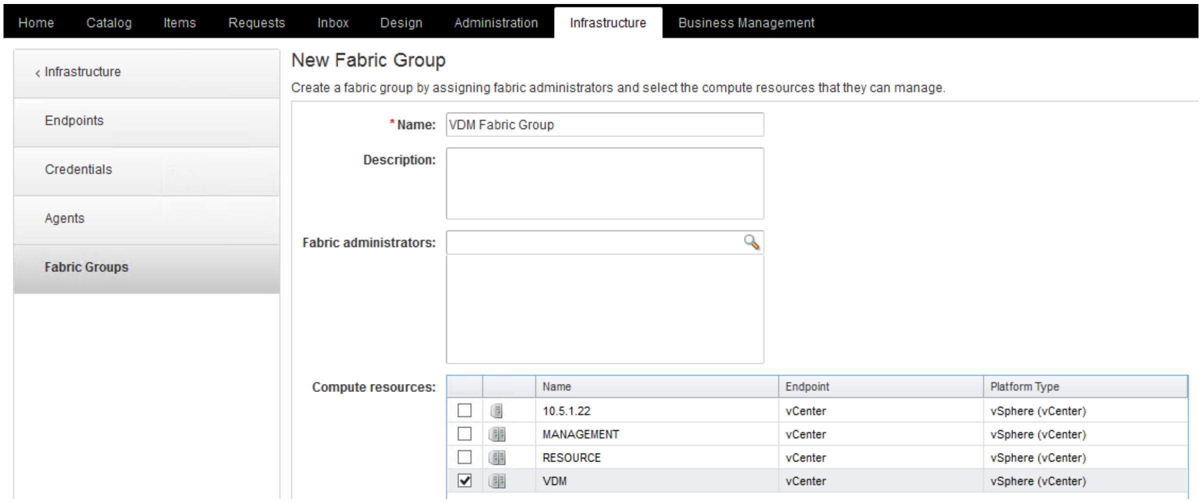
Request	Item	Description	Cost	Lease Cost	Status
126	VDM_Ubuntu	VDM Test	€0.00 / day	Not Applicable	Successful
125	VDM_CentOS	VDM Test	€0.00 / day	Not Applicable	Successful

5 References

Every single page on the internet about CentOS, Ubuntu, Ansible, Salt,... EVERY page.

6 Appendix A – vRA configuration

- Create Fabric Group:



New Fabric Group

Create a fabric group by assigning fabric administrators and select the compute resources that they can manage.

* Name: VDM Fabric Group

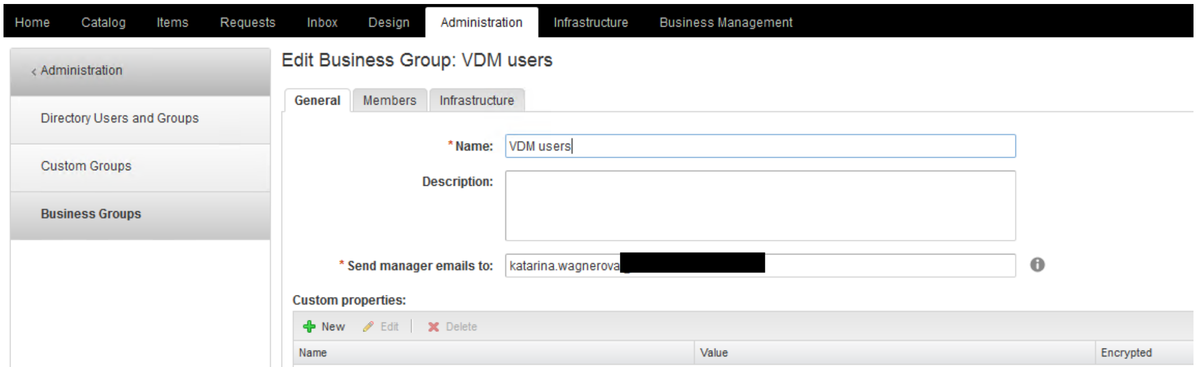
Description:

Fabric administrators:

Compute resources:

	Name	Endpoint	Platform Type
□	10.5.1.22	vCenter	vSphere (vCenter)
□	MANAGEMENT	vCenter	vSphere (vCenter)
□	RESOURCE	vCenter	vSphere (vCenter)
☑	VDM	vCenter	vSphere (vCenter)

- Create Business group



Edit Business Group: VDM users

* Name: VDM users

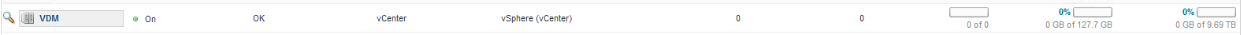
Description:

* Send manager emails to: katarina.wagnerova[redacted]

Custom properties:

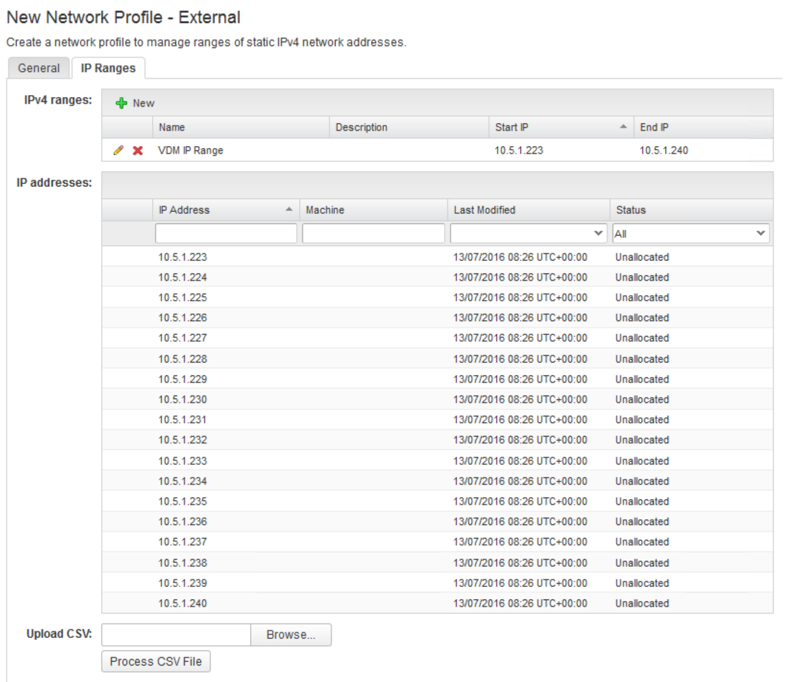
Name	Value	Encrypted

- Add Compute resource



VDM	On	OK	vCenter	vSphere (vCenter)	0	0	0 of 0	0% 0 GB of 127.7 GB	0% 0 GB of 9.69 TB

- Create Network Profile



New Network Profile - External

Create a network profile to manage ranges of static IPv4 network addresses.

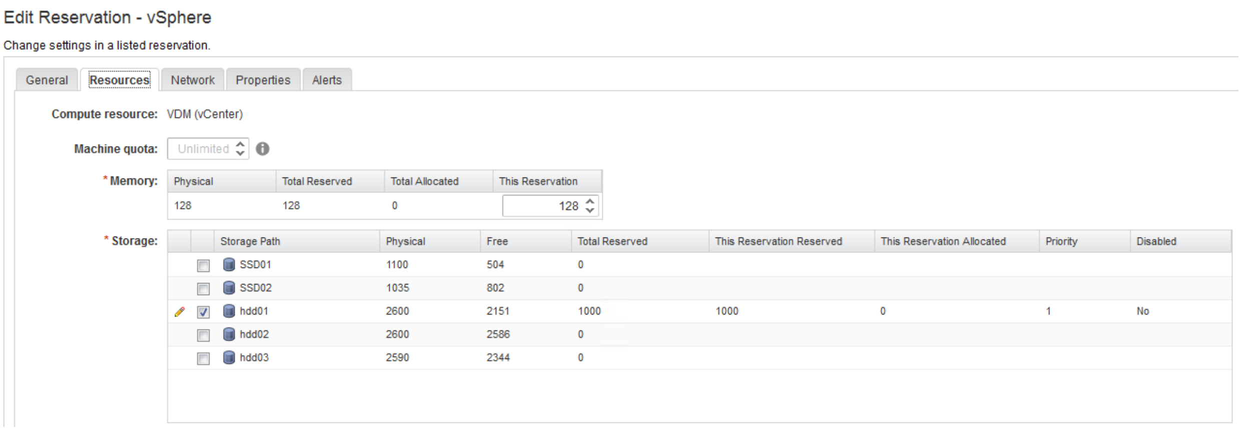
IPv4 ranges: + New

Name	Description	Start IP	End IP
VDM IP Range		10.5.1.223	10.5.1.240

IP addresses:

IP Address	Machine	Last Modified	Status
			All
10.5.1.223		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.224		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.225		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.226		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.227		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.228		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.229		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.230		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.231		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.232		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.233		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.234		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.235		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.236		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.237		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.238		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.239		13/07/2016 08:26 UTC+00:00	Unallocated
10.5.1.240		13/07/2016 08:26 UTC+00:00	Unallocated

Upload CSV:



Edit Reservation - vSphere

Change settings in a listed reservation.

Compute resource: VDM (vCenter)

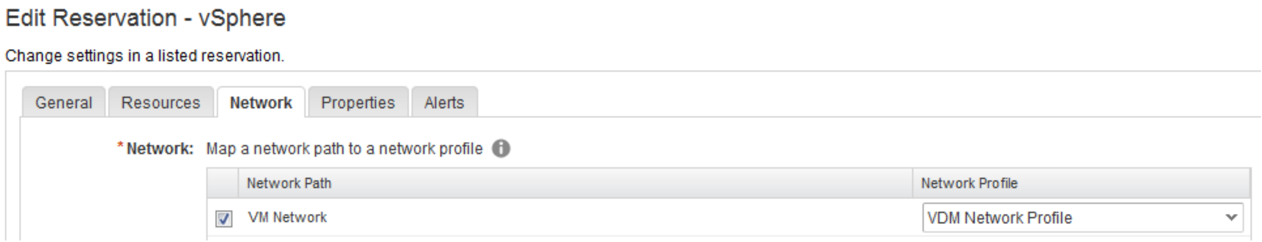
Machine quota: Unlimited

* **Memory:**

Physical	Total Reserved	Total Allocated	This Reservation
128	128	0	128

* **Storage:**

		Storage Path	Physical	Free	Total Reserved	This Reservation Reserved	This Reservation Allocated	Priority	Disabled
	□	SSD01	1100	504	0				
	□	SSD02	1035	802	0				
✎	☑	hdd01	2600	2151	1000	1000	0	1	No
	□	hdd02	2600	2586	0				
	□	hdd03	2590	2344	0				



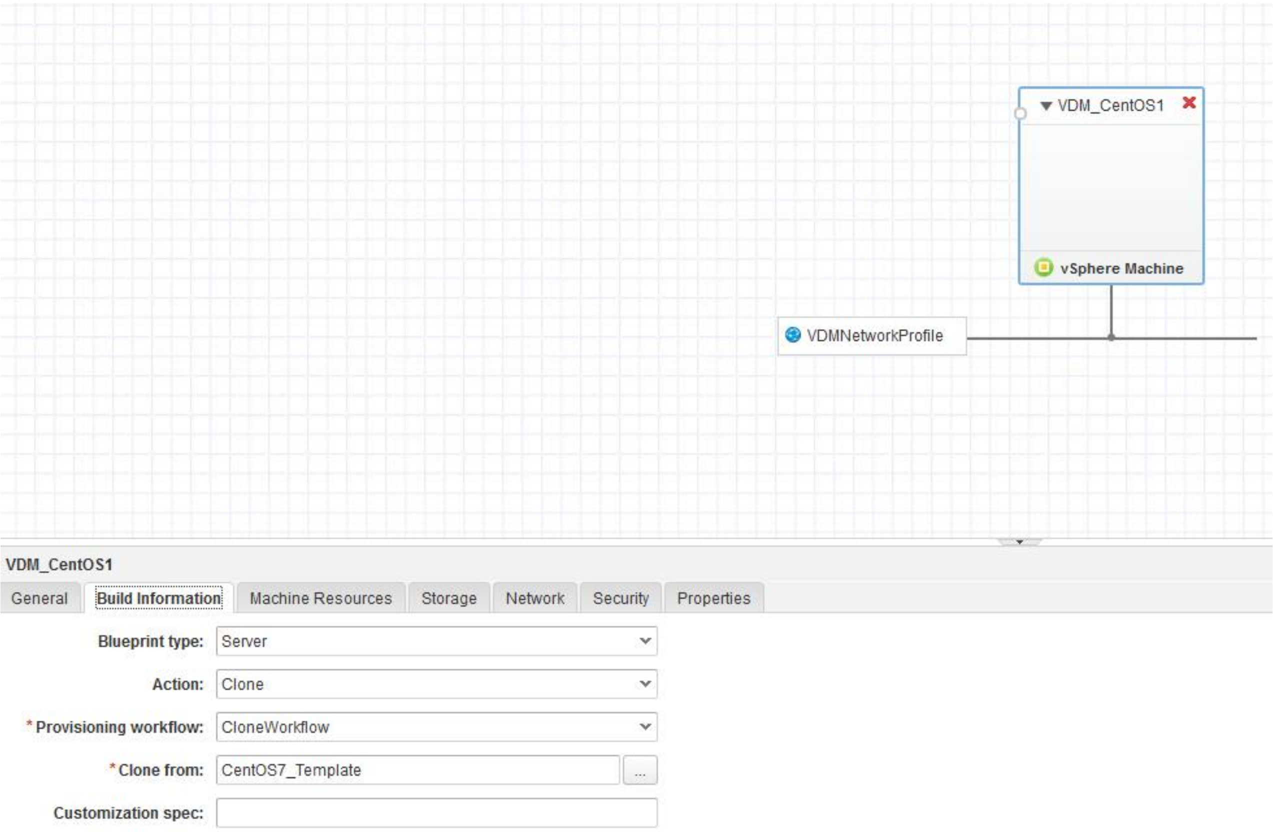
Edit Reservation - vSphere

Change settings in a listed reservation.

* **Network:** Map a network path to a network profile

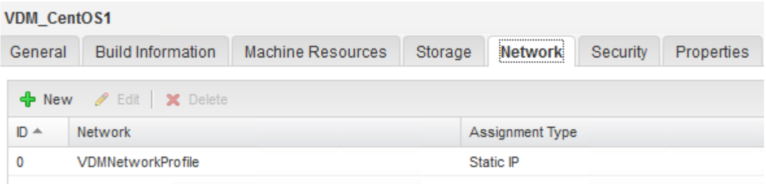
	Network Path	Network Profile
☑	VM Network	VDM Network Profile

- Create blueprints



VDM_CentOS1

Blueprint type:	Server
Action:	Clone
* Provisioning workflow:	CloneWorkflow
* Clone from:	CentOS7_Template
Customization spec:	


VDM_CentOS1

ID	Network	Assignment Type
0	VDMNetworkProfile	Static IP