```
+ ansible-playbook -i files/hosts init.yaml

PLAY [master] ******************************************************************

TASK [Gathering Facts] *********************************************************
ok: [k8s]

TASK [initialize the cluster] **************************************************
changed: [k8s]

TASK [debug] *******************************************************************
ok: [k8s] => {
    "kubeadm.stdout_lines": [
        "[init] Using Kubernetes version: v1.26.0",
        "[preflight] Running pre-flight checks",
        "[preflight] Pulling images required for setting up a Kubernetes cluster",
        "[preflight] This might take a minute or two, depending on the speed of your internet connection",
        "[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'",
        "[certs] Using certificateDir folder \"/etc/kubernetes/pki\"",
        "[certs] Generating \"ca\" certificate and key",
        "[certs] Generating \"apiserver\" certificate and key",
        "[certs] apiserver serving cert is signed for DNS names [k8s kubernetes kubernetes.default kubernetes.default.svc
kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 192.168.1.4]",
        "[certs] Generating \"apiserver-kubelet-client\" certificate and key",
        "[certs] Generating \"front-proxy-ca\" certificate and key",
        "[certs] Generating \"front-proxy-client\" certificate and key",
        "[certs] Generating \"etcd/ca\" certificate and key",
        "[certs] Generating \"etcd/server\" certificate and key",
        "[certs] etcd/server serving cert is signed for DNS names [k8s localhost] and IPs [192.168.1.4 127.0.0.1 ::1]",
        "[certs] Generating \"etcd/peer\" certificate and key",
        "[certs] etcd/peer serving cert is signed for DNS names [k8s localhost] and IPs [192.168.1.4 127.0.0.1 ::1]",
        "[certs] Generating \"etcd/healthcheck-client\" certificate and key",
        "[certs] Generating \"apiserver-etcd-client\" certificate and key",
        "[certs] Generating \"sa\" key and public key",
        "[kubeconfig] Using kubeconfig folder \"/etc/kubernetes\"",
        "[kubeconfig] Writing \"admin.conf\" kubeconfig file",
        "[kubeconfig] Writing \"kubelet.conf\" kubeconfig file",
        "[kubeconfig] Writing \"controller-manager.conf\" kubeconfig file",
        "[kubeconfig] Writing \"scheduler.conf\" kubeconfig file",
        "[kubelet-start] Writing kubelet environment file with flags to file \"/var/lib/kubelet/kubeadm-flags.env\"",
        "[kubelet-start] Writing kubelet configuration to file \"/var/lib/kubelet/config.yaml\"",
        "[kubelet-start] Starting the kubelet",
        "[control-plane] Using manifest folder \"/etc/kubernetes/manifests\"",
        "[control-plane] Creating static Pod manifest for \"kube-apiserver\"",
        "[control-plane] Creating static Pod manifest for \"kube-controller-manager\"",
        "[control-plane] Creating static Pod manifest for \"kube-scheduler\"",
```

```
        "[etcd] Creating static Pod manifest for local etcd in \"/etc/kubernetes/manifests\"",
        "[wait-control-plane] Waiting for the kubelet to boot up the control plane as static Pods from directory
\"/etc/kubernetes/manifests\". This can take up to 4m0s",
        "[apiclient] All control plane components are healthy after 24.052230 seconds",
        "[upload-config] Storing the configuration used in ConfigMap \"kubeadm-config\" in the \"kube-system\" Namespace",
        "[kubelet] Creating a ConfigMap \"kubelet-config\" in namespace kube-system with the configuration for the kubelets in
the cluster",
        "[upload-certs] Skipping phase. Please see --upload-certs",
        "[mark-control-plane] Marking the node k8s as control-plane by adding the labels: [node-role.kubernetes.io/control-
plane node.kubernetes.io/exclude-from-external-load-balancers]",
        "[mark-control-plane] Marking the node k8s as control-plane by adding the taints [node-role.kubernetes.io/control-
plane:NoSchedule]",
        "[bootstrap-token] Using token: xl3ud5.1sxsyfmgclmya9aw",
        "[bootstrap-token] Configuring bootstrap tokens, cluster-info ConfigMap, RBAC Roles",
        "[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to get nodes",
        "[bootstrap-token] Configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long
term certificate credentials",
        "[bootstrap-token] Configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node
Bootstrap Token",
        "[bootstrap-token] Configured RBAC rules to allow certificate rotation for all node client certificates in the
cluster",
        "[bootstrap-token] Creating the \"cluster-info\" ConfigMap in the \"kube-public\" namespace",
        "[kubelet-finalize] Updating \"/etc/kubernetes/kubelet.conf\" to point to a rotatable kubelet client certificate and
key",
        "[addons] Applied essential addon: CoreDNS",
        "[addons] Applied essential addon: kube-proxy",
        "",
        "Your Kubernetes control-plane has initialized successfully!",
        "",
        "To start using your cluster, you need to run the following as a regular user:",
        "",
        "  mkdir -p $HOME/.kube",
        "  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config",
        "  sudo chown $(id -u):$(id -g) $HOME/.kube/config",
        "",
        "Alternatively, if you are the root user, you can run:",
        "",
        "  export KUBECONFIG=/etc/kubernetes/admin.conf",
        "",
        "You should now deploy a pod network to the cluster.",
        "Run \"kubectl apply -f [podnetwork].yaml\" with one of the options listed at:",
        "  https://kubernetes.io/docs/concepts/cluster-administration/addons/",
        "",
        "Then you can join any number of worker nodes by running the following on each as root:",
        "",
        "kubeadm join 192.168.1.4:6443 --token xl3ud5.1sxsyfmgclmya9aw \\",
        "\t--discovery-token-ca-cert-hash sha256:70115562aeb56cc5453800a6ba0c769f6b49f1762aacaebad96f282be75639ee "
```

```
    ]
}

TASK [k8s-copy-admin : create .kube directory] *********************************
ok: [k8s]

TASK [k8s-copy-admin : copy admin.conf to user's kube config] ******************
changed: [k8s]

PLAY RECAP *********************************************************************
k8s                        : ok=5    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```