Kubernetes集成GlusterFS集群和Heketi-安装指南与实战 (https://www.kubernetes.org.cn/3893.html)

2018-05-03 17:38 Mingo (https://www.kubernetes.org.cn/author/mingo) 分类：Kubernetes实践分享/开发实战 (https://www.kubernetes.org.cn/practice) / Kubernetes教程/入门教程 (https://www.kubernetes.org.cn/course) 阅读(3610) 评论(0)

本文翻译自heketi的github网址官方文档 (https://github.com/heketi/heketi/blob/master/docs/admin/install-kubernetes.md)（大部分为google翻译，少许人工调整，括号内为个人注解）其中注意事项部分为其他网上查询所得。 本文的整个过程将在kubernetes集群上的3个或以上节点安装glusterfs的服务端集群（DaemonSet方式），并将heketi以deployment的方式部署到kubernetes集群。在我的示例部分有StorageClass和PVC的样例。本文介绍的Heketi，GlusterFS这2个组件与kubernetes集成只适合用于测试验证环境，并不适合生产环境，请注意这一点。

Heketi是一个具有resetful接口的glusterfs管理程序，作为kubernetes的Storage存储的external provisioner。 “Heketi提供了一个RESTful管理界面，可用于管理GlusterFS卷的生命周期。借助Heketi，像OpenStack Manila，Kubernetes和OpenShift这样的云服务可以动态地配置GlusterFS卷和任何支持的持久性类型。Heketi将自动确定整个集群的brick位置，确保将brick及其副本放置在不同的故障域中。Heketi还支持任意数量的GlusterFS集群，允许云服务提供网络文件存储，而不受限于单个GlusterFS集群。”

(https://github.com/whmzsu/Study-Kubernetes/blob/master/using-heketi-gluster-for-persistent-storage.md#%E6%B3%A8%E6%84%8F%E4%BA%8B%E9%A1%B9)注意事项

- 安装Glusterfs客户端：每个kubernetes集群的节点需要安装gulsterfs的客户端，如ubuntu系统的 `apt-get install glusterfs-client` 。
- 加载内核模块：每个kubernetes集群的节点运行 `modprobe dm_thin_pool` ，加载内核模块。
- 至少三个slave节点：至少需要3个kubernetes slave节点用来部署glusterfs集群，并且这3个slave节点每个节点需要至少一个空余的磁盘。

(https://github.com/whmzsu/Study-Kubernetes/blob/master/using-heketi-gluster-for-persistent-storage.md#%E6%A6%82%E8%BF%B0)概述

本指南支持在Kubernetes集群中集成，部署和管理GlusterFS 容器化的存储节点。这使得Kubernetes管理员可以为其用户提供可靠的共享存储。

跟这个话题相关的另一个重要资源是gluster-kubernetes (https://github.com/gluster/gluster-kubernetes) 项目。它专注于在Kubernetes集群中部署GlusterFS，并提供简化的工具来完成此任务。它包含一个安装指南 setup guide (https://github.com/gluster/gluster-kubernetes/blob/master/docs/setup-guide.md)。它还包括一个样例 Hello World (https://github.com/gluster/gluster-kubernetes/tree/master/docs/examples/hello_world)。其中包含一个使用动态配置（dynamically-provisioned）的GlusterFS卷进行存储的Web server pod示例。对于那些想要测试或学习更多关于此主题的人，请按照主README (https://github.com/gluster/gluster-kubernetes) 的快速入门说明 进行操作。

本指南旨在展示Heketi在Kubernetes环境中管理Gluster的最简单示例。这是为了强调这种配置的主要组成组件，因此并不适合生产环境。

(https://github.com/whmzsu/Study-Kubernetes/blob/master/using-heketi-gluster-for-persistent-storage.md#%E5%9F%BA%E7%A1%80%E8%AE%BE%E6%96%BD%E8%A6%81%E6%B1%82)基础设施要求

- 正在运行的Kubernetes集群，至少有三个Kubernetes工作节点，每个节点至少有一个可用的裸块设备（如EBS卷或本地磁盘）.
- 用于运行GlusterFS Pod的三个Kubernetes节点必须为GlusterFS通信打开相应的端口（如果开启了防火墙的情况下，没开防火墙就不需要这些操作）。在每个节点上运行以下命令。

```
iptables -N heketi
iptables -A heketi -p tcp -m state --state NEW -m tcp --dport 24007 -j ACCEPT
iptables -A heketi -p tcp -m state --state NEW -m tcp --dport 24008 -j ACCEPT
iptables -A heketi -p tcp -m state --state NEW -m tcp --dport 2222 -j ACCEPT
iptables -A heketi -p tcp -m state --state NEW -m multiport --dports 49152:49251 -j ACCEPT
service iptables save
```

(https://github.com/whmzsu/Study-Kubernetes/blob/master/using-heketi-gluster-for-persistent-storage.md#%E5%AE%A2%E6%88%B7%E7%AB%AF%E5%AE%89%E8%A3%85)客户端安装

Heketi提供了一个CLI客户端，为用户提供了一种管理Kubernetes中GlusterFS的部署和配置的方法。 在客户端机器上下载并安装Download and install the heketi-cli (https://github.com/heketi/heketi/releases)。

(https://github.com/whmzsu/Study-Kubernetes/blob/master/using-heketi-gluster-for-persistent-storage.md#glusterfs%E5%92%8Cheketi%E5%9C%A8kubernetes%E9%9B%86%E7%BE%A4%E4%B8%AD%E7%9A%84%E9%83%A8%E7%BD%B2%E8%BF%87%E7%A8%8B)Glusterfs和Heketi在Kubernetes集群中的部署过程

以下所有文件都位于下方extras/kubernetes (`git clone https://github.com/heketi/heketi.git`)。

- 部署 GlusterFS DaemonSet

```
$ kubectl create -f glusterfs-daemonset.json
```

- 通过运行如下命令获取节点名称:

```
$ kubectl get nodes
```

- 通过设置storagenode=glusterfs节点上的标签，将gluster容器部署到指定节点上。

```
$ kubectl label node <...node...> storagenode=glusterfs
```

根据需要重复打标签的步骤。验证Pod在节点上运行至少应运行3个Pod（因此至少需要给3个节点打标签）。

```
$ kubectl get pods
```

- 接下来，我们将为Heketi创建一个服务帐户（service-account）：

```
$ kubectl create -f heketi-service-account.json
```

- 我们现在必须给该服务帐户的授权绑定相应的权限来控制gluster的pod。我们通过为我们新创建的服务帐户创建群集角色绑定（cluster role binding）来完成此操作。

```
$ kubectl create clusterrolebinding heketi-gluster-admin --clusterrole=edit --serviceaccount=default:heketi-service-account
```

- 现在我们需要创建一个Kubernetes secret来保存我们Heketi实例的配置。必须将配置文件的执行程序设置为 kubernetes才能让Heketi server控制gluster pod（配置文件的默认配置）。除此这些，可以尝试配置的其他选项。

```
$ kubectl create secret generic heketi-config-secret --from-file=./heketi.json
```

- 接下来，我们需要部署一个初始（bootstrap）Pod和一个服务来访问该Pod。在你用git克隆的repo中，会有一个heketi-bootstrap.json文件。

提交文件并验证一切正常运行，如下所示：

```
# kubectl create -f heketi-bootstrap.json
service "deploy-heketi" created
deployment "deploy-heketi" created

# kubectl get pods
NAME                                             READY     STATUS    RESTARTS   AGE
deploy-heketi-1211581626-2jotm                   1/1       Running   0          35m
glusterfs-ip-172-20-0-217.ec2.internal-1217067810-4gsvx   1/1       Running   0          1h
glusterfs-ip-172-20-0-218.ec2.internal-2001140516-i9dw9   1/1       Running   0          1h
glusterfs-ip-172-20-0-219.ec2.internal-2785213222-q3hba   1/1       Running   0          1h
```

- 当Bootstrap heketi服务正在运行，我们配置端口转发，以便我们可以使用Heketi CLI与服务进行通信。使用heketi pod的名称，运行下面的命令：

```
kubectl port-forward deploy-heketi-1211581626-2jotm :8080
```

如果在运行命令的系统上本地端口8080是空闲的，则可以运行port-forward命令，以便绑定到8080以方便使用（2个命令二选一即可，我选择第二个）：

```
kubectl port-forward deploy-heketi-1211581626-2jotm 8080:8080
```

现在通过对Heketi服务运行示例查询来验证端口转发是否正常。该命令应该已经打印了将从其转发的本地端口。将其合并到URL中以测试服务，如下所示：

```
curl http://localhost:8080/hello
Handling connection for 8080
Hello from heketi
```

最后，为Heketi CLI客户端设置一个环境变量，以便它知道Heketi服务器的地址。

```
export HEKETI_CLI_SERVER=http://localhost:8080
```

- 接下来，我们将向Heketi提供有关要管理的GlusterFS集群的信息。通过拓扑文件提供这些信息。克隆的repo中有一个示例拓扑文件，名为topology-sample.json。拓扑指定运行GlusterFS容器的Kubernetes节点以及每个节点的相应原始块设备。

确保hostnames/manage指向如下所示的确切名称kubectl get nodes得到的主机名（如ubuntu-1），并且hostnames/storage是存储网络的IP地址（对应ubuntu-1的ip地址）。

IMPORTANT: 重要提示，目前，必须使用与服务器版本匹配的Heketi-cli版本加载拓扑文件。另外，Heketi pod 带有可以通过 `kubectl exec ...` 访问的heketi-cli副本。

修改拓扑文件以反映您所做的选择，然后如下所示部署它（修改主机名，IP，block 设备的名称 如xvdg）：

```
heketi-client/bin/heketi-cli topology load --json=topology-sample.json
Handling connection for 57598
        Found node ip-172-20-0-217.ec2.internal on cluster e6c063ba398f8e9c88a6ed720dc07dd2
                Adding device /dev/xvdg ... OK
        Found node ip-172-20-0-218.ec2.internal on cluster e6c063ba398f8e9c88a6ed720dc07dd2
                Adding device /dev/xvdg ... OK
        Found node ip-172-20-0-219.ec2.internal on cluster e6c063ba398f8e9c88a6ed720dc07dd2
                Adding device /dev/xvdg ... OK
```

- 接下来，我们将使用heketi为其存储其数据库提供一个卷（不要怀疑，就是使用这个命令，openshift和kubernetes通用，此命令生成heketi-storage.json文件）：

```
# heketi-client/bin/heketi-cli setup-openshift-heketi-storage
# kubectl create -f heketi-storage.json
```

> Pitfall: 注意，如果在运行setup-openshift-heketi-storage子命令时heketi-cli报告“无空间”错误，则可能无意中运行topology load命令的时候服务端和heketi-cli的版本不匹配造成的。停止正在运行的heketi pod（kubectl scale deployment deploy-heketi –replicas=0），手动删除存储块设备中的任何签名，然后继续运行heketi pod（kubectl scale deployment deploy-heketi –replicas=1）。然后用匹配版本的heketi-cli重新加载拓扑，然后重试该步骤。

- 等到作业完成后，删除bootstrap Heketi实例相关的组件：

```
# kubectl delete all,service,jobs,deployment,secret --selector="deploy-heketi"
```

- 创建长期使用的Heketi实例（存储持久化的）：

```
# kubectl create -f heketi-deployment.json
service "heketi" created
deployment "heketi" created
```

- 这样做了以后，heketi db将使用GlusterFS卷，并且每当heketi pod重新启动时都不会重置（数据不会丢失，存储持久化）。

使用诸如heketi-cli cluster list和的命令heketi-cli volume list 来确认先前建立的集群存在，并且heketi可以列出在bootstrap阶段创建的db存储卷。

(https://github.com/whmzsu/Study-Kubernetes/blob/master/using-heketi-gluster-for-persistent-storage.md#%E4%BD%BF%E7%94%A8%E6%A0%B7%E4%BE%8B)使用样例

有两种方法来调配存储。常用的方法是设置一个StorageClass，让Kubernetes为提交的PersistentVolumeClaim自动配置存储。或者，可以通过Kubernetes手动创建和管理卷（PVs），或直接使用heketi-cli中的卷。

参考gluster-kubernetes hello world example (https://github.com/gluster/gluster-kubernetes/blob/master/docs/examples/hello_world/README.md) 获取关于 storageClass 的更多信息.

(https://github.com/whmzsu/Study-Kubernetes/blob/master/using-heketi-gluster-for-persistent-storage.md#%E6%88%91%E7%9A%84%E7%A4%BA%E4%BE%8B%E9%9D%9E%E7%BF%BB%E8%AF%91%E9%83%A8%E5%88%86%E5%86%85%E5%AE%B9)我的示例（非翻译部分内容）

- topology文件：我的例子（3个节点，ubuntu-1（192.168.5.191）,ubuntu-2（192.168.5.192）,ubuntu-3（192.168.5.193）,每个节点2个磁盘用来做存储（sdb，sdc））

```
# cat topology-sample.json
```

```
{
  "clusters": [
    {
      "nodes": [
        {
          "node": {
            "hostnames": {
              "manage": [
                "ubuntu-1"
              ],
              "storage": [
                "192.168.5.191"
              ]
            },
            "zone": 1
          },
          "devices": [
            "/dev/sdb",
            "/dev/sdc"
          ]
        },
        {
          "node": {
            "hostnames": {
              "manage": [
                "ubuntu-2"
              ],
              "storage": [
                "192.168.5.192"
              ]
            },
            "zone": 1
          },
          "devices": [
            "/dev/sdb",
            "/dev/sdc"
          ]
        },
        {
          "node": {
            "hostnames": {
              "manage": [
                "ubuntu-3"
              ],
              "storage": [
                "192.168.5.193"
              ]
            },
            "zone": 1
          },
          "devices": [
            "/dev/sdb",
            "/dev/sdc"
          ]
        }
      ]
    }
  ]
}
```

- 确认glusterfs和heketi的pod运行正常

```
# kubectl get pod
NAME                                          READY     STATUS    RESTARTS   AGE
glusterfs-gf5zc                               1/1       Running   2          8h
glusterfs-ngc55                               1/1       Running   2          8h
glusterfs-zncjs                               1/1       Running   0          2h
heketi-5c8ffcc756-x9gnv                       1/1       Running   5          7h
```

- StorageClass yaml文件示例

```
# cat storage-class-slow.yaml


apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: slow                                  #-------------SC的名字
provisioner: kubernetes.io/glusterfs
parameters:
  resturl: "http://10.103.98.75:8080"         #-------------heketi service的cluster ip 和端口
  restuser: "admin"                           #-------------随便填，因为没有启用鉴权模式
  gidMin: "40000"
  gidMax: "50000"
  volumetype: "replicate:3"                   #-------------申请的默认为3副本模式
```

- PVC举例

```
# cat pvc-sample.yaml
```

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: myclaim
  annotations:
    volume.beta.kubernetes.io/storage-class: "slow"          #-------------sc的名字,需要与storageclass的名字一致
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
```

查看创建的pvc和pv

```
# kubectl get pvc|grep myclaim
NAME                        STATUS    VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
myclaim                     Bound     pvc-e98e9117-3ed7-11e8-b61d-08002795cb26   1Gi        RWO            slow           28s

# kubectl get pv|grep myclaim
NAME                                       CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS        CLAIM                              STORAGECLASS   REASON
pvc-e98e9117-3ed7-11e8-b61d-08002795cb26   1Gi        RWO            Delete           Bound     default/myclaim                    slow                    1m
```

- 可以将slow的sc设置为默认，这样平台分配存储的时候可以自动从glusterfs集群分配pv

```
# kubectl patch storageclass slow -p '{"metadata": {"annotations":{"storageclass.kubernetes.io/is-default-class":"true"}}}'
storageclass.storage.k8s.io "slow" patched

# kubectl get sc
NAME             PROVISIONER               AGE
default          fuseim.pri/ifs            1d
slow (default)   kubernetes.io/glusterfs   6h
```

(https://github.com/whmzsu/Study-Kubernetes/blob/master/using-heketi-gluster-for-persistent-storage.md#%E5%AE%B9%E9%87%8F%E9%99%90%E9%A2%9D%E6%B5%8B%E8%AF%95)容量限额测试

已经通过Helm 部署的一个mysql2 实例，使用存储2G，信息查看如下：

```
# helm list
NAME            REVISION        UPDATED                         STATUS          CHART                   NAMESPACE
mysql2          1               Thu Apr 12 15:27:11 2018        DEPLOYED        mysql-0.3.7             default
```

查看PVC和PV，大小2G，mysql2-mysql

```
# kubectl get pvc
NAME                        STATUS    VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   AGE
mysql2-mysql                Bound     pvc-ea4ae3e0-3e22-11e8-8bb6-08002795cb26   2Gi        RWO            slow           19h

# kubectl get pv
NAME                                       CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS        CLAIM                              STORAGECLASS   REASON
pvc-ea4ae3e0-3e22-11e8-8bb6-08002795cb26   2Gi        RWO            Delete           Bound         default/mysql2-mysql               slow
```

查看mysql的pod

```
# kubectl get pod|grep mysql2
mysql2-mysql-56d64f5b77-j2v84                           1/1       Running   2          19h
```

进入mysql所在容器

```
# kubectl exec -it mysql2-mysql-56d64f5b77-j2v84  /bin/bash
```

查看挂载路径，查看挂载信息

```
root@mysql2-mysql-56d64f5b77-j2v84:/#cd /var/lib/mysql
root@mysql2-mysql-56d64f5b77-j2v84:/var/lib/mysql#
root@mysql2-mysql-56d64f5b77-j2v84:/var/lib/mysql# df -h
Filesystem                                        Size  Used Avail Use% Mounted on
none                                               48G  9.2G   37G  21% /
tmpfs                                             1.5G     0  1.5G   0% /dev
tmpfs                                             1.5G     0  1.5G   0% /sys/fs/cgroup
/dev/mapper/ubuntu--1--vg-root                     48G  9.2G   37G  21% /etc/hosts
shm                                                64M     0   64M   0% /dev/shm
192.168.5.191:vol_2c2227ee65b64a0225aa9bce848a9925  2.0G  264M  1.8G  13% /var/lib/mysql
tmpfs                                             1.5G   12K  1.5G   1% /run/secrets/kubernetes.io/serviceaccount
tmpfs                                             1.5G     0  1.5G   0% /sys/firmware
```

使用dd写入数据，写入一段时间以后，空间满了，会报错（报错信息有bug，不是报空间满了，而是报文件系统只读，应该是glusterfs和docker配合的问题）

```
root@mysql2-mysql-56d64f5b77-j2v84:/var/lib/mysql# dd if=/dev/zero of=test.img bs=8M count=300

dd: error writing 'test.img': Read-only file system
dd: closing output file 'test.img': Input/output error
```

查看写满以后的文件大小

```
root@mysql2-mysql-56d64f5b77-j2v84:/var/lib/mysql# ls -l
total 2024662
-rw-r----- 1 mysql mysql         56 Apr 12 07:27 auto.cnf
-rw-r----- 1 mysql mysql       1329 Apr 12 07:27 ib_buffer_pool
-rw-r----- 1 mysql mysql   50331648 Apr 12 12:05 ib_logfile0
-rw-r----- 1 mysql mysql   50331648 Apr 12 07:27 ib_logfile1
-rw-r----- 1 mysql mysql   79691776 Apr 12 12:05 ibdata1
-rw-r----- 1 mysql mysql   12582912 Apr 12 12:05 ibtmp1
drwxr-s--- 2 mysql mysql       8192 Apr 12 07:27 mysql
drwxr-s--- 2 mysql mysql       8192 Apr 12 07:27 performance_schema
drwxr-s--- 2 mysql mysql       8192 Apr 12 07:27 sys
-rw-r--r-- 1 root  mysql 1880887296 Apr 13 02:47 test.img
```

查看挂载信息（挂载信息显示bug，应该是glusterfs的bug）

```
root@mysql2-mysql-56d64f5b77-j2v84:/var/lib/mysql# df -h
Filesystem                                      Size  Used Avail Use% Mounted on
none                                             48G  9.2G   37G  21% /
tmpfs                                           1.5G     0  1.5G   0% /dev
tmpfs                                           1.5G     0  1.5G   0% /sys/fs/cgroup
/dev/mapper/ubuntu--1--vg-root                   48G  9.2G   37G  21% /etc/hosts
shm                                              64M     0   64M   0% /dev/shm
192.168.5.191:vol_2c2227ee65b64a0225aa9bce848a9925  2.0G  -16E     0 100% /var/lib/mysql
tmpfs                                           1.5G   12K  1.5G   1% /run/secrets/kubernetes.io/serviceaccount
tmpfs                                           1.5G     0  1.5G   0% /sys/firmware
```

查看文件夹大小，为2G

```
# du -h
25M         ./mysql
825K    ./performance_schema
496K    ./sys
2.0G    .
```

如上说明glusterfs的限额作用是起效的，限制在2G的空间大小。

关注微信公众号，加入社区

上一篇: Helm chart指南-系列（4）- Chart Repository 存储库指南 (https:
下一篇: 利用NFS client provisioner动态提供Kubernetes后端存储卷-安装

标签： Glusterfs (https://www.kubernetes.org.cn/tags/glusterfs) Heketi (https://www.kubernetes.org.cn/tags/heketi)

相关推荐

- 6个与弹性伸缩、调度相关的Kubernetes附加组件 (https://www.kubernetes.org.cn/4964.html)
- Kubernetes-部署API网关Kong (https://www.kubernetes.org.cn/4952.html)
- 记一次Kubernetes/Docker网络排障 | Pod被无故重启上千次 (https://www.kubernetes.org.cn/4954.html)
- 石油巨头如何与Kubernetes, DevOps共舞？ (https://www.kubernetes.org.cn/4906.html)
- Kubernetes-基于资源配额(ResourceQuota)进行资源管理 (https://www.kubernetes.org.cn/4905.html)
- 你想知道的RocketMQ Operator干货都在这里！ (https://www.kubernetes.org.cn/4908.html)
- Kubernetes网络分析之Flannel (https://www.kubernetes.org.cn/4887.html)
- 使用CSI和Kubernetes实现动态扩容 (https://www.kubernetes.org.cn/4877.html)

评论 抢沙发

社区交流

5 8 7 4 4 提交评论

昵称 昵称 (必填)

邮箱 邮箱 (必填)

网址 网址