Daily Business mit OpenShift

Cluster User Administration

Cluster-Admins verwalten

Um einem Benutzer cluster-admin Rechte auf der Plattform zu vergeben, müssen folgende Schritte beachtet werden:

1. Einloggen mit einem Benutzer, welcher bereits cluster-admin Rechte besitzt
2. Auflisten der berechtigten Benutzer der Rolle cluster-admin

```
# oc get clusterrolebinding | grep cluster-admin
```

```
...
cluster-admin           /cluster-admin                  [cluster-admin]
system:masters
...
```

3. Benutzer berechtigen

```
# oc adm policy add-cluster-role-to-group cluster-admin [group]
```

- Benutzer wieder entfernen

```
# oc adm policy remove-cluster-role-to-group cluster-admin [group]
```

Benutzer zu einem Projekt hinzufügen

Es gibt zwei verschiedene Arten, um einen Benutzer an einem Projekt zu berechtigen. Die drei meist verwendeten Berechtigungsvarianten sind folgende: admin, edit, view

Die erste Art wäre um GUI beim entsprechenden Projekt auf “View Membership” zu gehen. Darin können Benutzer mit Ihrem Suffix und der entsprechenden Berechtigung berechtigt werden.Development Platform > DailyBusiness > image2018-1-24_13-18-36.png

Die zweite Art wäre via Command Line den Benutzer auf ein Projekt zu berechtigen:

```
# oc policy -n [Projekt] add-role-to-user [Rolle] [Benutzer]
```

Projekt Berechtigungen anzeigen

Die Berechtigungen können entweder über das GUI unter “View Membership” angezeigt werden oder in der Kommandozeile:

```
# oc get rolebindings -n integration-service
```

Link zur Dokumentation:
https://docs.openshift.com/container-platform/3.3/admin_solutions/user_role_mgmt.html#adding-a-role-to-a-user

Ressourcen von einem Projekt wiederherstellen

Bei der Post läuft täglich um 24:00 ein cronjob, der alle wichtigen Projekt-Ressourcen täglich exportiert:

```
#!/bin/bash
BACKUP_DIR="/var/openshift_backup/"
BACKUP_DIR_WITH_DATE=${BACKUP_DIR}/backup_$(date +%Y%m%d%H%M)

### Remove folders older than 30days
find ${BACKUP_DIR} -maxdepth 1 -ctime +30 -type d -exec rm -rf {} \;

### Setup
mkdir -p $BACKUP_DIR_WITH_DATE

### Project Backup
# Check if executed as OSE system:admin
if [[ "$(oc whoami)" != "system:admin" ]]; then
  echo -n "Trying to log in as system:admin... "
  oc login -u system:admin > /dev/null && echo "done."
fi

# Backup all resources of every project
for project in $(oc get projects --no-headers | awk '{print $1}')
do
    echo -n "Backing up project $project... "
    mkdir -p ${BACKUP_DIR_WITH_DATE}/projects/${project}
    oc export all -o yaml -n ${project} >
${BACKUP_DIR_WITH_DATE}/projects/${project}/project.yaml 2>/dev/null
    oc get rolebindings -o yaml -n ${project} >
${BACKUP_DIR_WITH_DATE}/projects/${project}/rolebindings.yaml 2>/dev/null
    oc get serviceaccount -o yaml --export=true -n ${project} >
${BACKUP_DIR_WITH_DATE}/projects/${project}/serviceaccount.yaml 2>/dev/null
    oc get configmap -o yaml --export=true -n ${project} >
${BACKUP_DIR_WITH_DATE}/projects/${project}/configmap.yaml 2>/dev/null
    oc get daemonset -o yaml --export=true -n ${project} >
${BACKUP_DIR_WITH_DATE}/projects/${project}/daemonset.yaml 2>/dev/null
    oc get secret -o yaml --export=true -n ${project} >
```

```
${BACKUP_DIR_WITH_DATE}/projects/${project}/secret.yaml 2>/dev/null
    oc get pvc -o yaml --export=true -n ${project} >
${BACKUP_DIR_WITH_DATE}/projects/${project}/pvc.yaml 2>/dev/null
    oc get statefulset -o yaml --export=true -n ${project} >
${BACKUP_DIR_WITH_DATE}/projects/${project}/statefulset.yaml 2>/dev/null
    oc get buildconfigs -o yaml --export=true -n ${project} >
${BACKUP_DIR_WITH_DATE}/projects/${project}/buildconfigs.yaml 2>/dev/null
    oc get builds -o yaml --export=true -n ${project} >
${BACKUP_DIR_WITH_DATE}/projects/${project}/builds.yaml 2>/dev/null
    oc get imagestreams -o yaml --export=true -n ${project} >
${BACKUP_DIR_WITH_DATE}/projects/${project}/imagestreams.yaml 2>/dev/null
    oc get jobs -o yaml --export=true -n ${project} >
${BACKUP_DIR_WITH_DATE}/projects/${project}/jobs.yaml 2>/dev/null
    oc get route -o yaml --export=true -n ${project} >
${BACKUP_DIR_WITH_DATE}/projects/${project}/route.yaml 2>/dev/null
    oc get services -o yaml --export=true -n ${project} >
${BACKUP_DIR_WITH_DATE}/projects/${project}/services.yaml 2>/dev/null
    echo "done."
done
```

Dieser legt die Files in folgender Struktur auf dem ersten Master ab:

```
/var/openshift_backup/
├── backup_201806190000
│   └── projects
│       ├── appuio-infra
│       ├── default
│       ├── glusterfs
│       ├── kube-public
│       ├── kube-service-catalog
│       ├── kube-system
│       ├── logging
│       ├── management-infra
│       ├── ocp-grafana
│       ├── openshift
│       ├── openshift-ansible-service-broker
│       ├── openshift-infra
│       ├── openshift-metrics
│       ├── openshift-node
│       ├── openshift-template-service-broker
│       ├── openshift-web-console
│       └── patricks-testprojekt
├── backup_201806200000
│   └── projects
│       ├── appuio-infra
│       ├── default
│       ├── glusterfs
│       ├── kube-public
```

```
│       ├── kube-service-catalog
│       ├── kube-system
│       ├── logging
.....
```

Restore erfolg durch ein simples erstellen der Ressourcen. Zum Beispiel:

```
# oc create -f "Backup file in yaml" -n "namespace"
```

Problem OpenShift GlusterFS (nicht mehr im HP OVO)

Das Überwachungsskript(/root/openshift-script/check_gluster_volumes.sh) hat folgende E-Mail in die Mailbox SMSOPENSHIFTDL@post.ch gesendet. Dieselbe Fehlermeldung wird auch im HP-OVO Monitoring angezeigt.

Development Platform > DailyBusiness > 1.png

Auf dem Glusternode vekq7k wurde wegen eines anderen Fehler ein Neustart des Servers gemacht. Anschliessend konnte GlusterFS nicht vollständig gestartet werden. Somit muss das GlusterFS auf auf dem gluster-pv101 forciert neu gestartet werden:

[root@vekq7k ~]# gluster volume status gluster-pv101 Status of volume: gluster-pv101 Gluster process TCP Port RDMA Port Online Pid

---

Brick vhmsgj.pnet.ch:/data/gluster-pv101/br ick 49152 0 Y 6846 Brick vkyg7l.pnet.ch:/data/gluster-pv101/br ick 49152 0 Y 4553 Brick vsytjw.pnet.ch:/data/gluster-pv101/br ick 49152 0 Y 4939 Brick vu32g5.pnet.ch:/data/gluster-pv101/br ick 49152 0 Y 6774 Brick vpva0p.pnet.ch:/data/gluster-pv101/br ick 49152 0 Y 8864 Brick vekq7k.pnet.ch:/data/gluster-pv101/br ick 49202 0 Y 82665 Brick vlf5eh.pnet.ch:/data/gluster-pv101/br ick 49152 0 Y 103339 Brick vw7w5k.pnet.ch:/data/gluster-pv101/br ick 49152 0 Y 6902 Brick v2o8fv.pnet.ch:/data/gluster-pv101/br ick 49152 0 Y 3353 Self-heal Daemon on localhost N/A N/A Y 82687 Self-heal Daemon on v2o8fv.pnet.ch N/A N/A Y 42526 Self-heal Daemon on vs3tvs.pnet.ch N/A N/A Y 99276 Self-heal Daemon on vw7w5k.pnet.ch N/A N/A Y 18084 Self-heal Daemon on vsytjw.pnet.ch N/A N/A Y 4673 Self-heal Daemon on vhmsgj.pnet.ch N/A N/A Y 31894 Self-heal Daemon on vlf5eh.pnet.ch N/A N/A Y 22477 Self-heal Daemon on vkyg7l.pnet.ch N/A N/A Y 57012 Self-heal Daemon on vpva0p.pnet.ch N/A N/A Y 37754 Self-heal Daemon on vu32g5.pnet.ch N/A N/A Y 85923 Self-heal Daemon on vthbcn.pnet.ch N/A N/A Y 117650

Task Status of Volume gluster-pv101

There are no active volume tasks

[root@vekq7k ~]# gluster volume start gluster-pv101 force

Kube Config wiederherstellen

Funktioniert der `oc get` command nicht mehr korrekt, oder besser gesagt wird hier ein Passwort verlangt, so ist die Kube Config verschossen.. Ein gültiges Backup befindet sich hier: `/etc/origin/master/admin.kubeconfig`

```
# cp /etc/origin/master/admin.kubeconfig .kube/config
```

OpenShift Gluster Volume extend - over heketi

Eine zusätzliche Terabyte Disk auf dem Gluster Cluster via Heketi einbinden: (siehe auch : https://access.redhat.com/solutions/3164841)

Login auf dem OpenShift GUI, Project “glusterfs”, Pod “heketi-storage”.

Zuerst die Admin Credientials über das YAML-File vom Pod “heketi-storage” auslesen.

Development Platform > DailyBusiness > image2019-1-25_14-29-55.png

Auf der Heketi-Pod Konsole node und Cluster ID auslesen.

```
sh-4.2# heketi-cli node list --user admin --secret
p/J+OhZgDjwuMy/vZBoD74hP4CWmjWWE32ye6czr2aU=
Id:0680dabe91ee5a7f36da8cb6fe49cdd4 Cluster:7234de4476a10cb0d138e3fd3d387c40
Id:648f2115e99bf41fb78271acd55bd8f9 Cluster:7234de4476a10cb0d138e3fd3d387c40
Id:b016e52ee7378debc04427385f81cd82 Cluster:7234de4476a10cb0d138e3fd3d387c40
```

Jetzt kann das neue Device nach dem Schema “`hecketi-cli device add –name /dev/DISK –node NODE_ID`” für jede Node-ID des Clusters auf der Pod-Konsole eingegeben werden.

```
sh-4.2# heketi-cli device add --name /dev/sdc --node
0680dabe91ee5a7f36da8cb6fe49cdd4 --user admin --secret
p/J+OhZgDjwuMy/vZBoD74hP4CWmjWWE32ye6czr2aU=
Device added successfully

sh-4.2# heketi-cli device add --name /dev/sdc --node
648f2115e99bf41fb78271acd55bd8f9 --user admin --secret
p/J+OhZgDjwuMy/vZBoD74hP4CWmjWWE32ye6czr2aU=
Device added successfully

sh-4.2# heketi-cli device add --name /dev/sdc --node
b016e52ee7378debc04427385f81cd82 --user admin --secret
p/J+OhZgDjwuMy/vZBoD74hP4CWmjWWE32ye6czr2aU=
```

```
Device added successfully
```

Als Resultat der Erweiterung, sieht man nun auf dem entsprechenden Node mit dem Kommando “pvscan” die von heketi neu bereitgestellte Volumegroup.

```
[root@vosge1 ~]# pvscan
  PV /dev/sdc    VG vg_5e4ce3cca0b61f43f749d3cef0447e2e   lvm2 [999.87 GiB /
<979.72 GiB free]
  PV /dev/sdb    VG vg_bc2782bf157d2cde474ff55ae298715f   lvm2 [999.87 GiB /
5.40 GiB free]
  PV /dev/sda2   VG vgsys                                 lvm2 [<79.47 GiB /
0    free]
  Total: 3 [2.03 TiB] / in use: 3 [2.03 TiB] / in no VG: 0 [0   ]
```

Neuer Host zu existierenden Cluster hinzufügen

Red Hat Anleitung:
https://docs.openshift.com/container-platform/3.10/install_config/adding_hosts_to_existing_cluster.html

Bevor der neue Host dem Cluster hinzugefügt werden kann, muss dieser die nötigen OpenShift Pakete erhalten. Hierzu muss das Package “atomic-openshift-utils” installiert werden. Jedoch wird dieses im offiziellen Repo nur bis zur Version 3.9 gepflegt. Deshalb müssen zuerst unter /etc/yum.repos.d/ose.repo die 3.9 OpenShift Repos angehängt werden. Erst danach kann das neuste Paket mit yum installiert werden:

```
[root@new_host]# cat /etc/yum.repos.d/ose.repo
[rhel-7-server-ose310-rpms]
baseurl =
http://vinstp.pnet.ch/distributor/testing/rhel7-server-$basearch/ose/3.10
enabled = 1
gpgcheck = 1
gpgkey = file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
name = Red Hat OpenShift Container Platform 3.10 (RPMs)

[rhel-7-server-ose39-rpms]
baseurl =
http://vinstp.pnet.ch/distributor/testing/rhel7-server-$basearch/ose/3.9
enabled = 1
gpgcheck = 1
gpgkey = file:///etc/pki/rpm-gpg/RPM-GPG-KEY-redhat-release
name = Red Hat OpenShift Container Platform 3.9 (RPMs)
```

```
[root@new_host]# yum install atomic-openshift-utils -y
[root@new_host]# systemctl reboot
```

Der Host ist nun für die Integration bereit. Jetzt muss noch das Hostfile dementsprechend angepasst werden:

```
[rebermi@vosbh1]# cat openshift-hostfiles/hosts_shared.yml
---
all:
  children:
    OSEv3:
      children:
        bastion: {}
        glusterfs: {}
        glusterfs_registry: {}
        etcd: {}
        masters: {}
        nodes:
          children:
            masters: {}
            infra_nodes: {}
            user_nodes: {}
            new_nodes: {}  <---
...
    user_nodes:
      hosts:
        EXISTING_HOSTS.pnet.ch:
          openshift_hostname: vosne1.pnet.ch
          openshift_ip: 172.18.184.14
          openshift_node_group_name: node-config-compute-eh-prod
          openshift_node_local_quota_per_fsgroup: 1Gi
    new_nodes: <---
      hosts:
        NEW_HOST.pnet.ch:
          openshift_hostname: NEW_HOST.pnet.ch
          openshift_ip: IP_ADRESS
          openshift_node_group_name: node-config-compute-eh-test
          openshift_node_local_quota_per_fsgroup: 1Gi
```

Zum Schluss wird nun das Scaleup Playbook von Red Hat ausgeführt:

```
# ansible-playbook [-i /path/to/file] /usr/share/ansible/openshift-
ansible/playbooks/openshift-node/scaleup.yml</sxh>

Damit auch Pods auf dem neuen Cluster-Node erstellt werden, müssen die
LABELS auf dem Node angepasst werden:

<code>[user@vosbh1 ~]$ oc login https://plattform.pnet.ch
```

```
[user@vosbh1 ~]$ oc get nodes --show-labels
```

Man kann jetzt die Labels eines bereits integrierten Node kopieren und folgendermassen beim neuen Node setzen:

```
[user@vosbh1 ~]$ oc label node NEWNODE compute=true
[user@vosbh1 ~]$ oc label node NEWNODE region=primary
[user@vosbh1 ~]$ oc label node NEWNODE stage=test
[user@vosbh1 ~]$ oc label node NEWNODE zone=defaul
```

Everyting below here is depricated!

Gluster PV Administration (obsolete, neu mit heketi)

Gluster PVs erstellen (obsolete)

Um ein neues PV zu erstellen, muss als erstes berücksichtig werden, ob genügen Platz zur Verfügung steht.

Folgender Ansible Befehl kann hierfür verwendet werden:

```
# ansible gluster -m shell -a "vgs | grep vg_gluster"
```

Wenn genügend Platz zur Verfügung steht, muss das Hostfile der jeweiligen Plattform angepasst werden:

```
# cd /root/openshift-hostfiles/
# vim hosts_os***.yaml
```

Anschliessend im Hostfile das neue PV am Ende des Abschnittes “gluster_pvs” einfügen:

```
...
gluster_pvs:
        - name: gluster-pv1
          options:
            performance.write-behind-window-size: 32MB
          size: 100
        - name: gluster-pv2
          options:
            cluster.consistent-metadata: 'on'
            cluster.post-op-delay-secs: '0'
          size: 50
...
        - name: gluster-pv42
```

```
          size: 40
        - name: gluster-pv43
          size: 10
        - name: NEW GLUSTER PW
          size: NEW PV SIZE
...
```

Nun kann das Playbook main.yml im Verzeichnis /root/gluster-pvs/ im Check Modus ausgeführt werden. Damit kann überprüft werden, was das Playbook tatsächlich machen würde, ohne etwas zu ändern (dry run)

```
# cd /root/gluster-pvs/
# ansible-playbook --check main.yml
```

Nachdem der Check durchgelaufen ist (mit Fehler, da die Änderungen nicht gemacht wurden und so das PV so nicht gefunden wurde) kann das Playbook ohne das –check ausgeführt werden:

```
# cd /root/gluster-pvs/
# ansible-playbook main.yml
```

Zum Schluss muss noch überprüft werden, ob das PV tatsächlich angelegt wurde:

```
# oc get pv | grep CREATED GLUSTER-PV
```

Gluster PVs vergrössern (obsolete)

An OpenShift GlusterFS PV can be online resized, increased in size. This needs to be done on the OpenShift master node.

These are the steps:

Edit the Ansible Inventory file and set the new size of the pv. Push the changes also to gitit.pnet.ch

```
# cd /root/openshift-hostfiles/
# vim hosts_osit2.yaml
```

```
...
- name: gluster-pv118
size: 5
...
```

```
# git status
```

```
# On branch master
# Changes not staged for commit:
```

```
#   (use "git add <file>..." to update what will be committed)
#   (use "git checkout -- <file>..." to discard changes in working
directory)
#
#       modified:   hosts_osit2.yaml
#
no changes added to commit (use "git add" and/or "git commit -a")
```

```
# git add hosts_osit2.yaml
# git commit -m "Resize pv118 to 5Gi"
# git push
```

Run /root/gluster-pvs/ansible-playbook –check main.yml. This is a dry run only to check what the changes will be

```
# cd /root/gluster-pvs/
ansible-playbook --check main.yml
```

After successful check run /root/gluster-pvs/ansible-playbook main.yml

```
# cd /root/gluster-pvs/
# ansible-playbook main.yml
```

Check the size of the gluster-pv on the local node. Should now show 1/3rd of the size in mounted position, because GlustrFS is replicated on the node instances.

```
# df -h | grep gluster-pv118
```

Run ov edit pv <pv-name> to have the metadata of the pv volume in consistent state on the master node

```
# oc edit pv gluster-pv118
```

```
...
  capacity:
    storage: 5Gi
  claimRef:
...
```

```
# oc get pv | grep pv118
```

CAUTION: Do not continue if there are errors to see in a step. If there are errors in the Task “[Configure Gluster management firewall rules]“, when running ansible-playbook –check

```
                main.yml, note that during container restart,
                there is a firewall reconfiguration ongoing,
                which may disturb running ansible-playbook.
                Try this check again some moments later.
```

Gluster PV's löschen (obsolete)

Auf dem ersten Master das PV unmounten:

```
# umount /gluster/gluster-pvXX
# df | grep endpoint
```

Das PV im OpenShift entfernen:

```
# oc delete pv gluster-pvXX
```

Das gelöschte PV anschliessend auf einem Gluster Node Stopen und löschen:

```
# gluster volume stop gluster-pvXX
```

```

Stopping volume will make its data inaccessible. Do you want to continue?
(y/n) y
volume stop: gluster-pv11: success
```

```
# gluster volume delete gluster-pvXX
```

```

Deleting volume will erase all information about the volume. Do you want to
continue? (y/n) y
volume delete: gluster-pv11: success
```

Zum schluss folgende Ansible Befehle auf dem Master ausführen:

```
# ansible gluster -m lineinfile -a "path=/etc/fstab state=absent
regexp='^/dev/vg_gluster/gluster-pv11' backup=yes"

# ansible gluster -a "umount /dev/vg_gluster/gluster-pv11"

# ansible gluster -a "wipefs -a /dev/vg_gluster/gluster-pv11"     #(2x
ausführen zum checken)

# ansible gluster -a "lvremove /dev/vg_gluster/gluster-pv11 -y"
```

Last update: **2019/04/04 13:52**