EXTRA 1:
SETTING UP THE MULTI NODE CLUSTER USING ANSIBLE PLAYBOOK

What You'll Need

In order to successfully install Kubernetes (and create a cluster), you'll need at least two machines. I'll be demonstrating on two CentOS 8 servers, running on IP addresses:
You'll also need a user account with sudo privileges and access to the root user account.
Requirement's: two vm's with minimum 2gb ram to master and 1gb ram to slave and must contain 2 cpu's (both)

MASTER_NODE

STEP1: configuring yum in the master node , creating a directory and mounting the software into it and creating dvd1.repo and dvd2.repo

```
- name: kubernetes_setup
  become: true
  gather_facts: No
  hosts:  server1
  tasks:
  - name: create_dvd_folder
    file:
      path:  /root/dvd
      state:  directory
      mode:  "0755"

  - name: mount_dvd
    mount:
      path:  /dvd/
      src:  /dev/cdrom
      fstype:  iso9660
      opts:  ro,loop
      state:  mounted

  - name: yum_repo_BaseOS
    yum_repository:
      name: BaseOS
```

```
    description: Local_baseos
    file: dvd1
    baseurl: file:///dvd/BaseOS
    gpgcheck: no

- name: yum_repo_AppStream
  yum_repository:
    name: AppStream
    description: Local_appstream
    file: dvd2
    baseurl: file:///dvd/AppStream
    gpgcheck: no
```

```
[root@localhost html]# mkdir /dvd
mkdir: cannot create directory ‘/dvd’: File exists
[root@localhost html]# cd /dvd
[root@localhost dvd]# ls
AppStream  EFI   extra_files.json  images    media.repo              RPM-GPG-KEY-redhat-release
BaseOS     EULA  GPL              isolinux  RPM-GPG-KEY-redhat-beta  TRANS.TBL
[root@localhost dvd]# yum repolist
Updating Subscription Management repositories.
Unable to read consumer identity
This system is not registered to Red Hat Subscription Management. You can use subscription-manager t
o register.
Repository docker-ce-edge is listed more than once in the configuration
Repository docker-ce-edge-debuginfo is listed more than once in the configuration
Repository docker-ce-edge-source is listed more than once in the configuration
Repository docker-ce-nightly is listed more than once in the configuration
Repository docker-ce-nightly-debuginfo is listed more than once in the configuration
Repository docker-ce-nightly-source is listed more than once in the configuration
Repository docker-ce-stable is listed more than once in the configuration
Repository docker-ce-stable-debuginfo is listed more than once in the configuration
Repository docker-ce-stable-source is listed more than once in the configuration
Repository docker-ce-test is listed more than once in the configuration
Repository docker-ce-test-debuginfo is listed more than once in the configuration
Repository docker-ce-test-source is listed more than once in the configuration
Last metadata expiration check: 0:55:10 ago on Sat 01 Aug 2020 03:30:10 AM IST.
repo id                                repo name                                              status
AppStream                              Local_appstream                                        4,672
BaseOS                                 Local_baseos                                           1,658
docker-ce-stable                       Docker CE Stable - x86_64                                 71
[root@localhost dvd]# _
```

```
[root@localhost /]# cd /etc/yum.repos.d
[root@localhost yum.repos.d]# ls
docer-ce.repo  docker-ce.repo  dvd1.repo  dvd2.repo  redhat.repo
[root@localhost yum.repos.d]# cat dvd1.repo
[BaseOS]
baseurl = file:///dvd/BaseOS
gpgcheck = 0
name = Local_baseos

[root@localhost yum.repos.d]# cat dvd2.repo
[AppStream]
baseurl = file:///dvd/AppStream
gpgcheck = 0
name = Local_appstream

[root@localhost yum.repos.d]# _
```

STEP2: Remove unwanted software if any and downloading the prerequisites software required

```
- name: Remove docker if installed from CentOS repo
  yum:
    name:
     - docker
     - docker-client
     - docker-client-latest
     - docker-common
     - docker-latest
     - docker-latest-logrotate
     - docker-logrotate
     - docker-engine
    state: removed
```

```
- name: Install yum utils
  yum:
    name: yum-utils
    state: latest

- name: Install device-mapper-persistent-data
  yum:
    name: device-mapper-persistent-data
    state: latest

- name: Install lvm2
  yum:
    name: lvm2
    state: latest
```

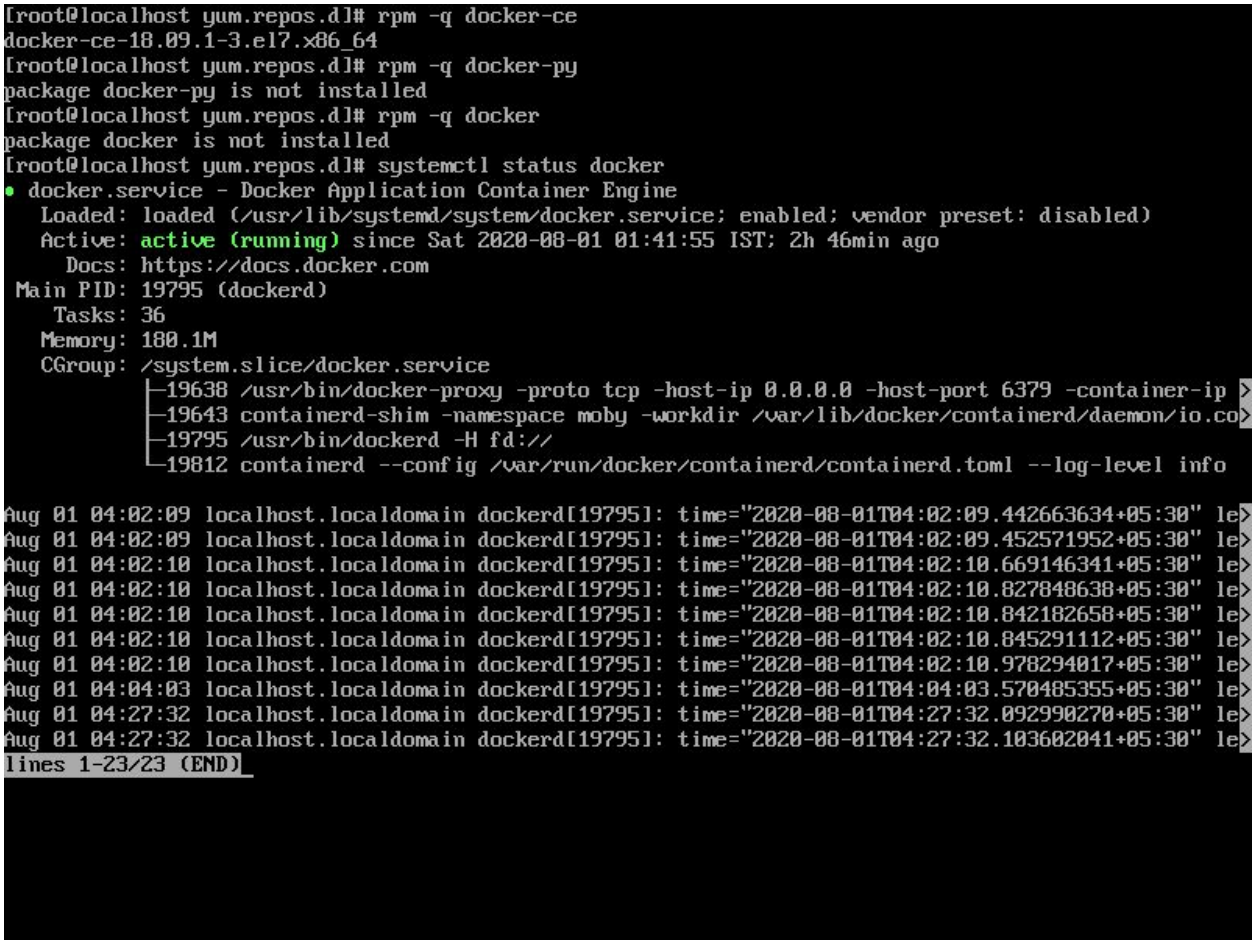
STEP3: creating a docker repo and installing the docker and other dependencies using package module

```
- name: Add Docker repo
  get_url:
     url: https://download.docker.com/linux/centos/docker-ce.repo
     dest: /etc/yum.repos.d/docker-ce.repo
  become: yes

- name: Enable Docker Edge repo
  ini_file:
    dest: /etc/yum.repos.d/docker-ce.repo
    section: 'docker-ce-edge'
    option: enabled
    value: 0
  become: yes

- name: Enable Docker Test repo
  ini_file:
    dest: /etc/yum.repos.d/docker-ce.repo
    section: 'docker-ce-test'
    option: enabled
    value: 0
```

```
  become: yes

- name: Install docker and its dependecies
  package:
    name: "{{ packages }}"
    state: present
    update_cache: yes
  vars:
    packages:
    - docker-ce
    - docker-ce-cli
    - containerd.io
  notify:
    - docker status
```

STEP4: using service module starting docker service and disabling firewalld

```
- name: Start Docker service
  service:
    name: docker
    state: started
    enabled: yes
  become: yes

- name: docker-py
  shell: "pip3 install docker"

- firewalld:
    permanent: yes
    state: disabled
```



STEP5: Installing other packages

```
- name: Install packages
  package:
   name: "{{ packages }}"
   state: present
   update_cache: yes
  vars:
   packages:
   - ca-certificates
   - curl
   - wget
```

STEP6: Adding the kubernetes yum repo and adding gpgKey and installing kubeadm, kubectl and kubelet

```
- name: Adding yum repository for Kubernetes
```

```
    yum_repository:
      baseurl:
https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64
      name: KUBERNETES
      state: present
      gpgcheck: yes
      repo_gpgcheck: yes
      description: Kubernetes
      gpgkey: https://packages.cloud.google.com/yum/doc/yum-key.gpg
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
      enabled: yes

  - name: Install Kubernetes binaries
    package:
      name: "{{ packages }}"
      state: present
      update_cache: yes
    vars:
      packages:
       - kubelet
       - kubeadm
       - kubectl
```

```
[root@localhost ~]# systemctl status firewalld
● firewalld.service - firewalld - dynamic firewall daemon
   Loaded: loaded (/usr/lib/systemd/system/firewalld.service; disabled; vendor preset: enabled)
   Active: inactive (dead)
     Docs: man:firewalld(1)
[root@localhost ~]# cd /etc/yum.repos.d
[root@localhost yum.repos.d]# ls
docer-ce.repo  docker-ce.repo  dvd1.repo  dvd2.repo  KUBERNETES.repo  redhat.repo
[root@localhost yum.repos.d]# cat KUBERNETES.repo
[KUBERNETES]
baseurl = https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64
enabled = 1
gpgcheck = 1
gpgkey = https://packages.cloud.google.com/yum/doc/yum-key.gpg https://packages.cloud.google.com/yum
/doc/rpm-package-key.gpg
name = Kubernetes
repo_gpgcheck = 1

[root@localhost yum.repos.d]#
```

STEP7: Disabling the selinux service an copying the docker daemon file , hence restarting the docker services

- name: Put SELinux in permissive mode, logging actions that would be blocked.
selinux:
policy: targeted
state: permissive

- name: copy daemon file
copy:
src: daemon.json
dest: /etc/docker/daemon.json

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=permissive
# SELINUXTYPE= can take one of these three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=targeted


~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"/etc/selinux/config" 14L, 549C
```

STEP8: Removing the fstab file and restarting the docker services , downloading the iproute-tc package and starting the kubelet service

```
- name: Remove swapfile from /etc/fstab
  mount:
    name: "{{ item }}"
    fstype: swap
    state: absent
  with_items:
    - swap
    - none


- name: Restart Docker service
  service:
    name: docker
```

```
    state: restarted
    enabled: yes
  become: yes

- name: Install iproute-tc
  package:
    name: iproute-tc
    state: present
    update_cache: yes

- name: start kubelet
  service:
    name: kubelet
    daemon_reload: yes
    state: started
    enabled: yes
```

```
#
# /etc/fstab
# Created by anaconda on Tue Jul 28 07:23:21 2020
#
# Accessible filesystems, by reference, are maintained under '/dev/disk/'.
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info.
#
# After editing this file, run 'systemctl daemon-reload' to update systemd
# units generated from this file.
#
/dev/mapper/rhel-root   /                       xfs     defaults        0 0
UUID=4af011bf-6a6e-4aee-b32c-70dbec1e4589 /boot                   xfs     defaults        0 0
/dev/cdrom /root/dvd1 iso9660 ro,noauto 0 0
/dev/cdrom /dvd/ iso9660 ro,loop 0 0
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"/etc/fstab" 15L, 584C
```

STEP9: Initializing the kubeadm by passing the range , setting up the kubeconfig for home user and finaling storing the token in a file

```
  - name: Initialize the Kubernetes cluster using kubeadm
    shell: "kubeadm init --node-name k8s-master --pod-network-cidr=10.10.1.0/16"

   - name: Setup kubeconfig for home user
   command: "{{ item }}"
   with_items:
    - sudo mkdir -p $home/.kube
    - sudo cp -i /etc/kubernetes/admin.conf $home/.kube/config
    - sudo chown $(id -u):$(id -g) $home/.kube/config

  - name: Install calico pod network
    become: false
    command: kubectl create -f
https://raw.githubusercontent.com/coreos/flannel/master/Documentation/kube-fla
nnel.yml


  - name: Generate join command
    command: kubeadm token create --print-join-command
    register: join_command

  - name: Copy join command to local file
     local_action: copy content="{{ join_command.stdout_lines[0] }}"
dest="./join-command"
```

```
localhost login: root
Password:
Last login: Sun Aug  2 19:00:55 from 192.168.43.243
l[root@localhost ~]# lscpu
Architecture:        x86_64
CPU op-mode(s):      32-bit, 64-bit
Byte Order:          Little Endian
CPU(s):              2
On-line CPU(s) list: 0,1
Thread(s) per core:  1
Core(s) per socket:  2
Socket(s):           1
NUMA node(s):        1
Vendor ID:           GenuineIntel
CPU family:          6
Model:               69
Model name:          Intel(R) Core(TM) i5-4210U CPU @ 1.70GHz
Stepping:            1
CPU MHz:             1696.076
BogoMIPS:            3392.15
Hypervisor vendor:   KVM
Virtualization type: full
L1d cache:           32K
L1i cache:           32K
L2 cache:            256K
L3 cache:            3072K
NUMA node0 CPU(s):   0,1
Flags:               fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush
 mmx fxsr sse sse2 ht syscall nx rdtscp lm constant_tsc rep_good nopl xtopology nonstop_tsc cpuid ts
c_known_freq pni pclmulqdq ssse3 cx16 pcid sse4_1 sse4_2 x2apic movbe popcnt aes xsave avx rdrand hy
pervisor lahf_lm abm invpcid_single pti fsgsbase avx2 invpcid flush_l1d
[root@localhost ~]# kubeadm token create --print-join-command
W0802 19:16:29.671447    4991 configset.go:202] WARNING: kubeadm cannot validate component configs f
or API groups [kubelet.config.k8s.io kubeproxy.config.k8s.io]
kubeadm join 192.168.43.234:6443 --token igmss7.fcpjj68sxkge8crc     --discovery-token-ca-cert-hash
sha256:3aff389119882965b147cde666b5e1abd05caed509c8b8172178e31ca56e710c
[root@localhost ~]# _
```

SLAVE_NODE

STEP1: configuring yum in the master node , creating a directory and mounting the software into it and creating dvd1.repo and dvd2.repo

```
- name: kubernetes_setup
  become: true
  gather_facts: No
  hosts:  server1
  tasks:
  - name: create_dvd_folder
    file:
      path:  /root/dvd
      state:  directory
      mode:  "0755"
```

```
- name: mount_dvd
  mount:
    path: /dvd/
    src: /dev/cdrom
    fstype: iso9660
    opts: ro,loop
    state: mounted

- name: yum_repo_BaseOS
  yum_repository:
    name: BaseOS
    description: Local_baseos
    file: dvd1
    baseurl: file:///dvd/BaseOS
    gpgcheck: no

- name: yum_repo_AppStream
  yum_repository:
    name: AppStream
    description: Local_appstream
    file: dvd2
    baseurl: file:///dvd/AppStream
    gpgcheck: no
```

STEP2: Remove unwanted software if any and downloading the prerequisites software required

```
- name: Remove docker if installed from CentOS repo
  yum:
    name:
      - docker
      - docker-client
      - docker-client-latest
      - docker-common
      - docker-latest
      - docker-latest-logrotate
      - docker-logrotate
      - docker-engine
```

```
    state: removed

- name: Install yum utils
  yum:
    name: yum-utils
    state: latest

- name: Install device-mapper-persistent-data
  yum:
    name: device-mapper-persistent-data
    state: latest

- name: Install lvm2
  yum:
    name: lvm2
    state: latest
```

STEP3: creating a docker repo and installing the docker and other dependencies using package module

```
- name: Add Docker repo
  get_url:
     url: https://download.docker.com/linux/centos/docker-ce.repo
     dest: /etc/yum.repos.d/docker-ce.repo
  become: yes

- name: Enable Docker Edge repo
  ini_file:
    dest: /etc/yum.repos.d/docker-ce.repo
    section: 'docker-ce-edge'
    option: enabled
    value: 0
  become: yes

- name: Enable Docker Test repo
  ini_file:
    dest: /etc/yum.repos.d/docker-ce.repo
    section: 'docker-ce-test'
```

```
    option: enabled
    value: 0
  become: yes

- name: Install docker and its dependecies
  package:
    name: "{{ packages }}"
    state: present
    update_cache: yes
  vars:
    packages:
    - docker-ce
    - docker-ce-cli
    - containerd.io
  notify:
    - docker status
```

STEP4: using service module starting docker service and disabling firewalld

```
- name: Start Docker service
  service:
    name: docker
    state: started
    enabled: yes
  become: yes
- name: docker-py
  shell: "pip3 install docker"

- firewalld:
    permanent: yes
    state: disabled
```

STEP5: Installing other packages

```
- name: Install packages
  package:
   name: "{{ packages }}"
```

```
    state: present
    update_cache: yes
   vars:
    packages:
    - ca-certificates
    - curl
    - wget
```

STEP6: Adding the kubernetes yum repo and adding gpgKey and installing kubeadm, kubectl and kubelet

```
  - name: Adding yum repository for Kubernetes
    yum_repository:
      baseurl:
https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64
      name: KUBERNETES
      state: present
      gpgcheck: yes
      repo_gpgcheck: yes
      description: Kubernetes
      gpgkey: https://packages.cloud.google.com/yum/doc/yum-key.gpg
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
      enabled: yes

  - name: Install Kubernetes binaries
    package:
      name: "{{ packages }}"
      state: present
      update_cache: yes
    vars:
      packages:
       - kubelet
       - kubeadm
       - kubectl
```

STEP7: Disabling the selinux service an copying the docker daemon file , hence restarting the docker services

```
  - name: Put SELinux in permissive mode, logging actions that would be
blocked.
    selinux:
      policy: targeted
      state: permissive

  - name: copy daemon file
    copy:
      src: daemon.json
      dest: /etc/docker/daemon.json
```

STEP8: Removing the fstab file and restarting the docker services , downloading the iproute-tc package and starting the kubelet service

```
  - name: Remove swapfile from /etc/fstab
    mount:
      name: "{{ item }}"
      fstype: swap
      state: absent
    with_items:
      - swap
      - none

  - name: Restart Docker service
    service:
      name: docker
      state: restarted
      enabled: yes
    become: yes

  - name: Install iproute-tc
    package:
      name: iproute-tc
      state: present
      update_cache: yes

  - name: start kubelet
    service:
      name: kubelet
```

daemon_reload: yes
state: started
enabled: yes

STEP9 : Join the nodes to the Kubernetes cluster using below code.

- name: Copy the join command to server location
copy: src=join-command
dest=/tmp/join-command.sh
mode="0777"

- name: Join the node to cluster
shell: "sh /tmp/join-command.sh"