Einrichtung eins CTDB-Clusters mit Samba 4.2 und GlusterFS 3.6.2

Author:
Stefan Kania

Ort:
St. Michaelisdonn

20. Mai 2015

Inhalt

1 Einleitung

In diesem Tutorial geht es um die Einrichtung eines Fileserverclusters, bestehend aus zwei Knoten, mit Samba 4.2. Durch das Zusammenspiel von CTDB und GlusterFS kann so ein Cluster aufgebaut werden, der sowohl im Falle eines Ausfalls eines der beiden Knoten ein *failover* realisieren kann, als auch eine Lastverteilung der Clientzugriffe.

Bei der Verwendung eines Clusters auf dem von alle Knoten gleichzeitig geschrieben werden kann, ist es ein *filelocking* zu realisieren, dass alle Knoten mitbekommen, wenn eine Datei von einem Client bearbeitet wird. Diese Aufgabe kann nur zusammen mit einem *Clusterdateisystem* gelöst werden. Es gibt verschiedene *Clusterdateisysteme* die hier verwendet werden können. Hier im Tutorial soll *GlusterFS* verwendet werden. *GlusterFS* ist einfach einzurichten, da es viele Funktionen, die bei anderen Clusterdateisystemen von Hand eingerichtet werden müssen, automatisch durchführt. Dazu gehört unter anderem, dass GlusterFS eine *split-brain*-Situation selbst heilen kann und ein automatisches *failover* durchführen kann.

Mit der Samba Version 4.2 wird CTDB zum Aufbau eines Clusters mitgeliefert. In diesem Tutorial wird deshalb die Samba-Version 4.2 verwendet um diese Funktionalität nutzen zu können.

Als Distribution kommt hier sowohl Debian als auch Ubuntu zum Einsatz. Hintergrund ist der, dass bei Debian Wheezy die Vergabe der Berechtigungen, beim Einsatz von CTDB, nicht richtig funktionieren. Im Abschnitt 6 wird näher auf die Problematik eingegangen.

2 Grundkonfiguration der Systeme

Beide Systeme wurden mit einer Grundinstallation von Debian Wheezy oder Ubuntu 14.04 ausgestattet. Es werden drei Netzwerkkarten für die Systeme eingerichtet. Eine dient zur Installation der Pakete aus dem Internet. Eine wird für die Kommunikation der Knoten untereinander eingerichtet und die dritte Netzwerkkarte soll später für die virtuelle IP des CTDB-Clusters verwendet werden. Listing 2.1 zeigt die Konfiguration eines der beiden Knoten:

```
allow-hotplug eth0
iface eth0 inet static
       address 10.0.2.15
       netmask 255.255.255.0
       gateway 10.0.2.2
       dns-nameservers 8.8.8.8
       dns-search example.net

auto eth1
               address 192.168.56.100
               netmask 255.255.255.0

auto eth2
iface eth2 inet static
       address 192.168.57.101
       netmask 255.255.255.0
```

Listing 2.1: Die Netzwerkkonfiguration

Die Schnittstelle *eth0* dient zur Kommunikation mit dem Internet. Als DNS-Server wurde hier der bereits laufende Samba 4 Domain Controller eingetragen, da der Cluster später Mitglied in der Domäne werden soll. Der Schnittstelle *eth1* wurde keine IP zugewiesen. Dieses wird später durch CTDB realisiert. Die Schnittstelle *eth2* dient nur zur Kommunikation der beiden Knoten.

3 Installation und Konfiguration von GlusterFS

Im ersten Schritt soll jetzt das Paket glusterfs-server installiert und konfiguriert werden. Dieser Vorgang muss auf beiden Knoten durchgeführt werden.

3.1 Installation der Pakte unter Debian

Bei Debian werden mit der Version 3.2 relativ alte Pakete installiert, deshalb wird jetzt als erstes das Repository von `gluster.org` wie in Listing 3.1.1 auf beiden Knoten eingebunden:

```
root@samba42-fs1:~# wget -O - http://download.gluster.org/pub/gluster/glusterfs/3.5/3.5.0/\
                          Debian/pubkey.gpg | apt-key add -
root@samba42-fs1:~# echo deb http://download.gluster.org/pub/gluster/glusterfs/3.6/3.6.2/\
                  Debian/wheezy/apt wheezy main > /etc/apt/sources.list.d/gluster.list
root@samba42-fs1:~# apt-get update
root@samba42-fs1:~# apt-get install glusterfs-server xfsprogs acl attr
```

Listing 3.1.1: Einbinden des Gluster-Repositories

Damit die Namen der beiden Hosts später richtig aufgelöst werden können und auch kein DNS-Server dafür benötigt wird, werden auf beiden Knoten die Dateien /etc/hosts win in Listing 3.1.2 angepasst:

```
192.168.57.101 samba42-1
192.168.57.102 samba42-2
```

Listing 3.1.2: Erweiterung der Datei /etc/hosts

3.2 Installation der Pakete unter Ubuntu 14.04

Um unter Ubuntu die aktuellsten Versionen von GlusterFS verwenden zu können sollte hier auch ein eigenes Repository eingebunden werden. Für die Installation des Repositories wird das Ubuntu-eigenen System *launchpad* verwendet. Listing 3.2.1 zeigt wie das Repository eingebunden wird:

```
root@ubuntu-01:~# add-apt-repository ppa:gluster/glusterfs-3.6

root@ubuntu-02:~# apt-get update

apt-get install glusterfs-server xfsprogs acl attr
```

Listing 3.2.1: Listinunterschrift

Das Repository muss auf allen Knoten eingerichtet werden. Damit die Namen der beiden Hosts später richtig aufgelöst werden können und auch kein DNS-Server dafür benötigt wird, werden auf beiden Knoten die Dateien /etc/hosts win in Listing 3.2.2 angepasst:

```
192.168.57.101 samba42-2
192.168.57.102 samba42-2
```

Listing 3.2.2: Erweiterung der Datei /etc/hosts

3.3 Konfiguration des Volumes

Nachdem das Paket installiert wurde, wird der Dienst sofort gestartet und es kann sofort begonnen werden die einzelnen Knoten zum Cluster hinzuzufügen. Auf jedem Knoten muss es mindestens ein Device geben mit dem das Volume gebildet wird. Dieses Device wird als *Brick* bezeichnet. Mehrere Bricks ergeben dann das Volume. Listing 3.3.1 zeigt wie die einzelnen Knoten zusammengefügt werden:

```
root@samba42-fs1:~# gluster peer probe samba42-2
Probe successful

root@samba42-fs2:~# gluster peer probe samba42-1
Probe successful
```

Listing 3.3.1: Einrichtung der einzelnen Knoten

Die Peers werde im Verzeichnis /var/lib/glusterd/peers/ verwaltet. Die Peers werden dort in einer Datei mit der Peer-ID gespeichert. Nachdem alle Peers bekannt gemacht wurden, kann jetzt der zusätzliche Datenträger partitioniert und formatiert werden. Listing 3.3.2 zeigt den Vorgang für einen der beiden Knoten:

```
root@samba42-fs1:~# fdisk /dev/sda
root@samba42-fs1:~# mkfs.xfs /dev/sda1
```

Listing 3.3.2: Partitionierung und Formatierung eines Knotens

Da jetzt auf beiden Knoten die Partition erstellt und formatiert wurden, kann die Partition jetzt in einen vorher angelegten Mountpoint gemountet werden. Der Vorgang wird erst manuell durchgeführt. Erst wenn das manuelle Mounten ohne Fehler durchgeführt werden konnte, kann die Datei /etc/fstab angepasst werden. Listing 3.3.3 zeigt alle Schritte:

```
mkdir /gluster
mount /dev/sda1 /gluster/
mkdir /gluster/brick
echo "/dev/sda1 /gluster xfs defaults,_netdev 0 0" >> /etc/fstab
```

Listing 3.3.3: Mounten eines Brick

Erst jetzt kann das eigentliche GlusterFS-Volume erzeugt werden. Dieser Vorgang muss nur auf einem der beiden Knoten durchgeführt werden. Hier soll ein *replica*-Volume angelegt werden, das quasi eine Spiegelung bedeutet. Es wäre auch möglich ein *distributed*-Volume zu erzeugen, dann werden die Datenträger aller Server als ein Volume bereitgestellt. Anders als beim *striped*-Volume werden hier die Dateien gleichmäßig auf alle Platten verteilt.

Bei einem *sriped*-Volume werden die Daten ähnlich eines *RAID0* zu einem großen Datenträger zusammengebunden und die Dateien in Datenblöcken auf allen Platten verteilt. Das würde zwar den Speicherplatz vergrößern und die Zugriffe beschleunigen, aber beim Totalausfall einer Platte des Volumes wären die gesamten Daten verloren. Listing 3.3.4 zeigt das Anlegen des Volumes:

```
root@samba42-fs1:~# gluster volume create gv0 replica 2 samba42-1:/gluster/brick\
                  samba42-2:/gluster/brick
Creation of volume gv0 has been successful. Please start the volume to access data.
```

Listing 3.3.4: Erzeugung des Volumes

Hinweis !

Sollte die Fehlermeldung *Host samba42-2 not a friend* kommen, muss die Namensauflösung geprüft werden. Vor allen Dingen der Eintrag in der /etc/hosts des entsprechenden Partners.

Wenn das Erstellen des Volumes fehlerfrei durchgeführt wurde, kann das Volume jetzt wie in Listing 3.3.5 getestet werden:

```
root@samba42-fs1:~# gluster volume info

Volume Name: gv0
Type: Replicate
Volume ID: d3393de7-cee1-48b3-a393-1c767304ef48
Status: Created
Number of Bricks: 1 x 2 = 2
Transport-type: tcp
Bricks:
Brick1: samba42-1:/gluster/brick
Brick2: samba42-2:/gluster/brick
```

Listing 3.3.5: Der erste Test des Volumes

Anhand des Status ist zu sehen, dass das Volume zwar erstellt wurde, aber das Volume noch nicht aktiv ist. Erst wenn das Volume gestartet wird, ändert sich der Status. Listing 3.3.6 zeigt das Starten des Volumes und den anschließenden Test:

```
root@samba42-1:~# gluster volume start gv0
Starting volume gv0 has been successful

root@samba42-fs1:~# gluster volume info

Volume Name: gv0
Type: Replicate
Volume ID: d3393de7-cee1-48b3-a393-1c767304ef48
Status: Started
Number of Bricks: 1 x 2 = 2
Transport-type: tcp
Bricks:
Brick1: samba42-1:/gluster/brick
Brick2: samba42-2:/gluster/brick
```

Listing 3.3.6: Starten des Volumes

Jetzt hat das Volume den Status *Started* und kann gemountet werden. Erst auf das gemoutete Volume können Daten geschrieben werden. Deshalb muss das Volume immer gemountet werden, auch wenn der Zugriff auf das Volume vom selben Host erfolgen soll auf dem auch der Brick liegt.

3.4 Einbinden des Volumes auf einem Client

Bei der Installation des Glusterfs-Servers wird deshalb auch immer der Glusterfs-Client gleich mit installiert. Soll das Volume über das Netzwerk gemountet werden, muss auf dem Client nur das Paket glusterfs-client installiert werden.

Hinweis !

Die Versionen des GlusterFS-Servers und aller GlusterFS-Clients müssen immer identisch sein, da es zwischen den einzelnen Versionen von GlusterFS große Unterschiede gibt.

GlusterFS läuft im Userspace, aus diesem Grund wird auf jeden Fall das Kernel-Module fuse benötigt. Ob das Modul bereits geladen ist, lässt sich wie in Listing 3.4.1 prüfen:

```
root@samba42-1:~# lsmod | grep fuse

root@samba42-1:~# modprobe fuse

root@samba42-1:~# lsmod | grep fuse
fuse 62012 1
```

Listing 3.4.1: Prüfen ob das Kernelmodule fuse geladen ist

Das Modul wird beim Mounten des Volumes automatisch geladen. Sollte später das Modul beim Mounten nicht automatisch geladen werden, kann das Modul in die Datei /etc/modules eingetragen werden. Als erstes sollte aber immer getestet werden ob sich das Modul automatisch lädt.

Für das Volume muss ein Mountpoint auf beiden Knoten erzeugt werden in den dann das Volume, beim ersten mal per Hand, gemountet werden soll. Erst wenn das Mounten auf allen Knoten per Hand ohne Fehler durchgeführt werden konnte, kann das Mounten in die Datei /etc/fstab eingetragen werden. Das Listing 3.4.2 zeigt alle benötigten Schritte für den ersten Knoten:

```
root@samba42-fs1:~# mkdir /glusterfs
root@samba42-fs1:~# mount -t glusterfs samba42-1:/gv0 /glusterfs

root@samba42-fs1:~# mount
.
.
.
samba42-1:/gv0 on /glusterfs type fuse.glusterfs (rw,relatime,user_id=0,group_id=0,\
                                          allow_other,max_read=131072)

root@samba42-fs1:~# echo samba42-1:/gv0 /glusterfs glusterfs defaults,_netdev,acl 0 0\
```

```
>> /etc/fstab
```

Listing 3.4.2: Mounten auf dem Client

Nachdem das Volume auf allen Knoten gemountet wurde, können jetzt das erste Mal Daten auf das Volume geschrieben werden. Nach dem Schreiben sollten die abgelegten Daten auf beiden Knoten vorhanden sein. Erst wenn dieses gelingt, kann mit der Konfiguration von CTDB begonnen werden.

Hinweis !

Welcher der beiden Knoten beim Mounten angegeben wird spielt keine Rolle, der GlusterFS-Client holt sich von ihm nur die Konfiguration und kommuniziert ab dem Zeitpunkt selbst mit den beteiligten Knoten. So kann der Client beim Ausfall eines Knotens automatisch auf einen anderen Knoten aus dem Cluster schwenken.

3.5 Der *ping_pong*-Test

Bei Clusterdateisystemen die zusammen mit CTDB verwendet werden sollen, ist es besonders wichtig, dass das *file-locking* die Locks auf allen Dateien knotenübergreifend konsistent hält.

Hinweis !

Auf den Debian-Systemen sind bereits alle benötigten Pakete installiert um ping_pong kompilieren zu können. Unter Ubuntu muss der *gcc* nachträglich installiert werde.

Das schöne an CTDB ist, dass es egal ist, welches Clusterdateisystem verwendet wird, solange konsistente Byte-Range-Locks durchgeführt werden können. Um das Locking des Dateisystems zu testen existiert ein das Programm `ping_pong`. Dieses kann als C-Quellcode von der URL `ftp://ftp.samba.org/pub/unpacked/ctdb/utils/ping_pong/ping_pong.c` heruntergeladen werden. Nach dem der Quellcode heruntergeladen wurde, kann das Programm kompiliert und auf alle Knoten verteilt werden. Listing 3.5.1 zeigt den Vorgang:

```
root@samba42-fs1:~# gcc -o ping_pong ping_pong.c
```

Listing 3.5.1: Kompilieren des Programms `ping_pong`

Hinweis !

Bei den folgenden Tests ist es wichtig nach jedem Test alle `ping_pong`-Prozesse erst zu beenden bevor der nächste Test gestartet wird.

Jetzt werden drei unterschiedliche Tests durchgeführt:

- Testen der Lockkohärenz
  Dieser Test zeigt wie viele Locks pro Sekunde auf einer Datei durchgeführt werden können. Der Test sollte immer erst auf einem Knoten gestartet werden. Nach dem Start des Testes wird die Anzahl der Locks pro Sekunde im Sekundenabstand angezeigt. Diese Zahl ist sehr stark von der Leistung der CPU abhängig. Die Syntax für den Aufruf des Kommandos lautet `ping_pong /pfad/zur/datei/im/cluster N` wobei der Wert `N` mindestens den Wert von *Anzahl der Knoten + 1* sein muss. Listing 3.5.2 zeigt die Ausgabe mit einem Knoten:

  ```
  root@samba42-fs1:/glusterfs# /root/ping_pong dat1 3
     1182 locks/sec
  ```

  Listing 3.5.2: ping_pong auf einem Knoten

  Wird jetzt auf dem zweiten Knoten der selbe Test gestartet, sollte sich die Anzahl der Locks auf jeden Fall reduzieren, da jetzt beiden Knoten abwechselnd auf die Datei zugreifen und diese locken. Listing 3.5.3 zeigt die Ausgabe bei zwei Knoten:

  ```
  root@samba42-fs2:/glusterfs# /root/ping_pong dat1 3
      573 locks/sec
  ```

  Listing 3.5.3: ping_pong auf zwei Knoten

Hier ist die Anzahl der Zugriffe um circa 50% reduziert, da jetzt zwei Knoten im Wechsel die Datei locken.

- Test der I/O-Kohärenz
  Bei diesem Test wird das Programm `ping_pong` nicht nur Bits zum Lesen locken, sondern auch zum Schreiben. Bei diesem Test wird die Anzahl der Locks pro Sekunde geringer sein als beim ersten Test. Auch hier wird wieder mit nur einem Knoten gestartet so wie in Listing 3.5.4:

```
root@samba42-fs1:/glusterfs# /root/ping_pong -rw dat1 3
data increment = 1
     751 locks/sec
```

Listing 3.5.4: I/O-Test mit ping_pong und einem Knoten

Auch jetzt wird wieder der gleiche Test auf dem zweiten Knoten gestartet. Listing 3.5.5 zeigt das Ergebnis:

```
root@samba42-fs2:/glusterfs# /root/ping_pong -rw dat1 3
data increment = 2
     274 locks/sec
```

Listing 3.5.5: I/O-Test mit ping_pong und zwei Knoten

Der Wert ist hier wiederum erheblich geringer als mit nur einem Knoten. Der Wert von *data increment* sollte bei jedem zusätzlichen Knoten um eins ansteigen.

- Test der mmap Kohärenz
  In diesem Test geht es darum, ein *memory-mapped file I/O*-Test durchzuführen. Dieser Test muss nicht unbedingt erfolgreich sein, nur sollte dann in der smb.conf der Parameter *use mmap = no* eingetragen werden. Listing 3.5.6 zeigt den erfolgreichen Test auf dem ersten Knoten:

```
root@samba42-fs1:/glusterfs# /root/ping_pong -rw -m dat1 3
data increment = 1
     971 locks/sec
```

Listing 3.5.6: Mmap Test mit ping_pong und einem Knoten

Jetzt kann der gleiche Test auf dem zweiten Knoten gestartet werden. Listing 3.5.7 zeigt die Ausgabe des Tests mit zwei Knoten:

```
root@samba42-fs2:/glusterfs# /root/ping_pong -rw -m dat1 3
data increment = 1
   541 locks/sec
```

Listing 3.5.7: Mmap Test mit ping_pong und zwei Knoten

Diese Tests sollten mit jedem Clusterdateisystem, dass zum ersten Mal zum Einsatz kommt, durchgeführt werden.

4 Einrichten von CTDB

CTDB dient dazu, zwei Samba-Fileserver als einen Server im Netz bereitzustellen. Die Clients nutzen nur noch den Cluster aber nicht mehr jeden einzelnen Server. Sollte einer der beiden Server nicht erreichbar sein oder gar ganz ausgefallen sein, merkt dieses der Client nicht. Sollte ein Client mit dem gerade ausgefallenen Server verbunden sein, wird er sich automatisch mit einem anderen Knoten des Clusters verbinden. Voraussetzung dafür ist, dass alle IP-Adresse über die der Cluster erreichbar ist im DNS mit dem selben Namen versehen sind. Auch sollte beim Ausfall eines der beiden Knoten der andere noch aktive Knoten die IP-Adressen des ausgefallenen Knoten übernehmen. All das ist die Aufgabe von CTDB.

4.1 Die Vorbereitungen

Die Pakete, die für den Cluster benötigt werden, sollen aus den SerNet-Repositories genutzt werden. Dazu müssen auf beiden Servern die Repositories von SerNet eingebunden und die Pakete wie in Listing 4.1.1 installiert werden. Für die Kerberos-Authentifizierung in einer AD-Domäne muss zusätzlich das Paket libpam-heimdal installiert sein:

```
root@samba42-fs1:~# apt-get install sernet-samba-winbind sernet-samba sernet-samba-ctdb\
                                  libpam-heimdal ethtool gawk
```

Listing 4.1.1: Installation der Pakete

Bevor irgendeine Konfiguration des CTDBs durchgeführt werden kann, ist unbedingt darauf zu achten, dass die Systemzeit des Clusters mit der in der Domäne übereinstimmt, da für den Beitritt und die Authentifizierung in der Domäne Kerberos verwendet wird. Aus dem Grund sollte auf den Knoten als erstes ein *NTP*-Dienst eingerichtet werden. Für den Abgleich der Zeit sollte dann der Domaincontroller in die Datei /etc/ntp.conf als erster Server eingetragen werden. Nachdem der Domaincontroller als Zeitserver eingetragen wurde, muss der NTP-Dienst neu gestartet werden. Der Dienst muss auf allen Knoten im Cluster eingetragen sein. Listing 4.1.2 zeigt die Meldung im Logfile /var/log/samba/log.wb-EXAMPLE wenn die Zeit nicht korrekt eingestellt ist:

```
[2015/02/20 18:43:17.428656, 0] ../source3/libads/sasl.c:1002(ads_sasl_spnego_bind)
kinit succeeded but ads_sasl_spnego_krb5_bind failed: Miscellaneous failure (see text) :\
        Clock skew too great
```

Listing 4.1.2: Auszug aus dem Logfile

Ein weiterer wichtiger Punkt ist die Namensauflösung in der Domäne. Die beiden Samba-Fileserver sollen später als CTDB-Cluster in die Domäne aufgenommen werden. Deshalb müssen zuerst in dem Domaincontroller die DNS-Informationen eingetragen werden. Hier ist es wichtig, dass sowohl die eigentlichen IP-Adressen der Samba-Server als auch die virtuellen IPs der Maschinen, über die später die Ausfallsicherheit und die Lastverteilung realisiert werden soll, eingetragen werden. Listing 4.1.3 zeigt das Anlegen aller Einträge:

```
root@dc:~# kinit administrator
administrator@EXAMPLE.NET's Password:

root@samba42-dc:~# samba-tool dns zonecreate samba42-dc.example.net\
                56.168.192.in-addr.arpa -k yes
Zone 56.168.192.in-addr.arpa created successfully

root@samba42-dc:~# samba-tool dns add samba42-dc.example.net example.net samba42-fs1\
         A 192.168.56.101 -k yes
Record added successfully

root@samba42-dc:~# samba-tool dns add samba42-dc.example.net example.net samba42-fs2\
         A 192.168.56.102 -k yes
Record added successfully

root@samba42-dc:~# samba-tool dns add samba42-dc.example.net 56.168.192.in-addr.arpa\
         101 PTR samba42-fs1.example.net -k yes
Record added successfully

root@samba42-dc:~# samba-tool dns add samba42-dc.example.net 56.168.192.in-addr.arpa\
         102 PTR samba-fs2.example.net -k yes
Record added successfully

root@samba42-dc:~# samba-tool dns add samba42-dc.example.net example.net cluster-42\
         A 192.168.56.201 -k yes
Record added successfully

root@samba42-dc:~# samba-tool dns add samba42-dc.example.net example.net cluster-42\
         A 192.168.56.202 -k yes
Record added successfully
```

```
root@samba42-dc:~# samba-tool dns add samba42-dc.example.net 56.168.192.in-addr.arpa 201\
          PTR cluster-42.example.net -k yes
Record added successfully

root@samba42-dc:~# samba-tool dns add samba42-dc.example.net 56.168.192.in-addr.arpa 202\
                 PTR cluster-42.example.net -k yes
Record added successfully
```

Listing 4.1.3: DNS-Einträge

Die IP-Adressen *192.168.56.201* und *192.168.56.202* sind dabei die virtuellen IP-Adressen des Clusters die von CTDB verwaltet werden.

Die "echten" IP-Adressen und die virtuellen IP-Adressen müssen alle in den DNS der Domäne eingetragen werden, sowohl in die Forward-Zone als auch in die Reverse-Zone. Erst wenn sich alle IP-Adressen auf den beiden CTDB-Knoten auflösen lassen, kann mit der der Installation und Konfiguration von CTDB begonnen werden.

4.2 Die Konfiguration von CTDB

Bei den SerNet-Paketen gibt es für CTDB die Datei /etc/default/sernet-samba-ctdb in dieser Datei werden die Parameter für den CTDB-Cluster eingetragen. Listing 4.2.1 zeigt diese Einstellungen.

Hinweis !

In vorherigen Versionen von CTDB wurden die Einstellungen immer der Datei /etc/sysconfig/ctdb vorgenommen. Um auf eine Einheitlichkeit für alle Distributionen zu erreichen, wurde der Speicherort der Datei verlagert. In dem Verzeichnis /etc/sysconfig befindet sich nur noch ein Link auf die Konfigurationsdatei.

```
# Do NOT run CTDB without a recovery lock file unless you know exactly
# what you are doing.
CTDB_RECOVERY_LOCK=/glusterfs/ctdb.lock

# List of nodes in the cluster. Default is below.
CTDB_NODES=/etc/ctdb/nodes

# List of public addresses for providing NAS services. No default.
CTDB_PUBLIC_ADDRESSES=/etc/ctdb/public_addresses
```

Listing 4.2.1: Einstellungen für CTDB

Zu diesem Zeitpunkt soll CTDB noch nicht verantwortlich für den Start der Samba-Dienste sein. Erst muss der Cluster fehlerfrei laufen, bevor CTDB die Kontrolle über die Dienste übernehmen kann.

Hinweis !

Der Lockfile für CTDB muss unbedingt im Clusterdateisystem liegen, damit alle Knoten auf diese Datei zugreifen können und das Locking ordnungsgemäß überprüfen kann.

Alle Änderungen müssen immer auf allen Knoten des CTDB-Clusters eingetragen werden. Im Anschluss müssen die beiden Dateien /etc/ctdb/nodes und /etc/ctdb/public_addresses erstellt werden. Die Datei /etc/ctdb/nodes muss auf allen Knoten identisch vorhanden sein. Die Datei /etc/ctdb/-public_addresses muss nicht auf allen Knoten identisch sein. Sie kann, je nach Netztopologie, auch unterschiedliche Adressen beinhalten.

Listing 4.2.2 zeigt die Datei /etc/ctdb/nodes:

```
192.168.57.101
192.168.57.102
```

Listing 4.2.2: Die Datei /etc/ctdb/nodes

In der Datei befindet sich nur die Liste aller Knoten des Clusters.

Listing 4.2.3 zeigt den Inhalt der Datei /etc/ctdb/public_addresses:

```
192.168.56.201/24 eth1
192.168.56.202/24 eth1
```

Listing 4.2.3: Inhalt der Datei /etc/ctdb/public_addresses

In dieser Datei befinden sich die IP-Adressen die in diesem Subnetz an die Knoten vergeben werden sollen und auf welcher Schnittstelle die IP-Adressen eingetragen werden sollen.

Jetzt kann CTDB auf allen Knoten gestartet werden. Erst wenn der Test auf allen Knoten ein *OK* zeigt, darf mit der Konfiguration des Samba-Servers begonnen werden. Das Listing 4.2.4 zeigt den Test:

```
root@samba42-fs1:~# ctdb status
Number of nodes:2
pnn:0 192.168.57.101 OK (THIS NODE)
pnn:1 192.168.57.102 OK
Generation:446598079
Size:2
hash:0 lmaster:0
hash:1 lmaster:1
Recovery mode:NORMAL (0)
Recovery master:1
```

Listing 4.2.4: Erster Test des Clusters

Jetzt muss noch der Samba-Dienst konfiguriert werden. In der Datei /etc/samba/smb.conf wird aber nur der Parameter für den Clusterbetrieb und der Verweis auf die Registry eingetragen. Alle anderen Parameter für die Domäne befinden sich dann in der Registry. Listing 4.2.5 der Datei /etc/samba/smb.conf:

```
[global]
  clustering = yes
  include =registry
```

Listing 4.2.5: Inhalt der Datei /etc/samba/smb.conf

Hinweis !

Erst wenn alle Knoten im Cluster *OK* sind, lässt sich die Registry schreiben.

Damit später alle Knoten die identische smb.conf besitzen, werden die gesamten Einstellungen in die Registry geschrieben. Wenn der Parameter *clustering = yes* in der smb.conf eingetragen ist, wird die Registry für alle Knoten über CTDB verteilt. Das private-Verzeichnis muss deshalb nicht im Cluster liegen. Das Listing 4.2.6 zeigt alle Einträge für die Grundkonfiguration der Samba-Dienste. Auch wird gleich eine erste Freigabe eingetragen:

```
root@samba42-fs1:~# net conf setparm global "workgroup" "example"
root@samba42-fs1:~# net conf setparm global "netbios name" "cluster-42"
root@samba42-fs1:~# net conf setparm global "security" "ads"
root@samba42-fs1:~# net conf setparm global "realm" "EXAMPLE.NET"
root@samba42-fs1:~# net conf setparm global "idmap config *:range" "10000-19999"
root@samba42-fs1:~# net conf setparm global "idmap config samba-ad:backend" "rid"
root@samba42-fs1:~# net conf setparm global "idmap config samba-ad:range"\
                                    "1000000-1999999"
root@samba42-fs1:~# net conf setparm global "winbind enum users" "yes"
root@samba42-fs1:~# net conf setparm global "winbind enum groups" "yes"
root@samba42-fs1:~# net conf setparm global "winbind use default domain" "yes"
root@samba42-fs1:~# net conf setparm global "winbind refresh tickets" "yes"
root@samba42-fs1:~# net conf setparm global "store dos attributes" "yes"
root@samba42-fs1:~# net conf setparm global "map acl inherit" "yes"
root@samba42-fs1:~# net conf setparm global "vfs objects" "acl_xattr"
root@samba42-fs1:~# net conf setparm global "template shell" "/bin/bash"
```

```
root@samba42-fs1:~# net conf setparm global "wins server" "192.168.56.100"

root@samba42-fs1:~# net conf addshare daten /glusterfs/daten writeable=yes\
                    guest_ok=n "Daten im Cluster"
root@samba42-fs1:~# net conf setparm daten "browsable" "no"
```

Listing 4.2.6: Einträge in die Registry

Zu Testzwecken sollten die Dienste *smbd nmbd* und *winbind* jetzt erst einmal von Hand gestartet werden und mit *smbclient* getestet werden. Erst wenn der Test überall erfolgreich war, dann sollte die Kontrolle der Dienste an CTDB übergeben werden. Nach erfolgreichem Test müssen die Dienste jetzt wieder von Hand gestoppt werden.

Da in Zukunft CTDB für den Start der Dienste `sernet-nmbd`, `sernet-smbd` und `sernet-winbindd` verantwortlich sein soll, müssen die Dienste auf allen Knoten aus der Runlevel-Konfiguration ausgetragen werden. Listing 4.2.7 zeigt wie die Dienste unter Debian ausgetragen werden:

```
root@samba42-fs1:~# insserv -r sernet-samba-winbindd
root@samba42-fs1:~# insserv -r sernet-samba-nmbd
root@samba42-fs1:~# insserv -r sernet-samba-smbd
```

Listing 4.2.7: Entfernen der Samba-Dienste aus den Runleveln unter Debian

Unter Ubuntu werden die Dienste wie in Listing 4.2.8 zu sehen mittels `update-rc.d` ausgetragen:

```
root@ubuntu-01:~# update-rc.d sernet-samba-smbd disable
root@ubuntu-01:~# update-rc.d sernet-samba-nmbd disable
root@ubuntu-01:~# update-rc.d sernet-samba-winbindd disable
```

Listing 4.2.8: Entfernen der Samba-Dienste aus den Runleveln unter Ubuntu

Damit der SerNet-Samba auch gestartet werden kann, muss in der Datei /etc/default/sernet-samba der Startmodus auf *classic* gesetzt werden. Das muss auf allen Knoten eingestellt werden! Listing 4.2.9 zeigt die geänderte Zeile in der Datei /etc/default/sernet-samba:

```
SAMBA_START_MODE="classic"
```

Listing 4.2.9: Anpassungen in der Datei /etc/default/sernet-samba

Wie schon zuvor für die Grundeinstellungen von CTDB muss jetzt die Datei /etc/default/sernet-samba-ctdb angepasst werden, damit CTDB die Kontrolle über die Samba-Dienste übernehmen kann. Listing 4.2.10 zeigt die Einträge in die Datei:

```
# What services should CTDB manage? Default is none.
CTDB_MANAGES_SAMBA=yes
CTDB_MANAGES_WINBIND=yes
```

Listing 4.2.10: Kontrolle der Samba-Dienste an CTDB übergeben

Da diese Datei von SerNet stammt, müssen die Namen der Dienste nicht angepasst werden.

Jetzt kann der Knoten der Domäne beitreten. Auch hier muss das *join* nur auf einer Maschine durchgeführt werden, alle Knoten bekommen die selbe Domäneninformationen. Listing 4.2.11 zeigt das Kommando zur Aufnahme des Clusters in die Domäne mit dem anschließenden Test auf allen Knoten:

```
root@samba42-fs1:~# net rpc join EXAMPLE -U administrator
Enter administrator's password:
Joined domain EXAMPLE.

root@samba42-fs1:~# net rpc testjoin
Join to 'EXAMPLE' is OK

root@samba42-fs2:~# net rpc testjoin
Join to 'EXAMPLE' is OK
```

Listing 4.2.11: Domänenbeitritt

Im Anschluss kann der ctdb-Dienst neu gestartet werden. Anschließend sollte der Status des CTDB-Clusters wieder getestet werden. Nach kurzer Zeit zeigt der Status auf allen Knoten wieder *OK* an.

Jetzt wird kontrolliert, ob die beiden Knoten im Cluster auch je eine IP-Adresse aus der Datei /etc/ctdb/public_addresses erhalten haben. Listing 4.2.12 zeigt die Information beider Knoten:

```
root@samba42-fs1:~# ip a l eth1
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:18:75:fa brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.201/24 brd 192.168.56.255 scope global eth1
    inet6 2003:5c:ad6a:4f88:a00:27ff:fe18:75fa/64 scope global dynamic
       valid_lft 14367sec preferred_lft 1767sec
    inet6 fe80::a00:27ff:fe18:75fa/64 scope link
       valid_lft forever preferred_lft forever

  root@samba42-fs2:~# ip a l eth1
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:42:c7:79 brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.202/24 brd 192.168.56.255 scope global eth1
    inet6 2003:5c:ad6a:4f88:a00:27ff:fe42:c779/64 scope global dynamic
       valid_lft 14350sec preferred_lft 1750sec
    inet6 fe80::a00:27ff:fe42:c779/64 scope link
       valid_lft forever preferred_lft forever
```

Listing 4.2.12: IP-Adressen der Knoten

4.3 Prüfen des Clusters mit `ctdb`

Um den CTDB-Cluster zu verwalten wird das Kommando `ctdb` verwendet. Mit dem Kommando können auch weitere Tests des Clusters oder der einzelnen Knoten durchgeführt werden. Der Statustest des Clusters wurde schon besprochen. Einer der wichtigsten Tests ist die Prüfung aller Skripte zum Starten und Stoppen der Dienste. Listing 4.3.1 zeigt den Test:

```
root@samba42-fs1:~# ctdb scriptstatus
18 scripts were executed last monitor cycle
00.ctdb Status:OK Duration:0.043 Mon Feb 16 11:08:48 2015
01.reclock Status:OK Duration:0.139 Mon Feb 16 11:08:48 2015
10.interface Status:OK Duration:0.186 Mon Feb 16 11:08:49 2015
11.natgw Status:OK Duration:0.031 Mon Feb 16 11:08:49 2015
11.routing Status:OK Duration:0.037 Mon Feb 16 11:08:49 2015
13.per_ip_routing Status:OK Duration:0.035 Mon Feb 16 11:08:49 2015
20.multipathd Status:OK Duration:0.026 Mon Feb 16 11:08:49 2015
31.clamd Status:OK Duration:0.030 Mon Feb 16 11:08:49 2015
40.fs_use Status:DISABLED
40.vsftpd Status:OK Duration:0.030 Mon Feb 16 11:08:49 2015
41.httpd Status:OK Duration:0.025 Mon Feb 16 11:08:49 2015
49.winbind Status:OK Duration:0.063 Mon Feb 16 11:08:49 2015
50.samba Status:OK Duration:0.412 Mon Feb 16 11:08:49 2015
60.ganesha Status:OK Duration:0.034 Mon Feb 16 11:08:49 2015
60.nfs Status:OK Duration:0.042 Mon Feb 16 11:08:50 2015
62.cnfs Status:OK Duration:0.040 Mon Feb 16 11:08:50 2015
70.iscsi Status:OK Duration:0.031 Mon Feb 16 11:08:50 2015
91.lvs Status:OK Duration:0.027 Mon Feb 16 11:08:50 2015
99.timeout Status:OK Duration:0.032 Mon Feb 16 11:08:50 2015
```

Listing 4.3.1: Überprüfung des Status des Clusters

Listing 4.3.2 zeigt weitere Tests um den Zustand und die IP-Adressen des Clusters anzuzeigen:

```
root@samba42-fs1:~# ctdb uptime
Current time of node : Mon Feb 16 11:13:34 2015
Ctdbd start time : (001 23:46:48) Sat Feb 14 11:26:46 2015
Time of last recovery/failover: (001 23:46:29) Sat Feb 14 11:27:05 2015
```

```
Duration of last recovery/failover: 3.149075 seconds

root@samba42-fs1:~# ctdb ping -n all
response from 0 time=0.000147 sec (11 clients)
response from 1 time=0.000959 sec (10 clients)

root@samba42-fs1:~# ctdb ip
Public IPs on node 0
192.168.56.201 1
192.168.56.202 0

root@samba42-fs1:~# ctdb ipinfo 192.168.56.201
Public IP[192.168.56.201] info on node 0
IP:192.168.56.201
CurrentNode:1
NumInterfaces:1
Interface[1]: Name:eth1 Link:up References:1
```

Listing 4.3.2: Weitere Tests

In der Manpage zu `ctdb` stehen weiter Möglichkeiten den Cluster zu testen und Informationen aus dem Log zu sehen.

4.4 Das Kommando `onnode`

Mit dem Kommando `onnode` können Kommandos auf mehreren Knoten des Clusters gleichzeitig ausgeführt werden. Das Kommando verwendet ssh zur Ausführung der Kommandos auf den verschiedenen Knoten. Aus diesem Grund ist es sinnvoll, dass der *root* eine public-key Authentifizierung ohne Passphrase auf den Knoten ausführen kann, da sonst für jeden Knoten nach dem Passwort gefragt wird. Um zum Beispiel den Status auf allen Knoten mit einem Kommando abfragen zu können, kann das Kommando `ctdb status` auch auf allen Hosts aufgerufen werden, so wie in Listing4.4.1:

```
root@samba42-fs1:~# onnode all ctdb status

>> NODE: 192.168.57.101 <<
Number of nodes:2
pnn:0 192.168.57.101 OK (THIS NODE)
pnn:1 192.168.57.102 OK
Generation:446598079
Size:2
hash:0 lmaster:0
hash:1 lmaster:1
Recovery mode:NORMAL (0)
Recovery master:1

>> NODE: 192.168.57.102 <<
Number of nodes:2
pnn:0 192.168.57.101 OK
pnn:1 192.168.57.102 OK (THIS NODE)
Generation:446598079
Size:2
hash:0 lmaster:0
hash:1 lmaster:1
Recovery mode:NORMAL (0)
Recovery master:1
```

Listing 4.4.1: Statusabfrage mit `onnode all ctdb status`

Soll der CTDB-Dienst auf allen Knoten neu gestartet werden, kann dieses auch wie in Listing 4.4.2mit `onnode` realisiert werden:

```
root@samba42-fs1:~# onnode all service ctdb restart

>> NODE: 192.168.57.101 <<
$Shutting down ctdbd service:
$Starting ctdbd service:
>> NODE: 192.168.57.102 <<
$Shutting down ctdbd service:
$killing ctdbd
$Starting ctdbd service:
```

Listing 4.4.2: Neustart des CTDB-Diesntes mit `onnode`

Als letztes Beispiel soll jetzt noch eine Datei auf allen Knoten mit `onnode` verteilt werden. Dabei ist darauf zu achten, dass die zu verteilende Datei sich im Cluster-Dateisystem befindet. Listing 4.4.3 zeigt das Kopieren einer Datei auf alle Knoten:

```
root@samba42-fs1:~# onnode all cp /glusterfs/dat1.txt /root/

>> NODE: 192.168.57.101 <<

>> NODE: 192.168.57.102 <<
```

Listing 4.4.3: Kopieren einer Datei auf alle Knoten mit `onnode`

So können Konfigurationsdateien oder Skripte auf alle Knoten kopiert und ausgeführt werden.

4.5 Testen der Domäneninformationen

Nachdem der Cluster jetzt Mitglied der Domäne ist und alle Knoten den Status *OK* haben, kann jetzt mit `wbinfo` getestet werden, ob die Benutzer und Gruppen aus der Domäne angezeigt werden. Das Listing 4.5.1 zeigt diesen Test:

```
root@samba42-fs1:~# wbinfo -u
administrator
stka
ktom
ptau
krbtgt
guest

root@samb42-fs1:~# wbinfo -g
allowed rodc password replication group
enterprise read-only domain controllers
denied rodc password replication group
read-only domain controllers
group policy creator owners
ras and ias servers
domain controllers
enterprise admins
domain computers
cert publishers
dnsupdateproxy
domain admins
domain guests
schema admins
domain users
abteilungen
produktion
verwaltung
dnsadmins
```

Listing 4.5.1: Testen ob Benutzer und Gruppen vorhanden sind

Wenn jetzt noch die Datei /etc/nsswitch.conf, auf beiden Knoten, wie in Listing 4.5.2 angepasst wird, dann sind die Benutzer und Gruppen auch mit dem Kommando `getent` sichtbar und können für die Rechtevergabe genutzt werden:

```
passwd: compat winbind
group: compat winbind
```

Listing 4.5.2: Anpassen der Datei nsswitch.conf

Hinweis !

Die Benutzer und Gruppen werden nur dann mit `getent` angezeigt wenn die Parameter *winbind enum users = yes* und/oder *winbind enum groups = yes* gesetzt sind. In Umgebungen mit sehr vielen Benutzern wird davon abgeraten diese Parameter zu setzen, da es durch die Generierung der IDs alle Benutzer oder Gruppen zu Performanceeinbrüchen kommt.

Listing 4.5.3 zeigt die Tests mit dem Kommando `getent`:

```
root@ctdb-2:~# getent passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
.
.
.
administrator:*:1000500:1000513:Administrator:/home/SAMBA-AD/administrator:/bin/bash
stka:*:1001105:1000513:Stefan Kania:/home/SAMBA-AD/stka:/bin/bash
ktom:*:1001106:1000513:kater Tom:/home/SAMBA-AD/ktom:/bin/bash
ptau:*:1001110:1000513:pan tau:/home/SAMBA-AD/ptau:/bin/bash
krbtgt:*:1000502:1000513:krbtgt:/home/SAMBA-AD/krbtgt:/bin/bash
guest:*:1000501:1000514:Guest:/home/SAMBA-AD/guest:/bin/bash

root@ctdb-2:~# getent group
root:x:0:
daemon:x:1:
bin:x:2:
.
.
.
allowed rodc password replication group:x:1000571:
enterprise read-only domain controllers:x:1000498:
denied rodc password replication group:x:1000572:krbtgt
read-only domain controllers:x:1000521:
group policy creator owners:x:1000520:administrator
ras and ias servers:x:1000553:
domain controllers:x:1000516:
enterprise admins:x:1000519:administrator
domain computers:x:1000515:
cert publishers:x:1000517:
dnsupdateproxy:x:1001102:
domain admins:x:1000512:administrator
domain guests:x:1000514:
schema admins:x:1000518:administrator
domain users:x:1000513:
abteilungen:x:1001107:
produktion:x:1001109:ktom
verwaltung:x:1001108:stka
dnsadmins:x:1001101:
```

Listing 4.5.3: Tests mit `getent`

Damit später Freigaben über Einträge in der Registry von Windows aus erstellt werden können, muss noch das Privileg *SeDiskOperatorPrivilege* gesetzt werden. Durch dieses Privileg hat der Benutzer, oder alle Mitglieder einer eingetragenen Gruppe, das Recht Freigaben über den `RegEdit` von Windows zu erstellen. Listing 4.5.4 zeigt diesen Vorgang:

```
root@samba42-fs1:~# net rpc rights grant 'example\domain admins' SeDiskOperatorPrivilege\
             -Uadministrator
Enter administrator's password:
Successfully granted rights.
```

Listing 4.5.4: Setzen des Privilegs

5 Behebung von Fehlern und Ausfällen

Wer meint, das jetzt alles gut ist, der irrt. Denn irgendwas fällt bestimmt irgendwann aus. Deshalb soll in diesem Abschnitt noch das Thema Fehlerbehebung und Ausfall eines Dienstes oder eines gesamten Server besprochen werden.

5.1 Ausfall eines CTDB-Knotens

Was passiert wenn einer der beiden Knoten ausfällt? Um das zu simulieren, wird jetzt auf einem der beiden Knoten der CDTB-Dienst beendet. Anschließen werden auf dem noch laufenden Knoten die Tests aus dem vorherigen Abschnitt wiederholt. Das Listing 5.1.1 zeigt alle Schritte. Dabei muss darauf geachtet werden, dass die Kommandos auf dem richtigen Knoten abgesetzt werden. Auf dem Knoten *samba42-fs1* soll der Dienst beendet werden:

```
root@samba42-fs1:/glusterfs# service sernet-samba-ctdbd stop
[ ok ing down CTDB ctdbd : .

root@samba42-fs1:/glusterfs# ps ax | grep winbind
19103 pts/1 S+ 0:00 grep winbind

root@samba42-fs1:/glusterfs# ps ax | grep nmbd
19105 pts/1 S+ 0:00 grep nmbd

root@samba42-fs2:/glusterfs# ctdb status
Number of nodes:2
pnn:0 192.168.57.101 DISCONNECTED|UNHEALTHY|INACTIVE
pnn:1 192.168.57.102 OK (THIS NODE)
Generation:1830540882
Size:1
hash:0 lmaster:1
Recovery mode:NORMAL (0)
Recovery master:1

root@samba42-fs2:/glusterfs# ctdb scriptstatus
18 scripts were executed last monitor cycle
00.ctdb Status:OK Duration:0.048 Mon Feb 16 16:54:14 2015
01.reclock Status:OK Duration:0.140 Mon Feb 16 16:54:14 2015
10.interface Status:OK Duration:0.226 Mon Feb 16 16:54:15 2015
11.natgw Status:OK Duration:0.027 Mon Feb 16 16:54:15 2015
11.routing Status:OK Duration:0.026 Mon Feb 16 16:54:15 2015
13.per_ip_routing Status:OK Duration:0.027 Mon Feb 16 16:54:15 2015
20.multipathd Status:OK Duration:0.026 Mon Feb 16 16:54:15 2015
31.clamd Status:OK Duration:0.025 Mon Feb 16 16:54:15 2015
40.fs_use Status:DISABLED
40.vsftpd Status:OK Duration:0.030 Mon Feb 16 16:54:15 2015
41.httpd Status:OK Duration:0.032 Mon Feb 16 16:54:15 2015
49.winbind Status:OK Duration:0.092 Mon Feb 16 16:54:15 2015
50.samba Status:OK Duration:0.457 Mon Feb 16 16:54:15 2015
60.ganesha Status:OK Duration:0.032 Mon Feb 16 16:54:16 2015
60.nfs Status:OK Duration:0.041 Mon Feb 16 16:54:16 2015
62.cnfs Status:OK Duration:0.061 Mon Feb 16 16:54:16 2015
70.iscsi Status:OK Duration:0.024 Mon Feb 16 16:54:16 2015
91.lvs Status:OK Duration:0.029 Mon Feb 16 16:54:16 2015
```

```
99.timeout Status:OK Duration:0.026 Mon Feb 16 16:54:16 2015

root@samba42-fs2:/glusterfs# ctdb ping -n all
response from 1 time=0.000106 sec (11 clients)

root@samba42-fs2:/glusterfs# ip a l eth1
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:42:c7:79 brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.201/24 brd 192.168.56.255 scope global eth1
    inet 192.168.56.202/24 brd 192.168.56.255 scope global secondary eth1
    inet6 2003:5c:ad6a:4f88:a00:27ff:fe42:c779/64 scope global dynamic
       valid_lft 14354sec preferred_lft 1754sec
    inet6 fe80::a00:27ff:fe42:c779/64 scope link
       valid_lft forever preferred_lft forever
```

Listing 5.1.1: Test des Ausfalls eines Knoten

Hier ist zu sehen, dass sich der Status des Knotens *samba42-fs1* mit der IP-Adresse *192.168.57.101* auf *DISCONNECTED—UNHEALTHY—INACTIVE* geändert hat. Bei dem Status der Skripte hat sich nichts geändert, da dieser Test nur die lokalen Prozesse prüft. Bei dem Versuch nur alle Knoten des Clusters mittels `ping` zu erreichen, wird auch nur noch der eine Knoten erreicht.

Interessant ist die Ausgabe des Kommandos `ip a l eth1`. Da ist zu sehen, das jetzt beide IP-Adressen des Clusters auf dem noch aktiven Knoten liegen.

Hinweis !

Wenn an Stelle von `ip a l eth1` das Kommando `ifconfig eth1` verwendet wird, werden nicht beide IP-Adressen angezeigt, sondern nur die IP-Adresse, die der Knoten ursprünglich besessen hat

Alle Clients die vorher mit dem Knoten *samba42-fs1* verbunden waren, schwenken automatisch auf den anderen Knoten. Damit können alle Clients weiterhin Daten auf dem Cluster speichern.

Jetzt kann der CTDB-Dienst auf dem Knoten wieder gestartet werden. Nach kurzer Zeit wird der Knoten die Kommunikation im Cluster wieder aufgenommen haben und auch die Samba-Dienste laufen wieder auf dem Server. Listing 5.1.2 zeigt dieses:

```
root@samba42-fs1:/glusterfs# service sernet-samba-ctdbd start
[ ok ing CTDB ctdbd : .

root@samba42-fs1:/glusterfs# ctdb status
Number of nodes:2
pnn:0 192.168.57.101 OK (THIS NODE)
pnn:1 192.168.57.102 OK
Generation:1835473178
Size:2
hash:0 lmaster:0
hash:1 lmaster:1
Recovery mode:NORMAL (0)
Recovery master:1

root@samba42-fs1:/glusterfs# ps ax | grep winbind
19591 ? Ss 0:00 /usr/sbin/winbindd -D
19595 ? S 0:00 /usr/sbin/winbindd -D
19641 ? S 0:00 /usr/sbin/winbindd -D
19656 ? S 0:00 /usr/sbin/winbindd -D
19657 ? S 0:00 /usr/sbin/winbindd -D
20932 pts/1 S+ 0:00 grep winbind

root@samba42-fs1:/glusterfs# ps ax | grep mbd
19619 ? SNs 0:00 /usr/sbin/nmbd -D
19624 ? SNs 0:00 /usr/sbin/smbd -D
19658 ? SN 0:00 /usr/sbin/smbd -D
21055 pts/1 S+ 0:00 grep mbd
```

Listing 5.1.2: Einschalten des CTDB-Dienstes auf den Knoten

5.2 Ausfall eines gesamten Knotens

Im vorherigen Abschnitt ging es darum, was passiert, wenn ein CTDB-Dienst auf einem Knoten ausfällt. Wenn nur der CTDB-Dienst ausfällt und ein Client Daten auf den Server schreibt, werden die Daten im Hintergrund immer noch auf beide Knoten geschrieben, da ja GlusterFS für die Integrität der Daten auf den Datenträgern sorgt und nicht CTDB. Was passiert aber wenn der gesamte Knoten ausfällt? Um das zu testen, wird jetzt einer der beiden Knoten, in diesem Test wiederum der Knoten *samba42-fs1*, hart abgeschaltet um einen Serverausfall zu simulieren.

Listing 5.2.1 zeigt die Tests auf dem noch aktiven Knoten:

```
root@samba42-fs2:/glusterfs# gluster peer status
Number of Peers: 1

Hostname: samba42-1
Uuid: fca9e001-0b79-4e7e-b73a-a0c2d08c660b
State: Peer in Cluster (Disconnected)

root@samba42-fs2:/glusterfs# ip a l eth1
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:42:c7:79 brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.201/24 brd 192.168.56.255 scope global eth1
    inet 192.168.56.202/24 brd 192.168.56.255 scope global secondary eth1
    inet6 2003:5c:ad6a:4f88:a00:27ff:fe42:c779/64 scope global dynamic
       valid_lft 14356sec preferred_lft 1756sec
    inet6 fe80::a00:27ff:fe42:c779/64 scope link
       valid_lft forever preferred_lft forever

  root@samba42-fs2:/glusterfs# ctdb status
Number of nodes:2
pnn:0 192.168.57.101 DISCONNECTED|UNHEALTHY|INACTIVE
pnn:1 192.168.57.102 OK (THIS NODE)
Generation:1390327256
Size:1
hash:0 lmaster:1
Recovery mode:NORMAL (0)
Recovery master:1
```

Listing 5.2.1: Tests auf dem verbleibendem Knoten

Jetzt soll eine neu Datei auf dem verbleibenden Knoten erstellt werden. Listing 5.2.2 zeigt diesen Vorgang:

```
root@samba42-fs2:/glusterfs# ls -l datei-mit-einem-knoten.txt
-rw-r--r-- 1 root root 0 Feb 16 17:21 datei-mit-einem-knoten.txt
```

Listing 5.2.2: Anlegen einer neuen Datei mit nur einem Knoten

Jetzt wird der Knoten *samba42-fs1* wieder eingeschaltet. Nachdem alle Dienste wieder gestartet sind, kann jetzt geprüft werden, ob die soeben erstellte Datei auf dem Knoten auch vorhanden ist. Wie in Listing 5.2.3 zu sehen ist, ist das der Fall:

```
root@samba42-fs1:~# ls -l /glusterfs/datei-mit-einem-knoten.txt
-rw-r--r-- 1 root root 0 Feb 16 17:21 /glusterfs/datei-mit-einem-knoten.txt
```

Listing 5.2.3: Nach dem Wiedereinschalten des Knoten

Hier greift jetzt die *Selbstheilung* des GlusterFS. Wenn der ausgefallene Knoten wieder online kommt, werden sofort und automatisch alle Daten zwischen den beiden Knoten repliziert. So kommt es nicht zu Datenverlusten. Jetzt sollte noch der Status des CTDB-Dienstes getestet werden und ob die virtuellen IP-Adressen wieder auf beide Knoten verteilt wurden. Wenn das der Fall ist, dann ist der Knoten im Cluster wieder voll einsatzbereit.

6 Verwaltung der Freigaben

Nachdem der Cluster komplett eingerichtet ist, geht es jetzt darum, die Freigaben auf dem Cluster einzurichten. Da über den Cluster auf GlusterFS zugegriffen wird, müssen die Freigaben auch entsprechend eingerichtet werden. Für die Verwendung von GlusterFS mit Samba wurde ein eigenes VFS-Modul entwickelt. Dieses Modul kann bei der Einrichtung von Freigaben verwendet werden. Dann wird auf dem Cluster das Volume ohne einen *fuse-mount* bereitgestellt. Das bedeutet schnellere Zugriffe und auf die Dauer mehr Funktionen. Die Verwendung von *fuse-mount* sollte nur noch für den Lock-file des CTDB verwendet werden. Dazu kann ein kleiner GlusterFS-Cluster erstellt werden, auf den dann alle Knoten auf die Lock-Datei zugreifen. Die eigentliche Daten-Cluster werden dann ohne *fuse-mount* verwendet. Da noch nicht alle Distributionen das vfs-Modul mitbringen werden hier beide Wege beschrieben.

Unter Debian gibt es ein Problem bei der Verwaltung der Berechtigungen wenn das Modul vfs_glusterfs nicht vorhanden ist. Dann können keine Rechte gesetzt werden, wenn das VFS-Modul acl_xattr verwendet wird. Es gibt für den Fall aber eine Zwischenlösung, und zwar kann das VFS-Modul acl_tdb verwendet werden. Dabei werden dann die Berechtigungen in einer tdb-Datenbank abgelegt und nicht im Dateisystem selbst. In größeren Dateisystemen kann das zu Performanceverlusten führen. Aber damit funktioniert die Vergabe der Rechte auch unter Windows. Listing 6.1 zeigt eine Freigabe mit dem Modul acl_tdb:

```
[daten2]
       path = /glusterfs/daten1
       read only = No
       vfs objects = acl_tdb
```

Listing 6.1: Freigabe mit acl_tdb

Wenn Centos 7 zum Einsatz kommt, kann auf die Freigabe auch zugegriffen und Rechte verwaltet werden, wenn der Cluster über fusemount gemountet wurde. Listing 6.2 zeigt eine Freigabe ohne vfs_glusterfs:

```
[daten]
       comment = Daten im Cluster
       path = /glusterfs/daten/
       read only = No
       vfs objects = acl_xattr
```

Listing 6.2: Freigabe in Centos 7

Die beste Lösung ist das mounten über das Modul vfs_glusterfs, denn dann wird direkt auf den Server zugegriffen und es muss nicht mehr der Umweg über *fusemount* gegangen werden. Listing 6.3 zeigt eine Freigabe bei der das Modul vfs_glusterfs verwendet wird:

```
[daten3]
       comment = daten 3
       guest ok = no
       read only = no
       vfs objects = acl_xattr glusterfs
       glusterfs:volume = gv0
       glusterfs:logfile = /var/log/samba/glusterfs-gv0.log
       glusterfs:loglevel = 8
       glusterfs:volfile_server = localhost
       kernel share modes = no
       path = /daten3
```

Listing 6.3: Freigaben mit vfs_glusterfs

Der Parameter *glusterfs:volfile_server = localhost* muss nur dann gesetzt werden, wenn das Volume nicht auf dem selben Server liegt. Der Wert *localhost* ist der Standardwert. Wichtig ist, dass bei der Angabe des Pfades immer der Pfad relativ zum Volume angegeben werden muss, da das Volume nicht mehr über fusemount angesprochen wird. Wenn das Modul vfs_glusterfs verwendet wird, muss die Datei /etc/default/sernet-samba-ctdb wie in Listing 6.4 angepasst werden:

```
CTDB_SAMBA_SKIP_SHARE_CHECK=yes
```

Listing 6.4: Anpassung an der CTDB-Konfiguration

Jetzt ist es so weit, der Cluster wurde mit *GlusterFS* eingerichtet, *CTDB* vergibt die IP-Adressen an die Knoten und stellt die Samba-Dienste bereit und Freigaben auf dem Cluster können eingerichtet und verwaltet werden. Damit ist der Cluster bereit für den Einsatz.

Index