Table des matières

Gluster FS Installe & Création de Volume répliqué

Documentation officielle Complète
Je vais prendre 2 serveurs pour la réplication et un client, tous sous Debian 11.
Ip Statique.
Debian1: 20.2 / Debian2: 20.3 / Debian3: 20.4 Sur chaque serveur membre avoir un disque de disponible 50go pour la maquette de réplication.

Configuration IP

Configuration en ip Fixe de préférence.
Carte N°1 Network:

```
nano /etc/network/interface
```

```
#Conf réseau Fixe carte VM 1:
iface ens3 inet static
  address 10.200.20.2/28
  gateway 10.200.20.1
  dns-nameservers 10.200.20.1 1.1.1.1
  dns-domain unifi.vm
```

Carte dédié au Gulster:

```
Insert Bash Code
```

Restart conf:

```
sudo systemctl restart networking.service ; sudo ifup ens3
```

FQDN

Dans le Fichier “hosts” de chaque serveur ajouter l'entrée suivante:
Debian1:

```
127.0.0.1 debian1.unifi.vm debian1 localhost
10.200.20.3 debian2.unifi.vm
10.200.20.3 debian2
```

Debian2:

```
127.0.0.1 debian2.unifi.vm debian2 localhost
10.200.20.2 debian1.unifi.vm
10.200.20.2 debian1
```

Il sera nécessaire de redémarrer le serveur

Debian3 Le Client Gluster:

```
127.0.0.1 debian3.unifi.vm debian3 localhost
10.200.20.2 debian1.unifi.vm
10.200.20.2 debian1
10.200.20.3 debian2.unifi.vm
10.200.20.3 debian2
```

Installation

Faire la même manipulation sur chaque serveur membre du cluster.

```
apt install glusterfs-server nfs-common software-properties-common lvm2 --
no-install-recommends
```

Activation du service

```
systemctl start glusterd && systemctl enable glusterd
```

Trusted Pool

Avant d'être en mesure de gérer des volumes de stockage, les membres d'un cluster GlusterFS doivent se reconnaître entre eux et faire partie d'un même trusted pool. Sur le serveur Maitre:

```
gluster peer probe debian2
```

renvoie:

```
« peer probe: success. »
```

Sur le serveur Esclave:

```
gluster peer status
Resultat :
Number of Peers: 1
```

```
Hostname: debian1.unifi.vm
Uuid: ca4bcc87-ef16-4ffd-82a3-c8e29190f43d
State: Peer in Cluster (Connected)
```

Ajout de table de partition

Sur chaque serveur membre il faudra avoir un disque de disponible pour y créer le volume à mettre en miroir et de ce fait assurer la réplication.
Afin de trouver le disque à dédier au raid utiliser *"lsblk"*
Ajouter une table de partition sur le disque : cfdisk /dev/vdc
Créer le LVM:

```
pvcreate /dev/vdc1
```

(Volume Physique)

```
vgcreate lvmgfs1 /dev/vdc1
```

(volume Group)

```
lvcreate -l 100%VG -n gbrick1 lvmgfs1
```

(volume Logic) **formatageXFS du volume logic gbrick1**

```
mkfs.xfs -b size=1024 /dev/lvmgfs1/gbrick1
```

Pour le serveur Debian2 le Volume Group sera " **lvmgfs2** " et le Volume Logique " **gbrick1** "

Fstab

Ajouter une entrée dans fstab:

```
echo '/dev/lvmgfs1/gbrick1 /mnt/glusterfs/vol0 xfs defaults 0 0' >>
/etc/fstab
```

Créer et monter un dossier où seront « stockées » les données :

```
mkdir -p /mnt/glusterfs/vol0 && mount /mnt/glusterfs/vol0
```

Créer le dossier pour accueillir le volume GlusterFS :

```
mkdir /mnt/glusterfs/vol0/brick1
```

Sur le serveur Debian2 la Brick sera “ **brick1** ” également

---

Volume Raid1

Creer le volume Raid1 sur le serveur Esclave depuis le maitre:

```
gluster volume create gvol0 replica 2 transport tcp
debian1:/mnt/glusterfs/vol0/brick1 debian2:/mnt/glusterfs/vol0/brick1 force
```

Renvoie:

```
volume create: gvol0: success: please start the volume to access data
gluster volume start gvol0
renvoie: volume start: gvol0: success
```

Suppression d'un volume Raid
Explication:
Je me suis trompé dans le N° de brick a mappé l'ors de la création du RAID .
Du coup cela me fait un décalage dans magestion de N° de Brick.
Voici comment faire:

Remove

```
sudo gluster volume remove-brick gvol0 replica 1
debian2:/mnt/glusterfs/vol0/brick2 force
Remove-brick force will not migrate files from the removed bricks, so they
will no longer be available on the volume.
Do you want to continue? (y/n) y
volume remove-brick commit force: success
```

Le paramètre **replica 1** est obligatoire afin de préciser qu'il restera plus qu'un nœud après suppression sans replica.

Add

Ajouter la nouvelle brick:

```
sudo gluster volume add-brick gvol0 replica 2
debian2:/mnt/glusterfs/vol0/brick1 force
volume add-brick: success
```

Tout dois être ok !
Vérifier en faisant un test depuis le client.


Contrôle

```
sudo gluster volume status
Status of volume: gvol0
Gluster process                                 TCP Port  RDMA Port  Online  Pid
------------------------------------------------------------------------------
--
Brick debian1:/mnt/glusterfs/vol0/brick1    49152     0          Y       517
Brick debian2:/mnt/glusterfs/vol0/brick1    49152     0          Y       519
Self-heal Daemon on localhost               N/A       N/A        Y       530
Self-heal Daemon on debian1.unifi.vm        N/A       N/A        Y       529

Task Status of Volume gvol0
------------------------------------------------------------------------------
--
There are no active volume tasks
```

```
sudo gluster volume info

Volume Name: gvol0
Type: Replicate
Volume ID: 1a31d66c-a98c-40ea-8007-6893f30f65b4
Status: Started
Snapshot Count: 0
Number of Bricks: 1 x 2 = 2
Transport-type: tcp
Bricks:
Brick1: debian1:/mnt/glusterfs/vol0/brick1
Brick2: debian2:/mnt/glusterfs/vol0/brick1
Options Reconfigured:
performance.client-io-threads: off
nfs.disable: on
transport.address-family: inet
storage.fips-mode-rchecksum: on
```

```
cluster.granular-entry-heal: on
```

Client(s)

Installer les binaires du Client GlusterFs:

```
apt-get -y install glusterfs-client
```

Point de Montage

Sous /mnt créer un répertoire de montage nommé “glusterfs”

```
mkdir /mnt/glusterfs
```

Monter y le volume GlusterFs du serveur Maître:

```
mount -t glusterfs debian1:/gvol0 /mnt/glusterfs
```

Persistant

```
echo 'debian1:/gvol0 /mnt/glusterfs glusterfs defaults,_netdev 0 0 ' >>
/etc/fstab
```

Tester le réplica:
Créons un fichier aléatoire avec par exemple la commande ci-dessous.

```
dd if=/dev/urandom of=/mnt/glusterfs/Debian3.test  bs=1024 count=10240
```

Redondance

Il vous est possible de monter vos volume en mode redondé: 

Fstab Methode

Monter le volume en utilisant le paramètre **backupvolfile-server** dans fstab!!

```
echo 'debian1:/gvol0 /mnt/glusterfs glusterfs
defaults,_netdev,backupvolfile-server=debian2 0 0' >> /etc/fstab
```

Fichier de configuration de volume

Dans **/etc/glusterfs/** creer un fichier de configuration des volumes: (datastore.vol)

```
nano /etc/glusterfs/datastore.vol
```

Contenu de la configuration

```
volume remote1
  type protocol/client
  option transport-type tcp
  option remote-host debian1.unifi.vm
  option remote-subvolume /mnt/glusterfs
end-volume

volume remote2
  type protocol/client
  option transport-type tcp
  option remote-host debian2.unifi.vm
  option remote-subvolume /mnt/glusterfs
end-volume

volume replicate
  type cluster/replicate
  subvolumes remote1 remote2
end-volume

volume writebehind
  type performance/write-behind
  option window-size 1MB
  subvolumes replicate
end-volume

volume cache
  type performance/io-cache
  option cache-size 512MB
  subvolumes writebehind
end-volume
```

Il ne vous restera plus qu'à le monter dans **FSTAB** :

```
echo'/etc/glusterfs/datastore.vol /mnt/glusterfs glusterfs
rw,allow_other,default_permissions,max_read=131072 0 0' >> /etc/fstab
```

Ce type de configuration est fortement recommandée dans des grosses infrastructures avec plusieurs Node.

Vérification

Pour confirmer que la réplication est fonctionnelle, il suffit de vérifier avec une simple commande ls que le fichier est présent sur les bricks de chacun des deux serveurs GlusterFS :
Sur chacun des serveurs Membres:

```
ls /mnt/glusterfs/vol0/brick1/
```

pour le serveur Debian1 (Maître)

```
ls /mnt/glusterfs/vol0/brick1/
```

pour le serveur Debian2 (Esclave) Le fichier “Debian3.test” est bien synchronisé sur chacun des serveurs Membres.

Tolérance de panne

Nous éteindrons le serveur Debian1 pour le test:

```
sudo halt -p
```

Sur le Client Debian3 dans les logs: /var/log/glusterfs/mnt-glusterfs.log

```
[2022-05-23 16:57:45.842409 +0000] C [rpc-clnt-
ping.c:152:rpc_clnt_ping_timer_expired] 0-gvol0-client-1: server
10.200.20.3:49152 has not responded in the last 42 seconds, disconnecting.
[2022-05-23 16:57:45.843598 +0000] I [MSGID: 114018]
[client.c:2227:client_rpc_notify] 0-gvol0-client-1: disconnected from
client, process will keep trying to connect glusterd until brick's port is
available [{conn-name=gvol0-client-1}]
```

Nous voyons bien la déconnexion avec un time out de 42sec.

Modifier le Time Out

Un point que vous aurez noté également, c’est que la bascule n’est pas immédiate. En pratique, le délai est de 42 secondes. Pour ramener ce délai à une valeur plus raisonnable de 5 secondes, modifions notre nœud comme suit :

```
sudo gluster volume set gvol0 network.ping-timeout 5
volume set: success
```

Pour lister vos volumes GlusterFs:

```
sudo gluster volume list
```

Ce changement est tracé dans le log /var/log/glusterfs/glustershd.log avec une ligne par nœud comme celle-ci :

```
[2022-05-23 17:14:43.851448 +0000] I [rpc-clnt.c:1946:rpc_clnt_reconfig] 0-gvol0-client-1: changing ping timeout to 5 (from 42)
```

Ligne de Commande

Peer Commands

```
The peer commands are used to manage the Trusted Server Pool (TSP).
Command     Syntax     Description
peer probe  peer probe server     Add server to the TSP
peer detach     peer detach server     Remove server from the TSP
peer status     peer status     Display the status of all nodes in the TSP
pool list   pool list     List all nodes in the TSP
```

Volume Comands

```
Command     Syntax     Description
volume create   volume create volname [options] bricks     Create a volume called volname using the specified bricks with the configuration specified by options
volume start    volume start volname [force]     Start volume volname
volume stop     volume stop volname     Stop volume volname
volume info     volume info [volname]     Display volume info for volname if provided, else for all volumes on the TSP
volume status   volumes status[volname]     Display volume status for volname if provided, else for all volumes on the TSP
volume list     volume list     List all volumes in the TSP
volume set     volume set volname option value     Set option to value for volname
volume get  volume get volname <option|all>     Display the value of option (if specified)for volname , or all options otherwise
volume add-brick    volume add-brick brick-1 ... brick-n     Expand volname to include the bricks brick-1 to brick-n
volume remove-brick     volume remove-brick brick-1 ... brick-n \<start|stop|status|commit|force>     Shrink volname by removing the bricks brick-1 to brick-n . start will trigger a rebalance to migrate data from the removed bricks. stop will stop an ongoing remove-brick operation. force will
```

```
remove the bricks immediately and any data on them will no longer be
accessible from Gluster clients.
volume replace-brick    volume replace-brick volname old-brick new-brick
Replace old-brick of volname with new-brick
volume delete   volume delete volname     Delete volname
```

Plus d'infos “**man glusterfs**”

Sécurité

Il est possible de restreindre l’accès à notre volume en définissant une ACL similaire à ce qui existe en NFS via le fichier /etc/exports.

```
gluster volume set gvol0 auth.allow 10.200.20.3
volume set: success
```

l est également possible de définir une wildcard, par exemple 10.200.20.* afin d’autoriser tout un réseau. Dans cet exemple, nous avons autorisé explicitement une adresse IP à se connecter au volume.

Nous aurions également pu autoriser un nom d’hôte ou plusieurs adresses IP ou noms séparés par des virgules. Le fait de définir l’attribut **auth.allow** a comme **effet immédiat d’interdire toutes les autres machines qui n’ont pas été explicitement autorisées**. Pour revenir au comportement par défaut, il faut autoriser le caractère **wildcard (*)** tout simplement. A l’inverse, **l’attribut auth.reject** n’interdit aucune machine par défaut (auth.reject avec comme valeur NONE). Il sert comme vous l’avez deviné à **interdire explicitement une machine**. Pour résumer, le contrôle d’accès a une logique similaire avec ce qui existe côté TCP Wrappers.

Corbeille sur le Volume

GlusterFS sait gérer une corbeille au niveau volume pour conserver les fichiers supprimés. Le dossier est créé automatiquement par gluster et **ne peut être supprimé**. Fait intéressant, gluster sait si on le lui dit, tirer parti de cette corbeille pour ses opérations internes. Activons donc une corbeille pour les fichiers de moins de 10 Mio.

```
$ gluster volume set gvol0 features.trash on
$ gluster volume set gvol0 features.trash-dir "Corbeille"
$ gluster volume set gvol0 features.trash-max-filesize 10485760
$ gluster volume set gvol0 features.trash-internal-op on
```

Dépannage

Node HS ? Pas de panique !

Un incident majeur sur un équipement sensible d'un système d'information, c'est bien entendu quelque chose auquel on se doit d'être préparé. Dans un système hautement disponible, tout élément qui n'est pas considéré comme un point unique de défaillance (SPOF) doit pouvoir être indisponible sans impacter fortement le bon fonctionnement du système. Nous nous retrouvons dans un état de fonctionnement dégradé. Si le système défaillant ne peut être dépanné, un processus de reconstruction doit être mis en œuvre.

Nous allons considérer que le nœud debian1 est irrémédiablement défaillant, la VM est même supprimée. Cela se vérifie par la commande suivante :

Status

```
gluster volume heal gvol0 info
Brick debian1:/mnt/glusterfs/vol0/brick1
Status: DisConnected
Number of entries: 0

Brick debian2:/mnt/glusterfs/vol0/brick1
Status: Connected
Number of entries: 0
```

Voyons étape par étape comment le nouveau serveur nommé Debian4 va prendre de relais de celui-ci. Pour cela, la première étape que je ne vais pas détailler consiste à provisionner un nouveau serveur avec le disque de données et les dépendances comme indiqué précédemment.

Ajout d'un nœud

Premièrement, on ajoute le nouveau nœud et on va confirmer qu'on a bien un nouveau nœud présent, et un ancien toujours connu du cluster mais manquant :

```
gluster peer probe Debian4
peer probe: success.

gluster peer status
Number of Peers: 2

Hostname: debian1
Uuid: a920b020-9e5a-46f6-b073-1cc8ec00ba0e
State: Peer in Cluster (Disconnected)

Hostname: debian4
```

```
Uuid: f2a03465-11bb-4c2a-a882-22933cfa2d08
State: Peer in Cluster (Connected)
```

Remplacement

Remplaçons maintenant la brick du **debian1** par celle de notre nouveau serveur **debian4** et vérifions son état de santé :

```
gluster volume replace-brick gvol0 debian1:/mnt/glusterfs/vol0/brick1
debian4:/mnt/glusterfs/vol0/brick1 commit force
volume replace-brick: success: replace-brick commit force operation
successful
```

On réconcilie le volume :

Réconciliation

```
gluster volume heal gvol0 full
Launching heal operation to perform full self heal on volume gvol0 has been
successful
Use heal info commands to check status
```

```
gluster volume heal gvol0 info
Brick debian4:/mnt/glusterfs/vol0/brick1
Status: Connected
Number of entries: 0

Brick debian2:/mnt/glusterfs/vol0/brick1
Status: Connected
Number of entries: 0
```

Et **depuis le nouveau node (Debian4)**, lançons une synchronisation :

Synchronisation

```
root@debian4:debian4:/mnt/glusterfs/vol0/brick1# gluster volume sync debian2
gvol0
Sync volume may make data inaccessible while the sync is in progress. Do you
want to continue? (y/n) y
```

Il nous reste une dernière étape :
répliquer le volume id dans les attributs étendus du système de fichiers et le propager au second serveur.
Pour le récupérer, il faut lancer la commande suivante :
Sur le serveur debian2.

Volume-id

```
root@debian2:~# getfattr -n trusted.glusterfs.volume-id
/mnt/glusterfs/vol0/brick1/
getfattr: Suppression des « / » en tête des chemins absolus
# file: mnt/glusterfs/vol0/brick1/
trusted.glusterfs.volume-id=0seEhN1zXZTFOXmRGV92ibvw==
```

Sur le nouveau serveur, on applique l'ID du volume sur la brick :

```
root@debian4://mnt/glusterfs/vol0/brick1/# setfattr -n
trusted.glusterfs.volume-id -v '0seEhN1zXZTFOXmRGV92ibvw=='
/mnt/glusterfs/vol0/brick1/
service glusterfs-server restart
```

La configuration de notre volume est bien mise à jour comme on peut le voir ci-dessous. Dans le cadre d'un volume distribué, il faudrait lancer un rééquilibrage (rebalance) du volume :

Check Status

```
root@debian2~# gluster volume info gvol0
Volume Name: gvol0
Type: Replicate
Volume ID: 1c493043-9c2d-4be6-afcd-8512577342c9
Status: Started
Number of Bricks: 1 x 2 = 2
Transport-type: tcp
Bricks:
brick-0=debian4:-mnt-glusterfs-vol0-brick1
brick-1=debian2:-mnt-glusterfs-vol0-brick1
Options Reconfigured:
performance.readdir-ahead: on
cluster.self-heal-daemon: enable
network.ping-timeout: 5
```

Enlever une Brick

Enfin, il ne reste plus qu'à retirer l'ancien nœud des peers autorisés dans le trusted pool :

```
root@debian2:~# gluster peer detach debian1
peer detach: success
root@debian2:~# gluster peer status
Number of Peers: 1
Hostname: debian4
Uuid: f2a03465-11bb-4c2a-a882-22933cfa2d08
State: Peer in Cluster (Connected)
```

Il ne doit plus apparaître dans la liste des nœuds :

```
gluster pool list
```

Vous savez désormais comment remplacer un nœud défaillant, sachant que ce processus s'applique également en cas de migration de la brick de stor0 vers un nouveau serveur.

Étendre le volume

Quand l'espace disque commence à manquer, une première solution peut être d'étendre l'espace libre sur les bricks, d'où l'intérêt d'être partis au départ sur du LVM. Une autre solution est d'étendre le cluster avec de nouveaux nœuds afin d'améliorer la disponibilité du système dans son ensemble. Cette extension du cluster se fait en outre sans interruption de service.

Pour étendre un cluster répliqué, il faut ajouter un nombre de bricks avec un nombre multiple du nombre de réplicas. Nous avons monté un volume à deux réplicas, il nous faut donc ajouter deux bricks supplémentaires. La commande **gluster volume info gvol0** nous permet de le confirmer (1×2). Nous allons ajouter donc deux serveurs **debian5** et **debian6**, avec le volume disque préparé et le package glusterfs-server installé.

La première étape consiste à **autoriser les deux nœuds** avec la commande **gluster peer probe** vue précédemment. On peut donc ensuite ajouter des bricks au volume en spécifiant les bricks de nos deux nouveaux serveurs :

```
root@debian4:~# gluster volume add-brick gvol0
debian5:/mnt/glusterfs/vol0/brick1 debian6:/mnt/glusterfs/vol0/brick1

volume add-brick: success
```

Vérifions notre volume, nous devons retrouver nos deux bricks supplémentaires.

```
root@debian4:~# gluster volume info gvol0
Volume Name: gvol0
Type: Distributed-Replicate
Volume ID: 78484dd7-35d9-4c53-9799-1195f7689bbf
Status: Started
Number of Bricks: 2 x 2 = 4
Transport-type: tcp
Bricks:
Brick debian4:/mnt/glusterfs/vol0/brick1
Brick debian2:/mnt/glusterfs/vol0/brick1
Brick debian5:/mnt/glusterfs/vol0/brick1
Brick debian6:/mnt/glusterfs/vol0/brick1
Options Reconfigured:
performance.readdir-ahead: on
```

Notre volume à deux réplicas comportant quatre nœuds se comporte donc désormais

comme un volume distribué répliqué par la magie de l'extension du volume. Seul problème, il n'y a aucune donnée sur les serveurs debian5 et debian6, ce qui n'a pas eu pour effet de libérer de l'espace disque sur les deux premiers serveurs. Il est donc nécessaire de répartir la volumétrie sur l'ensemble des bricks qui composent le volume :

```
root@debian4:~# gluster volume rebalance gvol0 start< volume rebalance:
gvol0: success: Rebalance on gvol0 has been started successfully. Use
rebalance status command to check status of the rebalance process.
ID: dffbed2e-3a0c-4d7d-9f43-9d978a546b04
```

Pour vérifier il suffit de lancer la même commande avec le paramètre status :

```
root@debian4:~# gluster volume rebalance gvol0 status
                                    Node Rebalanced-files          size
scanned      failures       skipped               status   run time in secs
                               ---------      -----------   -----------   --
---------   -----------   -----------         ------------     -------------
-
                               localhost                5        0Bytes
10             0             0            completed               2.00
                                    debian2                0        0Bytes
0              0             0            completed               1.00
                                    debian5                0        0Bytes
2              0             0            completed               1.00
                                    debian6                0        0Bytes
0              0             0            completed               1.00
volume rebalance: gvol0: success
```

Les Quotas

GlusterFS dispose d'un mécanisme permettant de définir des quotas au niveau dossier.
Ils ne sont pas activés par défaut. Pour changer ce comportement :

```
root@debian4:~# gluster volume quota gvol0 enable
volume quota : success
```

Nous allons appliquer une **limite à 1Gio sur le sous-dossier subdir** de notre volume. **Ce dossier devra avoir été impérativement créé depuis le client glusterfs** ajouté précédemment. Pour créer ce quota :

```
root@debian4:~# gluster volume quota gvol0 limit-usage /subdir 1GB
volume quota : success
```

Si nous avions souhaité créer un quota au niveau du volume, **il suffit d'indiquer / dans le chemin**. Créons un fichier approchant le quota depuis notre client GlusterFS :

```
root@debian3:/mnt/glusterfs/subdir# dd if=/dev/zero
of=/mnt/glusterfs/subdir/toto bs=1024 count=1024000
1024000+0 enregistrements lus
1024000+0 enregistrements écrits
1048576000 bytes (1,0 GB, 1000 MiB) copied, 223,044 s, 4,7 MB/s
```

Et voyons l'état du quota :

```
root@stor1:~# gluster volume quota gvol0 list
                  Path                   Hard-limit  Soft-limit      Used
Available  Soft-limit exceeded? Hard-limit exceeded?
-------------------------------------------------------------------------------
-------------------------------------------------------
/subdir                                     1.0GB     80%(819.2MB) 1000.0MB
24.0MB             Yes                   No
```

Reprenons notre commande précédente, en créant un fichier au nom différent, la création est bien interrompue sur le dépassement de quota hard :

```
root@debian3:/data# dd if=/dev/zero of=/mnt/glusterfs/subdir/tata bs=1024
count=1024000
dd: erreur d'écriture de '/mnt/glusterfs/subdir/tata': Débordement du quota
d'espace disque
dd: fermeture du fichier de sortie '/mnt/glusterfs/subdir/tata': Débordement
du quota d'espace disque
```

Vous trouverez les infos de config dans /var/lib/glusterd.
