Version : **2021.01**

Dernière mise-à-jour : 2021/01/26 13:54

HAR100 - Gestion de la Haute Disponibilité avec Red Hat High-Availability Cluster

Contenu du Module

Red Hat High Availability Cluster versus Red Hat Cluster Suite

Dans la version Red Hat High Availability Cluster :

- **pcs** remplace **luci**,
- **keepalived**remplace **Pirahna**. Keepalived est configuré dans le fichier **/etc/keepalived/keepalived.conf** - voir **man keepalived.conf**,
- **rgmanager** et **cman** sont remplacés par **pacemaker** et **corosync**. Cette modification introduit des **agents de ressource** qui fonctionnent avec le gestionnaire de ressources Pacemaker,
- **votequorum** remplace **qdiskd** - voir **man 5 votequorum**.

Installer le Logiciel du Module Red Hat High Availability

L'installation de Red Hat High Availability se fait simplement en utilisant **yum** :

```
[root@centos7 ~]# yum install pcs pacemaker fence-agents-all
```

Firewalld

RHEL/CentOS 7 utilisent firewalld :

```
[root@centos7 ~]# firewall-cmd --state
running
```

De ce fait, il convient de créer les règles adéquates pour la haute disponibilité :

```
[root@centos7 ~]# firewall-cmd --permanent --add-service=high-availability
success
[root@centos7 ~]# firewall-cmd --add-service=high-availability
success
```

hacluster

L'utilisateur **hacluster** est utilisé par **pcs** afin de configurer et communiquer avec chaque noeud dans le cluster. Cet utilisateur doit avoir un mot de passe :

```
[root@centos7 ~]# passwd hacluster
Changing password for user hacluster.
New password: fenestros
BAD PASSWORD: The password fails the dictionary check - it is based on a dictionary word
Retype new password: fenestros
passwd: all authentication tokens updated successfully.
```

> **Important** - Il est recommendé à ce que le mot de passe de l'utilisateur **hacluster** soit identique sur chaque noeud. Dans le cas de notre exemple, le mot de passe est **fenestros**. Le mot de passe ci-dessus vous est montré dans le cadre de l'exemple. Dans le cas d'un système en production, le mot de passe ne sera pas visible et doit être autre que fenestros !

Démarrer le daemon pcsd

Sous RHEL/CentOS 7 , le daemon **pcsd** doit être démarré sur chaque noeud afin que les mises-à-jour de la configuration du cluster puissent être propagées aux autres noeuds.

Vérifiez si pcsd est démarré :

```
[root@centos7 ~]# systemctl status pcsd
● pcsd.service - PCS GUI and remote configuration interface
   Loaded: loaded (/usr/lib/systemd/system/pcsd.service; disabled; vendor preset: disabled)
   Active: inactive (dead)
     Docs: man:pcsd(8)
```

```
           man:pcs(8)
You have new mail in /var/spool/mail/root
```

Démarrez pcsd puis configurez-le pour un démarrage automatique :

```
[root@centos7 ~]# systemctl start pcsd.service
[root@centos7 ~]# systemctl enable pcsd.service
Created symlink from /etc/systemd/system/multi-user.target.wants/pcsd.service to
/usr/lib/systemd/system/pcsd.service.
[root@centos7 ~]# systemctl status pcsd
● pcsd.service - PCS GUI and remote configuration interface
   Loaded: loaded (/usr/lib/systemd/system/pcsd.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2018-08-28 09:39:29 CEST; 11s ago
     Docs: man:pcsd(8)
           man:pcs(8)
 Main PID: 2520 (pcsd)
   CGroup: /system.slice/pcsd.service
           └─2520 /usr/bin/ruby /usr/lib/pcsd/pcsd > /dev/null &

Aug 28 09:39:22 centos7.fenestros.loc systemd[1]: Starting PCS GUI and remote configuration interface...
Aug 28 09:39:29 centos7.fenestros.loc systemd[1]: Started PCS GUI and remote configuration interface.
```

La configuration de pcsd se trouve dans le fichier **/etc/sysconfig/pcsd** :

```
[root@centos7 ~]# cat /etc/sysconfig/pcsd
# pcsd configuration file

# Set PCSD_DEBUG to true for advanced pcsd debugging information
PCSD_DEBUG=false
# Set DISABLE_GUI to true to disable GUI frontend in pcsd
PCSD_DISABLE_GUI=false
# Set web UI sesions lifetime in seconds
PCSD_SESSION_LIFETIME=3600
# List of IP addresses pcsd should bind to delimited by ',' character
```

```
#PCSD_BIND_ADDR='::'
# Set port on which pcsd should be available
#PCSD_PORT=2224

# SSL settings
# set SSL options delimited by ',' character
# list of valid options can be obtained by running
# ruby -e 'require "openssl"; puts OpenSSL::SSL.constants.grep /^OP_/'
#PCSD_SSL_OPTIONS='OP_NO_SSLv2,OP_NO_SSLv3,OP_NO_TLSv1,OP_NO_TLSv1_1'
# set SSL ciphers
#PCSD_SSL_CIPHERS='DEFAULT:!RC4:!3DES:@STRENGTH'

# Proxy settings for pcsd node to node communication
# See ENVIRONMENT section in curl(1) man page for more details.
# Proxy address
#HTTPS_PROXY=
# Do not use proxy for specified hostnames
#NO_PROXY=

# Do not change
RACK_ENV=production
```

Préparation des Machines Virtuelles

A partir de votre serveur cloud, créez 2 clones complets de la VM CentOS_7 :

```
[root@centos7 ~]# exit

[trainee@centos7 ~]$ exit

desktop@serverXX:~$ VBoxManage controlvm CentOS_7 poweroff
```

```
desktop@serverXX:~$ VBoxManage clonevm CentOS_7 --name="node1.i2tch.loc" --register --mode=all

desktop@serverXX:~$ VBoxManage clonevm CentOS_7 --name="node2.i2tch.loc" --register --mode=all
```

Modifiez la configuration réseau de la première interface réseau de node2.i2tch.loc :

```
desktop@serverXX:~$ VBoxManage modifyvm "node2.i2tch.loc" --natpf1 "node2.i2tch.loc,tcp,,4022,,22
```

Mettez les interfaces 2 et 3 de chaque VM dans le réseau interne **intnet** :

```
desktop@serverXX:~$ VBoxManage modifyvm "node1.i2tch.loc" --nic2 intnet
desktop@serverXX:~$ VBoxManage modifyvm "node1.i2tch.loc" --nic3 intnet
desktop@serverXX:~$ VBoxManage modifyvm "node2.i2tch.loc" --nic2 intnet
desktop@serverXX:~$ VBoxManage modifyvm "node2.i2tch.loc" --nic3 intnet
```

Les deux VMs ont maintenant trois interfaces réseau :

Adaptateur	**Carte 1**	**Carte 2**	**Carte 3**
Type de réseau	NAT	intnet	intnet

Démarrez les machines virtuelles **node1.i2tch.loc** et **node2.i2tch.loc** :

```
desktop@serverXX:~$ VBoxManage startvm node1.i2tch.loc --type headless
...
desktop@serverXX:~$ VBoxManage startvm node2.i2tch.loc --type headless
...
```

Connectez-vous aux deux VMs :

```
desktop@serverXX:~$ ssh -l trainee localhost -p 3022
```

```
desktop@serverXX:~$ ssh -l trainee localhost -p 4022
```

Modifiez les noms d'hôtes des deux VMs :

```
[root@centos7 ~]# nmcli general hostname node1.i2tch.loc
[root@centos7 ~]# hostname
node1.i2tch.loc
```

```
[root@centos7 ~]# nmcli general hostname node2.i2tch.loc
[root@centos7 ~]# hostname
node2.i2tch.loc
```

Important - Déconnectez-vous de et re-connectez-vous à chaque VM.

Vérifiez la configuration réseau sur chaque nœud :

```
[root@node1 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:69:d2:e8 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
       valid_lft 84419sec preferred_lft 84419sec
    inet6 fe80::c031:adb3:2f96:1705/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:69:b7:6f brd ff:ff:ff:ff:ff:ff
    inet6 fe80::9e75:d62:334c:162/64 scope link
       valid_lft forever preferred_lft forever
```

```
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:1c:5d:3d brd ff:ff:ff:ff:ff:ff
    inet6 fe80::d7e2:3ad3:ef04:636d/64 scope link
       valid_lft forever preferred_lft forever
```

```
[root@node2 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:10:90:fc brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
       valid_lft 86366sec preferred_lft 86366sec
    inet6 fe80::2d4d:e50a:4f0e:65d3/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:16:89:9f brd ff:ff:ff:ff:ff:ff
    inet6 fe80::ecac:ad95:803e:976/64 scope link
       valid_lft forever preferred_lft forever
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:73:0e:e6 brd ff:ff:ff:ff:ff:ff
    inet6 fe80::a716:721e:e633:9598/64 scope link
       valid_lft forever preferred_lft forever
```

Ethernet Channel Bonding

Le **Channel Bonding** est un regroupement d'interfaces réseau sur le même serveur afin de mettre en place la redondance ou d'augmenter les performances.

Le Channel Bonding est géré nativement sous Linux. Aucune application tierce n'est requise.

Configuration du node1.i2tch.loc

Assurez-vous que le module **bonding** soit chargé :

```
[root@node1 ~]# lsmod | grep bonding
[root@node1 ~]# modprobe bonding
[root@node1 ~]# lsmod | grep bonding
bonding               145728  0
```

Pour rendre le chargement du module persistant lors du re-démarrage, injectez **modprobe bonding** dans le fichier **/etc/rc.modules** :

```
[root@node1 ~]# echo bonding >> /etc/modules-load.d/modules.conf
```

Consultez la configuration des interfaces réseaux :

```
[root@node1 ~]# nmcli c show
NAME                UUID                                  TYPE            DEVICE
Wired connection 1  79fe874b-fef7-3522-aedb-98ec2da7c789  802-3-ethernet  enp0s3
Wired connection 2  d5779aae-201a-30d2-9a15-cca7eccb07bc  802-3-ethernet  --
Wired connection 3  600a91aa-0b32-3d0c-a4b5-29563e9f31ad  802-3-ethernet  --
```

Créez le fichier **/etc/sysconfig/network-scripts/ifcfg-bond0** :

```
[root@node1 ~]# vi /etc/sysconfig/network-scripts/ifcfg-bond0
[root@node1 ~]# cat /etc/sysconfig/network-scripts/ifcfg-bond0
DEVICE=bond0
USERCTL=no
BOOTPROTO=none
ONBOOT=yes
IPADDR=10.0.3.15
```

```
NETMASK=255.255.255.0
NETWORK=10.0.3.0
BONDING_OPTS="miimon=100 mode=balance-xor"
TYPE=Unknown
IPV6INIT=no
```

Créez le fichier **/etc/sysconfig/network-scripts/ifcfg-enp0s8** :

```
[root@node1 ~]# vi /etc/sysconfig/network-scripts/ifcfg-enp0s8
[root@node1 ~]# cat /etc/sysconfig/network-scripts/ifcfg-enp0s8
DEVICE=enp0s8
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
```

Créez le fichier **/etc/sysconfig/network-scripts/ifcfg-enp0s9** :

```
[root@node1 ~]# vi /etc/sysconfig/network-scripts/ifcfg-enp0s9
[root@node1 ~]# cat /etc/sysconfig/network-scripts/ifcfg-enp0s9
DEVICE=enp0s9
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
```

Créez le fichier **/etc/modprobe.d/bonding.conf** :

```
[root@node1 ~]# vi /etc/modprobe.d/bonding.conf
[root@node1 ~]# cat /etc/modprobe.d/bonding.conf
alias bond0 bonding
```

Modifiez le fichier **/etc/hosts** :

```
[root@node1 ~]# vi /etc/hosts
[root@node1 ~]# cat /etc/hosts
127.0.0.1       localhost.localdomain localhost
::1     localhost6.localdomain6 localhost6
10.0.3.15   node1.i2tch.loc     node1
10.0.3.16   node2.i2tch.loc        node2
```

Re-démarrez le service **network** :

```
[root@node1 ~]# systemctl restart network
[root@node1 ~]# nmcli c show
NAME                UUID                                  TYPE            DEVICE
System enp0s8       00cb8299-feb9-55b6-a378-3fdc720e0bc6  802-3-ethernet  enp0s8
System enp0s9       93d13955-e9e2-a6bd-df73-12e3c747f122  802-3-ethernet  enp0s9
Wired connection 1  79fe874b-fef7-3522-aedb-98ec2da7c789  802-3-ethernet  enp0s3
bond0               234cf049-1b53-43eb-9377-7113c0fa363e  bond            bond0
```

Vérifiez la configuration du réseau :

```
[root@node1 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:69:d2:e8 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
       valid_lft 86251sec preferred_lft 86251sec
    inet6 fe80::c031:adb3:2f96:1705/64 scope link
       valid_lft forever preferred_lft forever
```

```
3: enp0s8: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master bond0 state UP qlen 1000
    link/ether 08:00:27:69:b7:6f brd ff:ff:ff:ff:ff:ff
4: enp0s9: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master bond0 state UP qlen 1000
    link/ether 08:00:27:69:b7:6f brd ff:ff:ff:ff:ff:ff
5: bond0: <BROADCAST,MULTICAST,MASTER,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP qlen 1000
    link/ether 08:00:27:69:b7:6f brd ff:ff:ff:ff:ff:ff
    inet 10.0.3.15/24 brd 10.0.3.255 scope global bond0
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe69:b76f/64 scope link tentative dadfailed
       valid_lft forever preferred_lft forever
```

Consultez la configuration de l'interface **bond0** :

```
[root@node1 ~]# cat /proc/net/bonding/bond0
Ethernet Channel Bonding Driver: v3.7.1 (April 27, 2011)

Bonding Mode: load balancing (xor)
Transmit Hash Policy: layer2 (0)
MII Status: up
MII Polling Interval (ms): 100
Up Delay (ms): 0
Down Delay (ms): 0

Slave Interface: enp0s8
MII Status: up
Speed: 1000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 08:00:27:69:b7:6f
Slave queue ID: 0

Slave Interface: enp0s9
MII Status: up
Speed: 1000 Mbps
```

```
Duplex: full
Link Failure Count: 0
Permanent HW addr: 08:00:27:1c:5d:3d
Slave queue ID: 0
```

> **Attention** - Avant de poursuivre, arrêtez votre machine virtuelle. Créez deux clones : **node10.i2tch.loc** et **node100i2tch.loc**. Démarrez la machine **node1.i2tch.loc**.

Configuration du node2.i2tch.loc

Assurez-vous que le module **bonding** soit chargé :

```
[root@node2 ~]# lsmod | grep bonding
[root@node2 ~]# modprobe bonding
[root@node2 ~]# lsmod | grep bonding
bonding               145728  0
```

Pour rendre le chargement du module persistant lors du re-démarrage, injectez **modprobe bonding** dans le fichier **/etc/rc.modules** :

```
[root@node1 ~]# echo bonding >> /etc/modules-load.d/modules.conf
```

Consultez la configuration des interfaces réseaux :

```
[root@node2 ~]# nmcli c show
NAME                UUID                                  TYPE            DEVICE
Wired connection 1  69caf9d4-3497-3c22-ad5a-b462c1d4a213  802-3-ethernet  enp0s3
Wired connection 2  e04d1654-d23e-3244-a970-8429b06c96ad  802-3-ethernet  enp0s8
Wired connection 3  982d3b86-e2ee-3a9b-be22-b3ecc9e8be13  802-3-ethernet  enp0s9
```

Créez le fichier **/etc/sysconfig/network-scripts/ifcfg-bond0** :

```
[root@node2 ~]# vi /etc/sysconfig/network-scripts/ifcfg-bond0
[root@node2 ~]# cat /etc/sysconfig/network-scripts/ifcfg-bond0
DEVICE=bond0
USERCTL=no
BOOTPROTO=none
ONBOOT=yes
IPADDR=10.0.3.16
NETMASK=255.255.255.0
NETWORK=10.0.3.0
BONDING_OPTS="miimon=100 mode=balance-xor"
TYPE=Unknown
IPV6INIT=no
```

Créez le fichier **/etc/sysconfig/network-scripts/ifcfg-enp0s8** :

```
[root@node2 ~]# vi /etc/sysconfig/network-scripts/ifcfg-enp0s8
[root@node2 ~]# cat /etc/sysconfig/network-scripts/ifcfg-enp0s8
DEVICE=enp0s8
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
```

Créez le fichier **/etc/sysconfig/network-scripts/ifcfg-enp0s9** :

```
[root@node2 ~]# vi /etc/sysconfig/network-scripts/ifcfg-enp0s9
[root@node2 ~]# cat /etc/sysconfig/network-scripts/ifcfg-enp0s9
DEVICE=enp0s9
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
```

```
SLAVE=yes
USERCTL=no
```

Créez le fichier **/etc/modprobe.d/bonding.conf** :

```
[root@node2 ~]# vi /etc/modprobe.d/bonding.conf
[root@node2 ~]# cat /etc/modprobe.d/bonding.conf
alias bond0 bonding
```

Modifiez le fichier **/etc/hosts** :

```
[root@node2 ~]# vi /etc/hosts
[root@node2 ~]# cat /etc/hosts
127.0.0.1       localhost.localdomain localhost
::1     localhost6.localdomain6 localhost6
10.0.3.15   node1.i2tch.loc     node1
10.0.3.16   node2.i2tch.loc        node2
```

Re-démarrez le service **network** :

```
[root@node2 ~]# systemctl restart network
[root@node2 ~]# nmcli c show
NAME                UUID                                  TYPE            DEVICE
System enp0s8       00cb8299-feb9-55b6-a378-3fdc720e0bc6  802-3-ethernet  enp0s8
System enp0s9       93d13955-e9e2-a6bd-df73-12e3c747f122  802-3-ethernet  enp0s9
Wired connection 1  69caf9d4-3497-3c22-ad5a-b462c1d4a213  802-3-ethernet  enp0s3
bond0               44e713ff-ba22-44bb-9cb8-603fe130431f  bond            bond0
Wired connection 2  e04d1654-d23e-3244-a970-8429b06c96ad  802-3-ethernet  --
Wired connection 3  982d3b86-e2ee-3a9b-be22-b3ecc9e8be13  802-3-ethernet  --
```

Vérifiez la configuration du réseau :

```
[root@node2 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1
```

```
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:10:90:fc brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
       valid_lft 86157sec preferred_lft 86157sec
    inet6 fe80::2d4d:e50a:4f0e:65d3/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master bond0 state UP qlen 1000
    link/ether 08:00:27:16:89:9f brd ff:ff:ff:ff:ff:ff
4: enp0s9: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master bond0 state UP qlen 1000
    link/ether 08:00:27:16:89:9f brd ff:ff:ff:ff:ff:ff
5: bond0: <BROADCAST,MULTICAST,MASTER,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP qlen 1000
    link/ether 08:00:27:16:89:9f brd ff:ff:ff:ff:ff:ff
    inet 10.0.3.16/24 brd 10.0.3.255 scope global bond0
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe16:899f/64 scope link tentative dadfailed
       valid_lft forever preferred_lft forever
```

Consultez la configuration de l'interface **bond0** :

```
[root@node2 ~]# cat /proc/net/bonding/bond0
Ethernet Channel Bonding Driver: v3.7.1 (April 27, 2011)

Bonding Mode: load balancing (xor)
Transmit Hash Policy: layer2 (0)
MII Status: up
MII Polling Interval (ms): 100
Up Delay (ms): 0
Down Delay (ms): 0
```

```
Slave Interface: enp0s8
MII Status: up
Speed: 1000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 08:00:27:16:89:9f
Slave queue ID: 0

Slave Interface: enp0s9
MII Status: up
Speed: 1000 Mbps
Duplex: full
Link Failure Count: 0
Permanent HW addr: 08:00:27:73:0e:e6
Slave queue ID: 0
```

Tester les Serveurs

Vérifiez que chaque machine puisse se voir sur le réseau **10.0.3.0**. Vérifiez que chaque machine a accès à Internet.

```
[root@node1 ~]# ping -c1 www.free.fr
PING www.free.fr (212.27.48.10) 56(84) bytes of data.
64 bytes from www.free.fr (212.27.48.10): icmp_seq=1 ttl=63 time=36.7 ms

--- www.free.fr ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 36.751/36.751/36.751/0.000 ms
[root@node1 ~]# ping -c1 10.0.3.16
PING 10.0.3.16 (10.0.3.16) 56(84) bytes of data.
64 bytes from 10.0.3.16: icmp_seq=1 ttl=64 time=3.41 ms

--- 10.0.3.16 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
```

```
rtt min/avg/max/mdev = 3.419/3.419/3.419/0.000 ms
```

```
[root@node2 ~]# ping -c1 www.free.fr
PING www.free.fr (212.27.48.10) 56(84) bytes of data.
64 bytes from www.free.fr (212.27.48.10): icmp_seq=1 ttl=63 time=17.0 ms

--- www.free.fr ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 17.080/17.080/17.080/0.000 ms
[root@node2 ~]# ping -c1 10.0.3.15
PING 10.0.3.15 (10.0.3.15) 56(84) bytes of data.
64 bytes from 10.0.3.15: icmp_seq=1 ttl=64 time=1.01 ms

--- 10.0.3.15 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.013/1.013/1.013/0.000 ms
```

Démarrer le Service pcsd si nécessaire

Dernièrement démarrez le service pcsd sur chaque noeud si necessaire :

```
[root@node1 ~]# systemctl status pcsd
● pcsd.service - PCS GUI and remote configuration interface
   Loaded: loaded (/usr/lib/systemd/system/pcsd.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2018-08-29 11:22:01 CEST; 13min ago
     Docs: man:pcsd(8)
           man:pcs(8)
 Main PID: 552 (pcsd)
   CGroup: /system.slice/pcsd.service
           └─552 /usr/bin/ruby /usr/lib/pcsd/pcsd > /dev/null &

Aug 29 11:21:33 node1.i2tch.loc systemd[1]: Starting PCS GUI and remote conf....
Aug 29 11:22:01 node1.i2tch.loc systemd[1]: Started PCS GUI and remote confi....
```

```
Hint: Some lines were ellipsized, use -l to show in full.
```

```
[root@node2 ~]# systemctl status pcsd
● pcsd.service - PCS GUI and remote configuration interface
   Loaded: loaded (/usr/lib/systemd/system/pcsd.service; enabled; vendor preset: disabled)
   Active: active (running) since Wed 2018-08-29 11:22:02 CEST; 12min ago
     Docs: man:pcsd(8)
           man:pcs(8)
 Main PID: 538 (pcsd)
   CGroup: /system.slice/pcsd.service
           └─538 /usr/bin/ruby /usr/lib/pcsd/pcsd > /dev/null &

Aug 29 11:21:30 node2.i2tch.loc systemd[1]: Starting PCS GUI and remote conf....
Aug 29 11:22:02 node2.i2tch.loc systemd[1]: Started PCS GUI and remote confi....
Hint: Some lines were ellipsized, use -l to show in full.
```

LAB #1 - L'Authentification de l'utilisateur pcs hacluster

La commande suivante authentifie l'utilisateur **hacluster** sur node1.i2tch.loc pour les deux neouds node1.i2tch.loc et node2.i2tch.loc :

```
[root@node1 ~]# pcs cluster auth node1.i2tch.loc node2.i2tch.loc
Username: hacluster
Password: fenestros
node2.i2tch.loc: Authorized
node1.i2tch.loc: Authorized
```

Les options de la commande **pcs** sont :

```
[root@node1 ~]# pcs --help

Usage: pcs [-f file] [-h] [commands]...
Control and configure pacemaker and corosync.
```

```
Options:
    -h, --help         Display usage and exit.
    -f file            Perform actions on file instead of active CIB.
    --debug            Print all network traffic and external commands run.
    --version          Print pcs version information. List pcs capabilities if
                       --full is specified.
    --request-timeout  Timeout for each outgoing request to another node in
                       seconds. Default is 60s.
    --force            Override checks and errors, the exact behavior depends on
                       the command. WARNING: Using the --force option is
                       strongly discouraged unless you know what you are doing.

Commands:
    cluster     Configure cluster options and nodes.
    resource    Manage cluster resources.
    stonith     Manage fence devices.
    constraint  Manage resource constraints.
    property    Manage pacemaker properties.
    acl         Manage pacemaker access control lists.
    qdevice     Manage quorum device provider on the local host.
    quorum      Manage cluster quorum settings.
    booth       Manage booth (cluster ticket manager).
    status      View cluster status.
    config      View and manage cluster configuration.
    pcsd        Manage pcs daemon.
    node        Manage cluster nodes.
    alert       Manage pacemaker alerts.
```

LAB #2 - Création du cluster my_cluster

Créez le cluster **my_cluster** en propageant les fichiers de configuration à chaque noeud et en démarrant les services avec l'option **–start** :

```
[root@node1 ~]# pcs cluster setup --start --name my_cluster node1.i2tch.loc node2.i2tch.loc
Destroying cluster on nodes: node1.i2tch.loc, node2.i2tch.loc...
node1.i2tch.loc: Stopping Cluster (pacemaker)...
node2.i2tch.loc: Stopping Cluster (pacemaker)...
node2.i2tch.loc: Successfully destroyed cluster
node1.i2tch.loc: Successfully destroyed cluster

Sending 'pacemaker_remote authkey' to 'node1.i2tch.loc', 'node2.i2tch.loc'
node1.i2tch.loc: successful distribution of the file 'pacemaker_remote authkey'
node2.i2tch.loc: successful distribution of the file 'pacemaker_remote authkey'
Sending cluster config files to the nodes...
node1.i2tch.loc: Succeeded
node2.i2tch.loc: Succeeded

Starting cluster on nodes: node1.i2tch.loc, node2.i2tch.loc...
node1.i2tch.loc: Starting Cluster...
node2.i2tch.loc: Starting Cluster...

Synchronizing pcsd certificates on nodes node1.i2tch.loc, node2.i2tch.loc...
node2.i2tch.loc: Success
node1.i2tch.loc: Success
Restarting pcsd on the nodes in order to reload the certificates...
node2.i2tch.loc: Success
node1.i2tch.loc: Success
```

Pour consulter le statut du cluster, utilisez la commande **pcs cluster status** :

```
[root@node1 ~]# pcs cluster status
Cluster Status:
 Stack: corosync
 Current DC: node1.i2tch.loc (version 1.1.18-11.el7_5.3-2b07d5c5a9) - partition with quorum
 Last updated: Wed Aug 29 11:59:03 2018
 Last change: Wed Aug 29 11:53:23 2018 by hacluster via crmd on node1.i2tch.loc
 2 nodes configured
```

```
 0 resources configured

PCSD Status:
  node1.i2tch.loc: Online
  node2.i2tch.loc: Online
```

LAB #3 - Activer les services cluster sur chaque noeud

Activer les services cluster sur chaque noeud dans le cluster quand le noeud est démarré :

```
[root@node1 ~]# pcs cluster enable --all
node1.i2tch.loc: Cluster Enabled
node2.i2tch.loc: Cluster Enabled
[root@node1 ~]#
[root@node1 ~]# pcs cluster status
Cluster Status:
 Stack: corosync
 Current DC: node1.i2tch.loc (version 1.1.18-11.el7_5.3-2b07d5c5a9) - partition with quorum
 Last updated: Wed Aug 29 12:00:38 2018
 Last change: Wed Aug 29 11:53:23 2018 by hacluster via crmd on node1.i2tch.loc
 2 nodes configured
 0 resources configured

PCSD Status:
  node2.i2tch.loc: Online
  node1.i2tch.loc: Online
```

LAB #4 - Mise en place d'une clôture

Commencez par modifier le fichier **/etc/hosts** sur les deux noeuds :

```
[root@node1 ~]# vi /etc/hosts
[root@node1 ~]# cat /etc/hosts
127.0.0.1       localhost.localdomain localhost
::1     localhost6.localdomain6 localhost6
10.0.3.15   node1.i2tch.loc     node1
10.0.3.16   node2.i2tch.loc        node2
10.0.3.17   apc.i2tch.loc       apc
```

```
[root@node2 ~]# vi /etc/hosts
[root@node2 ~]# cat /etc/hosts
127.0.0.1       localhost.localdomain localhost
::1     localhost6.localdomain6 localhost6
10.0.3.15   node1.i2tch.loc     node1
10.0.3.16   node2.i2tch.loc        node2
10.0.3.17   apc.i2tch.loc        apc
```

Créez maintenant la ressource STONITH :

```
[root@node1 ~]# pcs stonith create myapc fence_apc_snmp --force ipaddr="apc.i2tch.loc"
pcmk_host_map="node1.i2tch.loc:1;node2.i2tch.loc:2" pcmk_host_check="static-list"
pcmk_host_list="node1.i2tch.loc,node2.i2tch.loc" login="trainee" passwd="trainee"
```

Vérifiez que la ressource soit active :

```
[root@node1 ~]# pcs cluster status
Cluster Status:
 Stack: corosync
 Current DC: node1.i2tch.loc (version 1.1.18-11.el7_5.3-2b07d5c5a9) - partition with quorum
 Last updated: Thu Aug 30 08:12:26 2018
 Last change: Wed Aug 29 13:55:22 2018 by root via cibadmin on node1.i2tch.loc
 2 nodes configured
 1 resource configured
```

```
PCSD Status:
  node1.i2tch.loc: Online
  node2.i2tch.loc: Online
```

Il est possible de voir la liste de tous les agents STONITH en utilisant la commande suivante :

```
[root@node1 ~]# pcs stonith list
fence_amt_ws - Fence agent for AMT (WS)
fence_apc - Fence agent for APC over telnet/ssh
fence_apc_snmp - Fence agent for APC, Tripplite PDU over SNMP
fence_bladecenter - Fence agent for IBM BladeCenter
fence_brocade - Fence agent for HP Brocade over telnet/ssh
fence_cisco_mds - Fence agent for Cisco MDS
fence_cisco_ucs - Fence agent for Cisco UCS
fence_compute - Fence agent for the automatic resurrection of OpenStack compute
                instances
fence_drac5 - Fence agent for Dell DRAC CMC/5
fence_eaton_snmp - Fence agent for Eaton over SNMP
fence_emerson - Fence agent for Emerson over SNMP
fence_eps - Fence agent for ePowerSwitch
fence_evacuate - Fence agent for the automatic resurrection of OpenStack compute
                 instances
fence_heuristics_ping - Fence agent for ping-heuristic based fencing
fence_hpblade - Fence agent for HP BladeSystem
fence_ibmblade - Fence agent for IBM BladeCenter over SNMP
fence_idrac - Fence agent for IPMI
fence_ifmib - Fence agent for IF MIB
fence_ilo - Fence agent for HP iLO
fence_ilo2 - Fence agent for HP iLO
fence_ilo3 - Fence agent for IPMI
fence_ilo3_ssh - Fence agent for HP iLO over SSH
fence_ilo4 - Fence agent for IPMI
fence_ilo4_ssh - Fence agent for HP iLO over SSH
fence_ilo_moonshot - Fence agent for HP Moonshot iLO
```

```
fence_ilo_mp - Fence agent for HP iLO MP
fence_ilo_ssh - Fence agent for HP iLO over SSH
fence_imm - Fence agent for IPMI
fence_intelmodular - Fence agent for Intel Modular
fence_ipdu - Fence agent for iPDU over SNMP
fence_ipmilan - Fence agent for IPMI
fence_kdump - Fence agent for use with kdump
fence_mpath - Fence agent for multipath persistent reservation
fence_rhevm - Fence agent for RHEV-M REST API
fence_rsa - Fence agent for IBM RSA
fence_rsb - I/O Fencing agent for Fujitsu-Siemens RSB
fence_sbd - Fence agent for sbd
fence_scsi - Fence agent for SCSI persistent reservation
fence_virt - Fence agent for virtual machines
fence_vmware_rest - Fence agent for VMware REST API
fence_vmware_soap - Fence agent for VMWare over SOAP API
fence_wti - Fence agent for WTI
fence_xvm - Fence agent for virtual machines
```

ainsi que de consulter les détails de chaque agent :

```
[root@node1 ~]# pcs stonith describe fence_apc_snmp
fence_apc_snmp - Fence agent for APC, Tripplite PDU over SNMP

fence_apc_snmp is an I/O Fencing agent which can be used with the APC network power switch or Tripplite PDU
devices.It logs into a device via SNMP and reboots a specified outlet. It supports SNMP v1, v2c, v3 with all
combinations of  authenticity/privacy settings.

Stonith options:
  ipport: TCP/UDP port to use for connection with device
  snmp_version: Specifies SNMP version to use (1,2c,3)
  community: Set the community string
  snmp_priv_prot: Set privacy protocol (DES|AES)
  port: Physical plug number, name of virtual machine or UUID
```

```
  inet6_only: Forces agent to use IPv6 addresses only
  ipaddr (required): IP Address or Hostname
  snmp_priv_passwd: Set privacy protocol password
  snmp_priv_passwd_script: Script to run to retrieve privacy password
  snmp_auth_prot: Set authentication protocol (MD5|SHA)
  inet4_only: Forces agent to use IPv4 addresses only
  passwd_script: Script to retrieve password
  snmp_sec_level: Set security level (noAuthNoPriv|authNoPriv|authPriv)
  passwd: Login password or passphrase
  login: Login Name
  verbose: Verbose mode
  debug: Write debug information to given file
  separator: Separator for CSV created by operation list
  power_wait: Wait X seconds after issuing ON/OFF
  login_timeout: Wait X seconds for cmd prompt after login
  power_timeout: Test X seconds for status change after ON/OFF
  delay: Wait X seconds before fencing is started
  shell_timeout: Wait X seconds for cmd prompt after issuing command
  retry_on: Count of attempts to retry power on
  priority: The priority of the stonith resource. Devices are tried in order of
            highest priority to lowest.
  pcmk_host_map: A mapping of host names to ports numbers for devices that do
                 not support host names. Eg. node1:1;node2:2,3 would tell the
                 cluster to use port 1 for node1 and ports 2 and 3 for node2
  pcmk_host_list: A list of machines controlled by this device (Optional unless
                  pcmk_host_check=static-list).
  pcmk_host_check: How to determine which machines are controlled by the device.
                   Allowed values: dynamic-list (query the device), static-list
                   (check the pcmk_host_list attribute), none (assume every
                   device can fence every machine)
  pcmk_delay_max: Enable a random delay for stonith actions and specify the
                  maximum of random delay. This prevents double fencing when
                  using slow devices such as sbd. Use this to enable a random
                  delay for stonith actions. The overall delay is derived from
```

```
                  this random delay value adding a static delay so that the sum
                  is kept below the maximum delay.
  pcmk_delay_base: Enable a base delay for stonith actions and specify base
                   delay value. This prevents double fencing when different
                   delays are configured on the nodes. Use this to enable a
                   static delay for stonith actions. The overall delay is
                   derived from a random delay value adding this static delay so
                   that the sum is kept below the maximum delay.
  pcmk_action_limit: The maximum number of actions can be performed in parallel
                     on this device Pengine property concurrent-fencing=true
                     needs to be configured first. Then use this to specify the
                     maximum number of actions can be performed in parallel on
                     this device. -1 is unlimited.

Default operations:
  monitor: interval=60s
```

En utilisant la commande suivante, il est possible de consulter le statut des ressources STONITH :

```
[root@node1 ~]# pcs stonith show
 myapc  (stonith:fence_apc_snmp):    Stopped
```

L'ensemble des sous-commandes de la commande **pcs stonith** peuvent être visualisées grâce à la commande suivante :

```
[root@node1 ~]# pcs stonith --help

Usage: pcs stonith [commands]...
Configure fence devices for use with pacemaker

Commands:
    [show [stonith id]] [--full]
        Show all currently configured stonith devices or if a stonith id is
        specified show the options for the configured stonith device.  If
        --full is specified all configured stonith options will be displayed.
```

```
    list [filter] [--nodesc]
        Show list of all available stonith agents (if filter is provided then
        only stonith agents matching the filter will be shown). If --nodesc is
        used then descriptions of stonith agents are not printed.

    describe <stonith agent> [--full]
        Show options for specified stonith agent. If --full is specified, all
        options including advanced ones are shown.

    create <stonith id> <stonith device type> [stonith device options]
           [op <operation action> <operation options> [<operation action>
           <operation options>]...] [meta <meta options>...]
           [--group <group id> [--before <stonith id> | --after <stonith id>]]
           [--disabled] [--wait[=n]]
        Create stonith device with specified type and options.
        If --group is specified the stonith device is added to the group named.
        You can use --before or --after to specify the position of the added
        stonith device relatively to some stonith device already existing in the
        group.
        If --disabled is specified the stonith device is not used.
        If --wait is specified, pcs will wait up to 'n' seconds for the stonith
        device to start and then return 0 if the stonith device is started, or 1
        if the stonith device has not yet started.  If 'n' is not specified it
        defaults to 60 minutes.

    update <stonith id> [stonith device options]
        Add/Change options to specified stonith id.

    delete <stonith id>
        Remove stonith id from configuration.

    enable <stonith id> [--wait[=n]]
        Allow the cluster to use the stonith device. If --wait is specified, pcs
        will wait up to 'n' seconds for the stonith device to start and then
```

```
        return 0 if the stonith device is started, or 1 if the stonith device
        has not yet started. If 'n' is not specified it defaults to 60 minutes.

    disable <stonith id> [--wait[=n]]
        Attempt to stop the stonith device if it is running and disallow the
        cluster to use it. If --wait is specified, pcs will wait up to 'n'
        seconds for the stonith device to stop and then return 0 if the stonith
        device is stopped or 1 if the stonith device has not stopped.  If 'n' is
        not specified it defaults to 60 minutes.

    cleanup [<stonith id>] [--node <node>]
        Make the cluster forget failed operations from history of the stonith
        device and re-detect its current state. This can be useful to purge
        knowledge of past failures that have since been resolved. If a stonith
        id is not specified then all resources / stonith devices will be cleaned
        up. If a node is not specified then resources / stonith devices on all
        nodes will be cleaned up.

    refresh [<stonith id>] [--node <node>] [--full]
        Make the cluster forget the complete operation history (including
        failures) of the stonith device and re-detect its current state. If you
        are interested in forgetting failed operations only, use the 'pcs
        stonith cleanup' command. If a stonith id is not specified then all
        resources / stonith devices will be refreshed. If a node is not
        specified then resources / stonith devices on all nodes will be
        refreshed. Use --full to refresh a stonith device on all nodes,
        otherwise only nodes where the stonith device's state is known will be
        considered.

    level [config]
        Lists all of the fencing levels currently configured.

    level add <level> <target> <stonith id> [stonith id]...
        Add the fencing level for the specified target with the list of stonith
```

```
        devices to attempt for that target at that level. Fence levels are
        attempted in numerical order (starting with 1). If a level succeeds
        (meaning all devices are successfully fenced in that level) then no
        other levels are tried, and the target is considered fenced.
        Target may be a node name <node_name> or %<node_name> or
        node%<node_name>, a node name regular expression regexp%<node_pattern>
        or a node attribute value attrib%<name>=<value>.

    level remove <level> [target] [stonith id]...
        Removes the fence level for the level, target and/or devices specified.
        If no target or devices are specified then the fence level is removed.
        Target may be a node name <node_name> or %<node_name> or
        node%<node_name>, a node name regular expression regexp%<node_pattern>
        or a node attribute value attrib%<name>=<value>.

    level clear [target|stonith id(s)]
        Clears the fence levels on the target (or stonith id) specified or
        clears all fence levels if a target/stonith id is not specified. If
        more than one stonith id is specified they must be separated by a comma
        and no spaces.
        Target may be a node name <node_name> or %<node_name> or
        node%<node_name>, a node name regular expression regexp%<node_pattern>
        or a node attribute value attrib%<name>=<value>.
        Example: pcs stonith level clear dev_a,dev_b

    level verify
        Verifies all fence devices and nodes specified in fence levels exist.

    fence <node> [--off]
        Fence the node specified (if --off is specified, use the 'off' API
        call to stonith which will turn the node off instead of rebooting it).

    confirm <node> [--force]
        Confirm to the cluster that the specified node is powered off. This
```

```
        allows the cluster to recover from a situation where no stonith device
        is able to fence the node. This command should ONLY be used after
        manually ensuring that the node is powered off and has no access to
        shared resources.

        WARNING: If this node is not actually powered off or it does have
        access to shared resources, data corruption/cluster failure can occur.
        To prevent accidental running of this command, --force or interactive
        user response is required in order to proceed.

        NOTE: It is not checked if the specified node exists in the cluster
        in order to be able to work with nodes not visible from the local
        cluster partition.

    sbd enable [--watchdog=<path>[@<node>]] ... [--device=<path>[@<node>]] ...
               [<SBD_OPTION>=<value>] ...
        Enable SBD in cluster. Default path for watchdog device is
        /dev/watchdog. Allowed SBD options: SBD_WATCHDOG_TIMEOUT (default: 5),
        SBD_DELAY_START (default: no) and SBD_STARTMODE (default: always). It is
        possible to specify up to 3 devices per node.

        WARNING: Cluster has to be restarted in order to apply these changes.

        Example of enabling SBD in cluster with watchdogs on node1 will be
        /dev/watchdog2, on node2 /dev/watchdog1, /dev/watchdog0 on all other
        nodes, device /dev/sdb on node1, device /dev/sda on all other nodes and
        watchdog timeout will bet set to 10 seconds:
        pcs stonith sbd enable \
            --watchdog=/dev/watchdog2@node1 \
            --watchdog=/dev/watchdog1@node2 \
            --watchdog=/dev/watchdog0 \
            --device=/dev/sdb@node1 \
            --device=/dev/sda \
            SBD_WATCHDOG_TIMEOUT=10
```

```
    sbd disable
        Disable SBD in cluster.

        WARNING: Cluster has to be restarted in order to apply these changes.

    sbd device setup --device=<path> [--device=<path>]...
                     [watchdog-timeout=<integer>] [allocate-timeout=<integer>]
                     [loop-timeout=<integer>] [msgwait-timeout=<integer>]
        Initialize SBD structures on device(s) with specified timeouts.

        WARNING: All content on device(s) will be overwritten.

    sbd device message <device-path> <node> <message-type>
        Manually set a message of the specified type on the device for the node.
        Possible message types (they are documented in sbd(8) man page): test,
        reset, off, crashdump, exit, clear

    sbd status [--full]
        Show status of SBD services in cluster and local device(s) configured.
        If --full is specified, also dump of SBD headers on device(s)
        will be shown.

    sbd config
        Show SBD configuration in cluster.

Examples:
    pcs stonith create MyStonith fence_virt pcmk_host_list=f1
```

LAB #5 - Mise en place d'un Serveur Apache Actif/Passif

Création du Stockage Partagé - Mutualisation du Stockage

Vous allez simuler un SAN avec iSCSI. Démarrez la machine virtuelle **iscsi** :

```
[root@node1 ~]# exit

[trainee@node1 ~]$ exit

desktop@serverXX:~$ VBoxManage startvm iscsi --type headless
```

Connectez-vous à la VM **iscsi** :

```
desktop@serverXX:~$ ssh -l trainee localhost -p 6022
```

Commencez par installer le paquet **scsi-target-utils** :

```
[trainee@iscsi ~]$ su -
...
[root@iscsi ~]# yum install -y epel-release
[root@iscsi ~]# yum install -y scsi-target-utils
```

Le paquet contient la commande **tgtd** et la commande **tgtadm**.

Les options de la commande **tgtd** sont :

```
[root@iscsi ~]# tgtd --help
Linux SCSI Target framework daemon, version 1.0.55

Usage: tgtd [OPTION]
-f, --foreground        make the program run in the foreground
-C, --control-port NNNN use port NNNN for the mgmt channel
-t, --nr_iothreads NNNN specify the number of I/O threads
-d, --debug debuglevel  print debugging information
```

```
-V, --version           print version and exit
-h, --help              display this help and exit
```

Les options de la commande **tgtadm** sont :

```
[root@iscsi ~]# tgtadm --help
Linux SCSI Target administration utility, version 1.0.55

Usage: tgtadm [OPTION]
--lld <driver> --mode target --op new --tid <id> --targetname <name>
    add a new target with <id> and <name>. <id> must not be zero.
--lld <driver> --mode target --op delete [--force] --tid <id>
    delete the specific target with <id>.
    With force option, the specific target is deleted
    even if there is an activity.
--lld <driver> --mode target --op show
    show all the targets.
--lld <driver> --mode target --op show --tid <id>
    show the specific target's parameters.
--lld <driver> --mode target --op update --tid <id> --name <param> --value <value>
    change the target parameters of the target with <id>.
--lld <driver> --mode target --op bind --tid <id> --initiator-address <address>
--lld <driver> --mode target --op bind --tid <id> --initiator-name <name>
    enable the target to accept the specific initiators.
--lld <driver> --mode target --op unbind --tid <id> --initiator-address <address>
--lld <driver> --mode target --op unbind --tid <id> --initiator-name <name>
    disable the specific permitted initiators.
--lld <driver> --mode logicalunit --op new --tid <id> --lun <lun>
  --backing-store <path> --bstype <type> --bsopts <bs options> --bsoflags <options>
    add a new logical unit with <lun> to the specific
    target with <id>. The logical unit is offered
    to the initiators. <path> must be block device files
    (including LVM and RAID devices) or regular files.
    bstype option is optional.
```

```
    bsopts are specific to the bstype.
    bsoflags supported options are sync and direct
    (sync:direct for both).
--lld <driver> --mode logicalunit --op delete --tid <id> --lun <lun>
    delete the specific logical unit with <lun> that
    the target with <id> has.
--lld <driver> --mode account --op new --user <name> --password <pass>
    add a new account with <name> and <pass>.
--lld <driver> --mode account --op delete --user <name>
    delete the specific account having <name>.
--lld <driver> --mode account --op bind --tid <id> --user <name> [--outgoing]
    add the specific account having <name> to
    the specific target with <id>.
    <user> could be <IncomingUser> or <OutgoingUser>.
    If you use --outgoing option, the account will
    be added as an outgoing account.
--lld <driver> --mode account --op unbind --tid <id> --user <name> [--outgoing]
    delete the specific account having <name> from specific
    target. The --outgoing option must be added if you
    delete an outgoing account.
--lld <driver> --mode lld --op start
    Start the specified lld without restarting the tgtd process.
--control-port <port> use control port <port>
--help
    display this help and exit

Report bugs to <stgt@vger.kernel.org>.
```

Activez et démarrez le service **tgtd** :

```
[root@iscsi ~]# systemctl enable tgtd
[root@iscsi ~]# systemctl restart tgtd
```

Configurez firewalld pour le service :

```
[root@iscsi ~]# firewall-cmd --add-service=iscsi-target --permanent
[root@iscsi ~]# firewall-cmd --reload
```

Créez d'abord une cible à laquelle vous pouvez rajouter le disque :

```
[root@iscsi ~]# tgtadm --lld iscsi --op new --mode target --tid 1 -T target
```

Consultez la configuration actuelle :

```
[root@iscsi ~]# tgtadm --lld iscsi --op show --mode target
Target 1: target
    System information:
        Driver: iscsi
        State: ready
    I_T nexus information:
    LUN information:
        LUN: 0
            Type: controller
            SCSI ID: IET     00010000
            SCSI SN: beaf10
            Size: 0 MB, Block size: 1
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: No
            SWP: No
            Thin-provisioning: No
            Backing store type: null
            Backing store path: None
            Backing store flags:
    Account information:
    ACL information:
```

Ajoutez maintenant le disque au premier LUN :

```
[root@iscsi ~]# tgtadm --lld iscsi --op new --mode logicalunit --tid 1 --lun 1 -b /dev/sdb
```

Consultez la configuration :

```
[root@iscsi ~]# tgtadm --lld iscsi --op show --mode target
Target 1: target
    System information:
        Driver: iscsi
        State: ready
    I_T nexus information:
    LUN information:
        LUN: 0
            Type: controller
            SCSI ID: IET     00010000
            SCSI SN: beaf10
            Size: 0 MB, Block size: 1
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: No
            SWP: No
            Thin-provisioning: No
            Backing store type: null
            Backing store path: None
            Backing store flags:
        LUN: 1
            Type: disk
            SCSI ID: IET     00010001
            SCSI SN: beaf11
            Size: 1074 MB, Block size: 512
            Online: Yes
            Removable media: No
```

```
            Prevent removal: No
            Readonly: Yes
            SWP: No
            Thin-provisioning: No
            Backing store type: rdwr
            Backing store path: /dev/sdb
            Backing store flags:
    Account information:
    ACL information:
```

Configurez maintenant le deuxième LUN :

```
[root@iscsi ~]# tgtadm --lld iscsi --op new --mode logicalunit --tid 1 --lun 2 -b /dev/sdb
```

Configurez le serveur pour que tous les clients puissent y avoir accès :

```
[root@iscsi ~]# tgtadm --lld iscsi --op bind --mode target --tid 1 -I ALL
```

Dernièrement configurez LUN1 et LUN2 en mode r/w :

```
[root@iscsi ~]# tgtadm --lld iscsi --mode logicalunit --op update --tid 1 --lun 1 --params readonly=0
[root@iscsi ~]# tgtadm --lld iscsi --mode logicalunit --op update --tid 1 --lun 2 --params readonly=0
```

Consultez la configuration :

```
[root@iscsi ~]# tgtadm --lld iscsi --op show --mode target
Target 1: target
    System information:
        Driver: iscsi
        State: ready
    I_T nexus information:
    LUN information:
        LUN: 0
            Type: controller
```

```
            SCSI ID: IET     00010000
            SCSI SN: beaf10
            Size: 0 MB, Block size: 1
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: No
            SWP: No
            Thin-provisioning: No
            Backing store type: null
            Backing store path: None
            Backing store flags:
        LUN: 1
            Type: disk
            SCSI ID: IET     00010001
            SCSI SN: beaf11
            Size: 1074 MB, Block size: 512
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: No
            SWP: No
            Thin-provisioning: No
            Backing store type: rdwr
            Backing store path: /dev/sdb
            Backing store flags:
        LUN: 2
            Type: disk
            SCSI ID: IET     00010002
            SCSI SN: beaf12
            Size: 1074 MB, Block size: 512
            Online: Yes
            Removable media: No
            Prevent removal: No
```

```
            Readonly: No
            SWP: No
            Thin-provisioning: No
            Backing store type: rdwr
            Backing store path: /dev/sdb
            Backing store flags:
    Account information:
    ACL information:
        ALL
```

Important - Notez que les SCSI ID et les SCSI SN des deux LUNs ne sont pas identiques. Dans l'état donc, le multipathing ne fonctionera pas car celui-ci necéssite à ce que les deux enregistrements soient identiques.

L'option **–params** de la commande **tgtadm** permet de modifier les paramètres de la configuration :

```
[root@iscsi ~]# tgtadm --lld iscsi --op update --mode logicalunit --tid 1 --lun 1 --params
vendor_id="I2TCH",product_id="MyISCSIDisk",product_rev="1.0",scsi_id="scsi001",scsi_sn="0123456789"
[root@iscsi ~]# tgtadm --lld iscsi --op update --mode logicalunit --tid 1 --lun 2 --params
vendor_id="I2TCH",product_id="MyISCSIDisk",product_rev="1.0",scsi_id="scsi001",scsi_sn="0123456789"
```

Consultez maintenant la configuration :

```
[root@iscsi ~]# tgtadm --lld iscsi --op show --mode target
Target 1: target
    System information:
        Driver: iscsi
        State: ready
    I_T nexus information:
    LUN information:
        LUN: 0
            Type: controller
```

```
            SCSI ID: IET     00010000
            SCSI SN: beaf10
            Size: 0 MB, Block size: 1
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: No
            SWP: No
            Thin-provisioning: No
            Backing store type: null
            Backing store path: None
            Backing store flags:
        LUN: 1
            Type: disk
            SCSI ID: scsi001
            SCSI SN: 0123456789
            Size: 1074 MB, Block size: 512
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: No
            SWP: No
            Thin-provisioning: No
            Backing store type: rdwr
            Backing store path: /dev/sdb
            Backing store flags:
        LUN: 2
            Type: disk
            SCSI ID: scsi001
            SCSI SN: 0123456789
            Size: 1074 MB, Block size: 512
            Online: Yes
            Removable media: No
            Prevent removal: No
```

```
            Readonly: No
            SWP: No
            Thin-provisioning: No
            Backing store type: rdwr
            Backing store path: /dev/sdb
            Backing store flags:
    Account information:
    ACL information:
        ALL
```

Vérifiez que le port **3260** est bien ouvert et en état d'écoute :

```
[root@iscsi ~]# netstat -apn | grep 3260
tcp        0      0 0.0.0.0:3260            0.0.0.0:*               LISTEN      4169/tgtd
tcp6       0      0 :::3260                 :::*                    LISTEN      4169/tgtd
```

Pour accéder à notre cible ISCSI, le client doit disposer d'un **initiateur**. Installez donc le paquet **iscsi-initiator-utils** :

```
[root@node1 ~]# yum -y install iscsi-initiator-utils
```

```
[root@node2 ~]# yum -y install iscsi-initiator-utils
```

Lancez maintenant la découverte des cibles sur le serveur iscsi :

```
[root@node1 ~]# iscsiadm --mode discovery --type sendtargets --portal 10.0.3.18
192.168.121.3:3260,1 target
```

```
[root@node2 ~]# iscsiadm --mode discovery --type sendtargets --portal 10.0.3.18
192.168.121.3:3260,1 target
```

Connectez-vous à la cible à partir de chaque noeud :

```
[root@node1 ~]# iscsiadm --mode node --targetname target --login
```

```
Logging in to [iface: default, target: target, portal: 10.0.3.18,3260] (multiple)
Login to [iface: default, target: target, portal: 10.0.3.18,3260] successful.
```

```
[root@node2 ~]# iscsiadm --mode node --targetname target --login
Logging in to [iface: default, target: target, portal: 10.0.3.18,3260] (multiple)
Login to [iface: default, target: target, portal: 10.0.3.18,3260] successful.
```

Consultez le journal **/var/log/messages** de chaque noeud :

```
[root@node1 ~]# tail /var/log/messages
Sep 16 17:15:54 node1 kernel: sd 3:0:0:1: [sdb] 16777216 512-byte logical blocks: (8.58 GB/8.00 GiB)
Sep 16 17:15:54 node1 kernel: sd 3:0:0:2: Attached scsi generic sg4 type 0
Sep 16 17:15:54 node1 kernel: sd 3:0:0:1: [sdb] Write Protect is off
Sep 16 17:15:54 node1 kernel: sd 3:0:0:2: [sdc] 16777216 512-byte logical blocks: (8.58 GB/8.00 GiB)
Sep 16 17:15:54 node1 kernel: sd 3:0:0:1: [sdb] Write cache: enabled, read cache: enabled, supports DPO and FUA
Sep 16 17:15:54 node1 kernel: sd 3:0:0:2: [sdc] Write Protect is off
Sep 16 17:15:54 node1 kernel: sd 3:0:0:2: [sdc] Write cache: enabled, read cache: enabled, supports DPO and FUA
Sep 16 17:15:54 node1 kernel: sd 3:0:0:1: [sdb] Attached SCSI disk
Sep 16 17:15:54 node1 kernel: sd 3:0:0:2: [sdc] Attached SCSI disk
Sep 16 17:15:55 node1 iscsid: Connection1:0 to [target: target, portal: 10.0.3.18,3260] through [iface: default]
is operational now
```

```
[root@node2 ~]# tail /var/log/messages
Aug 31 10:18:24 node2 kernel: sd 3:0:0:2: [sdc] 16777216 512-byte logical blocks: (8.58 GB/8.00 GiB)
Aug 31 10:18:24 node2 kernel: sd 3:0:0:1: [sdb] Write cache: enabled, read cache: enabled, supports DPO and FUA
Aug 31 10:18:24 node2 kernel: sd 3:0:0:2: [sdc] Write Protect is off
Aug 31 10:18:24 node2 kernel: sd 3:0:0:2: [sdc] Write cache: enabled, read cache: enabled, supports DPO and FUA
Aug 31 10:18:24 node2 kernel: sd 3:0:0:1: [sdb] Attached SCSI disk
Aug 31 10:18:24 node2 kernel: sd 3:0:0:2: [sdc] Attached SCSI disk
Aug 31 10:18:25 node2 iscsid: Connection1:0 to [target: target, portal: 10.0.3.18,3260] through [iface: default]
is operational now
Aug 31 10:19:03 node2 systemd: Starting Cleanup of Temporary Directories...
Aug 31 10:19:05 node2 systemd: Started Cleanup of Temporary Directories.
```

```
Aug 31 10:19:21 node2 sh: Sleeping '' ''
```

Important - Notez que les deux disques SCSI sont vus comme /dev/sdb et /dev/sdc.

A partir de node1.i2tch.loc, créez la partition sdb1 :

```
[root@node1 ~]# fdisk /dev/sdb
Welcome to fdisk (util-linux 2.23.2).

Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table
Building a new DOS disklabel with disk identifier 0x6f284e72.

Command (m for help): n
Partition type:
   p   primary (0 primary, 0 extended, 4 free)
   e   extended
Select (default p): p
Partition number (1-4, default 1):
First sector (2048-16777215, default 2048):
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-16777215, default 16777215):
Using default value 16777215
Partition 1 of type Linux and of size 8 GiB is set

Command (m for help): p

Disk /dev/sdb: 8589 MB, 8589934592 bytes, 16777216 sectors
Units = sectors of 1 * 512 = 512 bytes
```

```
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x6f284e72

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1            2048    16777215     8387584   83  Linux

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
[root@node1 ~]# partprobe
[root@node1 ~]# fdisk -l

Disk /dev/sda: 21.5 GB, 21474836480 bytes, 41943040 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x000c5a90

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1   *        2048      411647      204800   83  Linux
/dev/sda2          411648    20891647    10240000   83  Linux
/dev/sda3        20891648    25083903     2096128   82  Linux swap / Solaris
/dev/sda4        25083904    41943039     8429568    5  Extended
/dev/sda5        25085952    26109951      512000   83  Linux
/dev/sda6        26112000    27135999      512000   83  Linux

Disk /dev/sdb: 8589 MB, 8589934592 bytes, 16777216 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
```

```
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x6f284e72

   Device Boot      Start         End      Blocks   Id  System
/dev/sdb1            2048    16777215     8387584   83  Linux

Disk /dev/sdc: 8589 MB, 8589934592 bytes, 16777216 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk label type: dos
Disk identifier: 0x6f284e72

   Device Boot      Start         End      Blocks   Id  System
/dev/sdc1            2048    16777215     8387584   83  Linux
```

Créez ensuite un PV sur sdb1 :

```
[root@node1 ~]# pvcreate /dev/sdb1
  Physical volume "/dev/sdb1" successfully created.
```

Créez maintenant un VG :

```
[root@node1 ~]# vgcreate my_vg /dev/sdb1
  Volume group "my_vg" successfully created
```

Dernièrement créez le LV **my_lv** :

```
[root@node1 ~]# lvcreate -L450 -n my_lv my_vg
  Rounding up size to full physical extent 452.00 MiB
  Logical volume "my_lv" created.
```

Constatez la présence du volume logique :

```
[root@node1 ~]# lvs
  LV    VG    Attr       LSize   Pool Origin Data%  Meta%  Move Log Cpy%Sync Convert
  my_lv my_vg -wi-a----- 452.00m
```

Créez maintenant un système de fichiers ext4 sur le volume logique **my_lv** :

```
[root@node1 ~]# mkfs.ext4 /dev/my_vg/my_lv
mke2fs 1.42.9 (28-Dec-2013)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
Stride=0 blocks, Stripe width=0 blocks
115824 inodes, 462848 blocks
23142 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=34078720
57 block groups
8192 blocks per group, 8192 fragments per group
2032 inodes per group
Superblock backups stored on blocks:
    8193, 24577, 40961, 57345, 73729, 204801, 221185, 401409

Allocating group tables: done
Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done
```

Installez Apache sur chaque noeud :

```
[root@node1 ~]# yum install -y httpd wget
```

```
[root@node2 ~]# yum install -y httpd wget
```

Afin que l'agent de la ressource Apache puisse obtenir le statut de chaque Apache, ajoutez les lignes suivantes à la fin du fichier **/etc/httpd/conf/httod.conf** de chaque noeud :

```
[root@node1 ~]# vi /etc/httpd/conf/httpd.conf
[root@node1 ~]# cat /etc/httpd/conf/httpd.conf
...
<Location /server-status>
    SetHandler server-status
    Require local
</Location>
```

```
[root@node2 ~]# vi /etc/httpd/conf/httpd.conf
[root@node2 ~]# cat /etc/httpd/conf/httpd.conf
...
<Location /server-status>
    SetHandler server-status
    Require local
</Location>
```

L'agent de la ressource Apache n'utilise pas systemd. Pour cette raison vous devez éditer le script de logrotate afin que celui-ci n'utilise pas la commande systemctl pour recharger Apache :

```
[root@node1 ~]# vi /etc/logrotate.d/httpd
[root@node1 ~]# cat /etc/logrotate.d/httpd
/var/log/httpd/*log {
    missingok
    notifempty
    sharedscripts
    delaycompress
    postrotate
    #    /bin/systemctl reload httpd.service > /dev/null 2>/dev/null || true
    /usr/sbin/httpd -f /etc/httpd/conf/httpd.conf -c "PidFile /var/run/httpd.pid" -k graceful > /dev/null
2>/dev/null || true
    endscript
```

```
}
```

```
[root@node2 ~]# vi /etc/logrotate.d/httpd
[root@node2 ~]# cat /etc/logrotate.d/httpd
/var/log/httpd/*log {
    missingok
    notifempty
    sharedscripts
    delaycompress
    postrotate
    #    /bin/systemctl reload httpd.service > /dev/null 2>/dev/null || true
    /usr/sbin/httpd -f /etc/httpd/conf/httpd.conf -c "PidFile /var/run/httpd.pid" -k graceful > /dev/null
2>/dev/null || true
    endscript
}
```

Créez le fichier **index.html** sur le volume logique **my_lv** :

```
[root@node1 ~]# mount /dev/my_vg/my_lv /var/www/
You have new mail in /var/spool/mail/root
[root@node1 ~]# mkdir /var/www/html
[root@node1 ~]# mkdir /var/www/cgi-bin
[root@node1 ~]# mkdir /var/www/error
[root@node1 ~]# restorecon -R /var/www
[root@node1 ~]# cat <<-END >/var/www/html/index.html
> <html>
> <body>Hello</body>
> </html>
> END
[root@node1 ~]# umount /var/www
```

Afin d'accorder le contrôle de l'activation du groupe de volumes **my_vg** au cluster et non au système d'exploitation, afin d'éviter toute possibilité de corruption des méta-données, il convient de modifier la directive **volume_list** du fichier **/etc/lvm/lvm.conf** de chaque noeud.

Premièrement, modifiez la valeur de la directive **use_lvmetad** dans le fichier **/etc/lvm/lvm.conf** de chaque noeud :

```
[root@node1 ~]# vi /etc/lvm/lvm.conf
[root@node1 ~]# cat /etc/lvm/lvm.conf | grep "use_lvmetad ="
    use_lvmetad = 0
```

```
[root@node2 ~]# vi /etc/lvm/lvm.conf
[root@node2 ~]# cat /etc/lvm/lvm.conf | grep "use_lvmetad ="
    use_lvmetad = 0
```

Arrêtez et désactivez tous les processus éventuels de **use_lvmetad** sur chaque noeud :

```
[root@node1 ~]# lvmconf --enable-halvm --services --startstopservices
Warning: Stopping lvm2-lvmetad.service, but it can still be activated by:
  lvm2-lvmetad.socket
Removed symlink /etc/systemd/system/sysinit.target.wants/lvm2-lvmetad.socket.
```

```
[root@node2 ~]# lvmconf --enable-halvm --services --startstopservices
Warning: Stopping lvm2-lvmetad.service, but it can still be activated by:
  lvm2-lvmetad.socket
Removed symlink /etc/systemd/system/sysinit.target.wants/lvm2-lvmetad.socket.
```

Modifiez la directive **volume_list** dans le fichier **/etc/lvm/lvm.conf** de chaque neoud :

```
[root@node1 ~]# vi /etc/lvm/lvm.conf
[root@node1 ~]# cat /etc/lvm/lvm.conf | grep my_vg
    volume_list = [ ]
```

```
[root@node2 ~]# vi /etc/lvm/lvm.conf
[root@node2 ~]# cat /etc/lvm/lvm.conf | grep my_vg
    volume_list = [ ]
```

Regénérez un initramfs sur chaque noeud afin de prendre en compte ces modifications :

```
[root@node1 ~]# dracut -H -f /boot/initramfs-$(uname -r).img $(uname -r)
```

```
[root@node2 ~]# dracut -H -f /boot/initramfs-$(uname -r).img $(uname -r)
```

Re-démarrez chaque noeud :

```
[root@node1 ~]# shutdown -r now
```

```
[root@node2 ~]# shutdown -r now
```

Création des Ressources du Cluster

Créez la ressource cluster **my_lvm** :

```
[root@node1 ~]# pcs resource create my_lvm LVM volgrpname=my_vg exclusive=true --group apachegroup
Assumed agent name 'ocf:heartbeat:LVM' (deduced from 'LVM')
```

Contrôlez le statut du cluster :

```
[root@node1 ~]# pcs status
Cluster name: my_cluster
Stack: corosync
Current DC: node1.i2tch.loc (version 1.1.18-11.el7_5.3-2b07d5c5a9) - partition with quorum
Last updated: Tue Sep 18 06:56:44 2018
Last change: Tue Sep 18 06:56:10 2018 by root via cibadmin on node1.i2tch.loc

2 nodes configured
2 resources configured

Online: [ node1.i2tch.loc node2.i2tch.loc ]

Full list of resources:
```

```
 myapc  (stonith:fence_apc_snmp):    Stopped
 Resource Group: apachegroup
     my_lvm (ocf::heartbeat:LVM):    Started node1.i2tch.loc

Failed Actions:
* myapc_start_0 on node1.i2tch.loc 'unknown error' (1): call=14, status=Error, exitreason='',
    last-rc-change='Tue Sep 18 06:48:02 2018', queued=0ms, exec=14307ms
* myapc_start_0 on node2.i2tch.loc 'unknown error' (1): call=6, status=Error, exitreason='',
    last-rc-change='Tue Sep 18 06:48:14 2018', queued=2ms, exec=16999ms


Daemon Status:
  corosync: active/enabled
  pacemaker: active/enabled
  pcsd: active/enabled
```

Créez ensuite les ressources **my_fs** ainsi que **VirtualIP** :

```
[root@node1 ~]# pcs resource create my_fs Filesystem device="/dev/my_vg/my_lv" directory="/var/www" fstype="ext4"
--group apachegroup
Assumed agent name 'ocf:heartbeat:Filesystem' (deduced from 'Filesystem')
[root@node1 ~]#
[root@node1 ~]# pcs resource create VirtualIP IPaddr2 ip=10.0.3.100 cidr_netmask=24 --group apachegroup
Assumed agent name 'ocf:heartbeat:IPaddr2' (deduced from 'IPaddr2')
```

Dernièrement, créez la ressource **Website** :

```
[root@node1 ~]# pcs resource create Website apache configfile="/etc/httpd/conf/httpd.conf"
statusurl="http://127.0.0.1/server-status" --group apachegroup
Assumed agent name 'ocf:heartbeat:apache' (deduced from 'apache')
```

Contrôlez de nouveau le statut du cluster et **noter** le neoud sur lequel ont été démarrés les services (dans ce cas **node1.i2tch.loc**) :

```
[root@node1 ~]# pcs status
```

```
Cluster name: my_cluster
Stack: corosync
Current DC: node1.i2tch.loc (version 1.1.18-11.el7_5.3-2b07d5c5a9) - partition with quorum
Last updated: Tue Sep 18 07:34:49 2018
Last change: Tue Sep 18 07:34:44 2018 by root via cibadmin on node1.i2tch.loc

2 nodes configured
5 resources configured

Online: [ node1.i2tch.loc node2.i2tch.loc ]

Full list of resources:

 myapc  (stonith:fence_apc_snmp):    Stopped
 Resource Group: apachegroup
     my_lvm (ocf::heartbeat:LVM):    Started node1.i2tch.loc
     my_fs  (ocf::heartbeat:Filesystem):    Started node1.i2tch.loc
     VirtualIP  (ocf::heartbeat:IPaddr2):    Started node1.i2tch.loc
     Website    (ocf::heartbeat:apache):    Started node1.i2tch.loc

Failed Actions:
* myapc_start_0 on node2.i2tch.loc 'unknown error' (1): call=22, status=Error, exitreason='',
    last-rc-change='Tue Sep 18 07:24:17 2018', queued=0ms, exec=13456ms
* myapc_start_0 on node1.i2tch.loc 'unknown error' (1): call=6, status=Error, exitreason='',
    last-rc-change='Tue Sep 18 07:24:03 2018', queued=0ms, exec=13600ms


Daemon Status:
  corosync: active/enabled
  pacemaker: active/enabled
  pcsd: active/enabled
```

Contrôlez la présence de l'adresse IP 10.0.3.100 sur le noeud concerné :

```
[root@node1 ~]# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN qlen 1
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:10:90:fc brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic enp0s3
       valid_lft 85984sec preferred_lft 85984sec
    inet6 fe80::2d4d:e50a:4f0e:65d3/64 scope link
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master bond0 state UP qlen 1000
    link/ether 08:00:27:16:89:9f brd ff:ff:ff:ff:ff:ff
4: enp0s9: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master bond0 state UP qlen 1000
    link/ether 08:00:27:16:89:9f brd ff:ff:ff:ff:ff:ff
5: bond0: <BROADCAST,MULTICAST,MASTER,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP qlen 1000
    link/ether 08:00:27:16:89:9f brd ff:ff:ff:ff:ff:ff
    inet 10.0.3.15/24 brd 10.0.3.255 scope global bond0
       valid_lft forever preferred_lft forever
    inet 10.0.3.200/24 brd 10.0.3.255 scope global secondary bond0
       valid_lft forever preferred_lft forever
    inet6 fe80::aadb:c074:fbc7:5e94/64 scope link tentative dadfailed
       valid_lft forever preferred_lft forever
    inet6 fe80::5915:b321:a5ae:321e/64 scope link tentative dadfailed
       valid_lft forever preferred_lft forever
    inet6 fe80::138a:5c7a:1284:aa3d/64 scope link tentative dadfailed
       valid_lft forever preferred_lft forever
```

Arrêtez le noeud hébergeant les services :

```
[root@node1 ~]# shutdown - h now
```

Consultez le statut du cluster et notez le basculement des services :

```
[root@node2 ~]# pcs status
Cluster name: my_cluster
Stack: corosync
Current DC: node1.i2tch.loc (version 1.1.18-11.el7_5.3-2b07d5c5a9) - partition with quorum
Last updated: Tue Sep 18 07:50:18 2018
Last change: Tue Sep 18 07:34:44 2018 by root via cibadmin on node1.i2tch.loc

2 nodes configured
5 resources configured

Online: [ node1.i2tch.loc node2.i2tch.loc ]

Full list of resources:

 myapc  (stonith:fence_apc_snmp):    Stopped
 Resource Group: apachegroup
     my_lvm (ocf::heartbeat:LVM):    Started node2.i2tch.loc
     my_fs  (ocf::heartbeat:Filesystem):    Started node2.i2tch.loc
     VirtualIP  (ocf::heartbeat:IPaddr2):    Started node2.i2tch.loc
     Website    (ocf::heartbeat:apache):    Starting node2.i2tch.loc

Failed Actions:
* myapc_start_0 on node2.i2tch.loc 'unknown error' (1): call=22, status=Error, exitreason='',
    last-rc-change='Tue Sep 18 07:38:00 2018', queued=1ms, exec=15975ms
* myapc_start_0 on node1.i2tch.loc 'unknown error' (1): call=6, status=Error, exitreason='',
    last-rc-change='Tue Sep 18 07:37:47 2018', queued=0ms, exec=13536ms


Daemon Status:
  corosync: active/enabled
  pacemaker: active/enabled
  pcsd: active/enabled
```

Démarrez node1. Passez la machine virtuelle en mode graphique avec les commandes suivantes :

```
[root@node1 ~]# rm -rf /etc/systemd/system/default.target
[root@node1 ~]# ln -s /lib/systemd/system/graphical.target /etc/systemd/system/default.target
[root@node1 ~]# shutdown -r now
```

Ouvrez un navigateur Internet et ouvrez l'URL **https://node1.i2tch.loc:2224**. Connectez-vous avec l'utilisateur **hacluster**.

Pour plus d'information concernant l'interface HTML, consultez ce **lien**.
