OpenShift Singlenode Installation Skript

Installiert ein vollwertiges OpenShift Origin Cluster auf einem single Node (VM)

Skript Sourcecode

Filename: install_openshift.sh

```
#!/bin/bash
################################################################################
################
#**************** OpenShift single Cluster Setup by Michael Reber - v 1.2
******************#
################################################################################
################

################################################################################
################

## Default variables to use
export INTERACTIVE=${INTERACTIVE:="true"}
export PVS=${INTERACTIVE:="true"}
export DOMAIN=${DOMAIN:="blackgate.org"}
export USERNAME=${USERNAME:="$(whoami)"}
export PASSWORD=${PASSWORD:="password"}
export ANSIBLE_USERNAME=${ANSIBLE_USERNAME:="$(whoami)"}
export VERSION=${VERSION:="3.11"}
export IP=${IP:="$(ip route get 8.8.8.8 | awk '{print $NF; exit}')"}
export API_PORT=${API_PORT:="443"}
export LETSENCRYPT=${LETSENCRYPT:="false"}
export MAIL=${MAIL:="example@blackgate.org"}

## Make the script interactive to set the variables
if [ "$INTERACTIVE" = "true" ]; then
        read -rp "Domain to use: ($DOMAIN): " choice;
        if [ "$choice" != "" ] ; then
                export DOMAIN="$choice";
        fi

        read -rp "OSE-Username: ($USERNAME): " choice;
        if [ "$choice" != "" ] ; then
                export USERNAME="$choice";
        fi

        read -rp "OSE-Password: ($PASSWORD): " choice;
```

```
        if [ "$choice" != "" ] ; then
                export PASSWORD="$choice";
        fi
        read -rp "Ansible-Username: ($ANSIBLE_USERNAME): " choice;
        if [ "$choice" != "" ] ; then
                export ANSIBLE_USERNAME="$choice";
        fi
        read -rp "OpenShift Version: ($VERSION): " choice;
        if [ "$choice" != "" ] ; then
                export VERSION="$choice";
        fi
        read -rp "IP: ($IP): " choice;
        if [ "$choice" != "" ] ; then
                export IP="$choice";
        fi

        read -rp "API Port: ($API_PORT): " choice;
        if [ "$choice" != "" ] ; then
                export API_PORT="$choice";
        fi

        echo "Do you wish to enable HTTPS with Let's Encrypt?"
        echo "Warnings: "
        echo "  Let's Encrypt only works if the IP is using publicly
accessible IP and custom certificates."
        echo "  This feature doesn't work with OpenShift CLI for now."
        select yn in "Yes" "No"; do
                case $yn in
                        Yes) export LETSENCRYPT=true; break;;
                        No) export LETSENCRYPT=false; break;;
                        *) echo "Please select Yes or No.";;
                esac
        done

        if [ "$LETSENCRYPT" = true ] ; then
                read -rp "Email(required for Let's Encrypt): ($MAIL): "
choice;
                if [ "$choice" != "" ] ; then
                        export MAIL="$choice";
                fi
        fi

        echo

fi

export METRICS="True"
export LOGGING="True"
```

```
memory=$(sudo cat /proc/meminfo | grep MemTotal | sed "s/MemTotal:[
]*\([0-9]*\) kB/\1/")

if [ "$memory" -lt "4194304" ]; then
        export METRICS="False"
fi

if [ "$memory" -lt "16777216" ]; then
        export LOGGING="False"
fi

echo "******"
echo "* Your domain is $DOMAIN "
echo "* Your IP is $IP "
echo "* Your username is $USERNAME "
echo "* Your password is $PASSWORD "
echo "* OpenShift version: $VERSION "
echo "* Enable HTTPS with Let's Encrypt: $LETSENCRYPT "
if [ "$LETSENCRYPT" = true ] ; then
        echo "* Your email is $MAIL "
fi
echo "******"

#-----------------------------------------------START-INVENTORY-------------
-------------------------------
cat <<EOF1 > inventory.ini
[OSEv3:children]
masters
nodes
etcd

[masters]
${IP} openshift_ip=${IP} openshift_schedulable=true

[etcd]
${IP} openshift_ip=${IP}

[nodes]
${IP} openshift_ip=${IP} openshift_schedulable=true
openshift_node_group_name="node-config-all-in-one"

[OSEv3:vars]
openshift_additional_repos=[{'id': 'centos-paas', 'name': 'centos-paas',
'baseurl'
:'https://buildlogs.centos.org/centos/7/paas/x86_64/openshift-origin311',
'gpgcheck' :'0', 'enabled' :'1'}]

ansible_ssh_user=${ANSIBLE_USERNAME}

# If ansible_ssh_user is not root, ansible_become must be set to true
ansible_become=true
```

```
enable_excluders=False
enable_docker_excluder=False
ansible_service_broker_install=False

containerized=True
os_sdn_network_plugin_name='redhat/openshift-ovs-multitenant'
openshift_disable_check=disk_availability,docker_storage,memory_availability
,docker_image_availability

deployment_type=origin
openshift_deployment_type=origin

template_service_broker_selector={"region":"infra"}
openshift_metrics_image_version="v${VERSION}"
openshift_logging_image_version="v${VERSION}"
openshift_logging_elasticsearch_proxy_image_version="v1.0.0"
openshift_logging_es_nodeselector={"node-role.kubernetes.io/infra":"true"}
logging_elasticsearch_rollout_override=false
osm_use_cockpit=true

openshift_metrics_install_metrics=${METRICS}
openshift_logging_install_logging=${LOGGING}

openshift_master_identity_providers=[{'name': 'htpasswd_auth', 'login':
'true', 'challenge': 'true', 'kind': 'HTPasswdPasswordIdentityProvider'}]
openshift_master_htpasswd_users={"${USERNAME}": "$(openssl passwd -apr1
${PASSWORD})"}

openshift_master_api_port=${API_PORT}
openshift_public_hostname=console.${DOMAIN}
openshift_master_cluster_hostname=$(hostname)
openshift_master_default_subdomain=apps.${DOMAIN}
openshift_master_console_port=${API_PORT}

#openshift_master_logout_url=https://example.com/logout
EOF1

#-------------------------------------------END-INVENTORY---------------
----------------------------


# install updates
sudo yum update -y

# install the following base packages
sudo yum install -y  wget git zile nano net-tools docker-1.13.1\
                                bind-utils iptables-services \
                                bridge-utils bash-completion \
                                kexec-tools sos psacct openssl-devel \
```

```
                                httpd-tools NetworkManager \
                                python-cryptography python2-pip python-devel
python-passlib \
                                java-1.8.0-openjdk-headless "@Development
Tools"

#install epel
sudo yum -y install epel-release

# Disable the EPEL repository globally so that is not accidentally used
during later steps of the installation
sudo sed -i -e "s/^enabled=1/enabled=0/" /etc/yum.repos.d/epel.repo

sudo systemctl | grep "NetworkManager.*running"
if [ $? -eq 1 ]; then
        sudo systemctl start NetworkManager
        sudo systemctl enable NetworkManager
fi

# install the packages for Ansible
sudo yum -y --enablerepo=epel install pyOpenSSL

curl -o ansible.rpm
https://releases.ansible.com/ansible/rpm/release/epel-7-x86_64/ansible-2.6.5
-1.el7.ans.noarch.rpm
sudo yum -y --enablerepo=epel install ansible.rpm

[ ! -d openshift-ansible ] && git clone
https://github.com/openshift/openshift-ansible.git

cd openshift-ansible && git fetch && git checkout release-${VERSION} && cd
..

sudo bash -c "cat <<EOF2 > /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4
localhost4.localdomain4
::1         localhost localhost.localdomain localhost6
localhost6.localdomain6
${IP}           $(hostname) console console.${DOMAIN}
EOF2"

if [ -z $DISK ]; then
        echo "Not setting the Docker storage."
else
        sudo cp /etc/sysconfig/docker-storage-setup /etc/sysconfig/docker-
storage-setup.bk

        sudo echo DEVS=$DISK > /etc/sysconfig/docker-storage-setup
        sudo echo VG=DOCKER >> /etc/sysconfig/docker-storage-setup
        sudo echo SETUP_LVM_THIN_POOL=yes >> /etc/sysconfig/docker-storage-
setup
```

```
        sudo echo DATA_SIZE="100%FREE" >> /etc/sysconfig/docker-storage-
setup

        sudo systemctl stop docker

        sudo rm -rf /var/lib/docker
        sudo wipefs --all $DISK
#       docker-storage-setup
        container-storage-setup

        systemctl restart docker
fi

#sudo systemctl restart docker
sudo systemctl enable docker

# add proxy in inventory.ini if proxy variables are set
if [ ! -z "${HTTPS_PROXY:-${https_proxy:-${HTTP_PROXY:-${http_proxy}}}}" ];
then
        echo >> inventory.ini
        echo "openshift_http_proxy=\"${HTTP_PROXY:-${http_proxy:-
${HTTPS_PROXY:-${https_proxy}}}}\"" >> inventory.ini
        echo "openshift_https_proxy=\"${HTTPS_PROXY:-${https_proxy:-
${HTTP_PROXY:-${http_proxy}}}}\"" >> inventory.ini
        if [ ! -z "${NO_PROXY:-${no_proxy}}" ]; then
                __no_proxy="${NO_PROXY:-${no_proxy}},${IP},.${DOMAIN}"
        else
                __no_proxy="${IP},.${DOMAIN}"
        fi
        echo "openshift_no_proxy=\"${__no_proxy}\"" >> inventory.ini
fi

# Let's Encrypt setup
if [ "$LETSENCRYPT" = true ] ; then
        # Install CertBot
        sudo yum install --enablerepo=epel -y certbot

        # Configure Let's Encrypt certificate
        certbot certonly --manual \
                        --preferred-challenges dns \
                        --email $MAIL \
                        --server
https://acme-v02.api.letsencrypt.org/directory \
                        --agree-tos \
                        -d $DOMAIN \
                        -d *.$DOMAIN \
                        -d *.apps.$DOMAIN

        ## Modify inventory.ini
```

```
        # Declare usage of Custom Certificate
        # Configure Custom Certificates for the Web Console or CLI =>
Doesn't Work for CLI
        # Configure a Custom Master Host Certificate
        # Configure a Custom Wildcard Certificate for the Default Router
        # Configure a Custom Certificate for the Image Registry
        ## See here for more explanation:
https://docs.okd.io/latest/install_config/certificate_customization.html
        cat <<EOF3 >> inventory.ini

        openshift_master_overwrite_named_certificates=true

        openshift_master_cluster_hostname=console-internal.${DOMAIN}
        openshift_master_cluster_public_hostname=console.${DOMAIN}

        openshift_master_named_certificates=[{"certfile":
"/etc/letsencrypt/live/${DOMAIN}/fullchain.pem", "keyfile":
"/etc/letsencrypt/live/${DOMAIN}/privkey.pem", "cafile":
"/etc/letsencrypt/live/${DOMAIN}/chain.pem", "names":
["console.${DOMAIN}"]}]

        openshift_hosted_router_certificate={"certfile":
"/etc/letsencrypt/live/${DOMAIN}/fullchain.pem", "keyfile":
"/etc/letsencrypt/live/${DOMAIN}/privkey.pem", "cafile":
"/etc/letsencrypt/live/${DOMAIN}/chain.pem"}

        openshift_hosted_registry_routehost=registry.apps.${DOMAIN}
        openshift_hosted_registry_routecertificates={"certfile":
"/etc/letsencrypt/live/${DOMAIN}/fullchain.pem", "keyfile":
"/etc/letsencrypt/live/${DOMAIN}/privkey.pem", "cafile":
"/etc/letsencrypt/live/${DOMAIN}/chain.pem"}
        openshift_hosted_registry_routetermination=reencrypt
EOF3

        # Add Cron Task to renew certificate
        echo "@weekly  certbot renew --pre-hook=\"oc scale --replicas=0 dc
router\" --post-hook=\"oc scale --replicas=1 dc router\"" > certbotcron
        crontab certbotcron
        rm certbotcron
fi

sudo mkdir -p /etc/origin/master/
#sudo touch /etc/origin/master/htpasswd
#sudo htpasswd -b /etc/origin/master/htpasswd ${USERNAME} ${PASSWORD}

ansible-playbook -i inventory.ini openshift-
ansible/playbooks/prerequisites.yml
ansible-playbook -i inventory.ini openshift-
ansible/playbooks/deploy_cluster.yml

sudo oc adm policy add-cluster-role-to-user cluster-admin ${USERNAME}
```

```
if [ "$PVS" = "true" ]; then

        cat <<EOF4 > vol.yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: vol
spec:
  capacity:
    storage: 500Gi
  accessModes:
    - ReadWriteOnce
    - ReadWriteMany
  persistentVolumeReclaimPolicy: Retain
  hostPath:
    path: /mnt/data/vol
EOF4

        for i in `seq 1 200`;
        do
                DIRNAME="vol$i"
                sudo mkdir -p /mnt/data/$DIRNAME
                sudo chcon -Rt svirt_sandbox_file_t /mnt/data/$DIRNAME
                sudo chmod 777 /mnt/data/$DIRNAME

                sed "s/name: vol/name: vol$i/g" vol.yaml > oc_vol.yaml
                sed -i "s/path: \/mnt\/data\/vol/path: \/mnt\/data\/vol$i/g"
oc_vol.yaml
                sudo oc create -f oc_vol.yaml
                echo "created volume $i"
        done
        sudo rm oc_vol.yaml
fi

echo "******"
echo "* Your console is https://console.$DOMAIN:$API_PORT"
echo "* Your username is $USERNAME "
echo "* Your password is $PASSWORD "
echo "*"
echo "* Login using:"
echo "*"
echo "$ oc login -u ${USERNAME} -p ${PASSWORD}
https://console.$DOMAIN:$API_PORT/"
echo "******"

sudo oc login -u ${USERNAME} -p ${PASSWORD}
https://console.$DOMAIN:$API_PORT/
```

Last update: **2019/04/04 16:33**