Dernière mise-à-jour : 2021/01/24 12:16

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

Introduction

Un cluster est ce que l'on appelle une **grappe** de serveurs. Chaque membre d'une grappe est un **noeud** ou **membre**. Il existe quatre types majeurs de cluster :

- un **cluster de stockage** - permet une écriture simultanée sur un système de fichiers partagé tel Red Hat GFS (*Global File System*)
- un **cluster à haute disponibilité** ou **cluster failover** - permet une augmentation de la disponibilité des services,
- un **cluster de répartition** - permet une répartition de la charge sur plusieurs noeuds,
- un **cluster à haute performance** - permet une utilisation des noeuds du cluster pour effectuer des calculs simultanés.

Red Hat Cluster Suite (RHCS) est un ensemble de logiciels fournissant les composants suivants :

- Serveur virtuel Linux,
- Gestionnaire de l'infrastructure du cluster,
- Gestionnaire des services à haute disponibilité,
- Outils d'administration du cluster.

De base, RHCS n'inclut pas les composants suivants :

Red Hat GFS

Red Hat GFS est un système de fichiers en cluster qui utilise :

- des **métadonnées distribués**,

- plusieurs **fichiers de journaux**,
- un **gestionnaire de verrouillage**,

Les modifications aux données effectuées par un noeud sur un système de fichiers GFS sont immédiatement visibles aux autres noeuds

Le système de fichiers GFS fournit une image unique du système de fichiers pour tous les noeuds permettant, entre autre :

- la simplification de l'infrastructure de données,
- la simplification de l'installation et de l'application de correctifs,
- l’absence de duplication des données de l'application,
- l'accès concurrent aux données en lecture/écriture par plusieurs clients,
- la simplification de la restauration de pertes de données,
- la maximisation de l'utilisation des ressources de stockage,
- la réduction des coûts d'administration du stockage,
- la prise en charge du multipathing,
- la gestion des volumes via CLVM.

Cluster Logical Volume Manager

CLVM fournit des volumes logique **LVM2** au niveau d'un cluster et utilise de daemon **clvmd** afin ajouter des extensions cluster aux outils LVM classiques. Avec des modifications minimes au fichier de configuration LVM - **/etc/lvm/lvm.conf** - il est ainsi possible d'utiliser es commandes classiques de LVM2 afin de configurer et gérer des volumes logiques au niveau du cluster. Le daemon clvmd fonctionne sur chaque noeud et distribue les mises-à-jour des métadonnées LVM aux autres noeuds lors d'une modification. Lors d'une modification, l'accès au stockage physique est verrouillé en utilisant le gestionnaire de verrouillage **DLM** du cluster.

Global Network Block Device

GNBD fournit un accès à Red Hat GFS aux périphériques blocs en utilisant le protocole TCP. GNBD est souvent utilisé quand la mise en place de la technologie fibre channel est trop couteuse.

GNBD consiste en deux composants logiciels :

- le **client** GNBD,
- le **serveur** GNBD.

Le client GNBD fonctionne sur un noeud et importe un périphérique bloc exporté par le serveur GNBD qui fonctionne sur un autre noeud. Le périphérique bloc exporté par le serveur GNBD peut être soit un périphérique bloc local soit un SAN.

Les Composants de RHCS

Serveur Virtuel Linux

Avec le **Serveur Virtuel Linux**, connu sous le nom **LVS**, la répartition de charge est gérée par un **directeur** (*Load Balancer*) qui est un **routeur** LVS **actif**. En général, un routeur LVS passif est configuré afin de prendre le relais du routeur actif an cas de défaillance. L'ensemble du LVS est vu de l'extérieur comme un seul serveur.

Dans RHCS, le daemon **pulse**, le **Gestionnaire de Ressources**, est démarré sur le routeur passif et sur le routeur actif. Le daemon pulse sur le routeur passif envoie un signal **Heartbeat** à l'interface réseau public du routeur actif afin de s'assurer que ce dernier fonctionne toujours. Le daemon pulse sur le routeur actif démarre le daemon **lvs** qui répond au Heartbeat du routeur passif.

Le daemon lvs appelle l'utilitaire **ipvsadm** afin de mettre en place et maintenir à jour une table de routage appelée **IPVS** (*IP Virtual Server*) puis démarre une instance du service **nanny** par serveur virtuel, aussi appelé un **service**, configuré sur chaque serveur physique.

Chaque instance de **nanny** surveille son serveur virtuel. Dans le cas d'une défaillance, nanny demande à lvs d'appeler ipvsadm afin de supprimer de la table IPVS le serveur physique sur lequel fonctionne le serveur virtuel.

Si le routeur passif ne reçoit pas de réponse du serveur actif, il initialise un processus de **Basculement** (*Failover*) en envoyant une instruction d'arrêt du daemon lvs sur le routeur actif et en appelant **send_arp** afin de configurer ses interfaces réseaux physiques avec les adresses IP virtuelles du serveur actif.

LVS ne contient pas de composant de partage de données. De ce fait, soit les données doivent être synchronisées entre le routeur actif et le routeur passif avec, par exemple, **rsync**, soit il est nécessaire de mettre en place une mutualisation de stockage.

Gestionnaire de l'Infrastructure du Cluster

Gestionnaire du Cluster

La gestion du cluster est la tâche de **CMAN** (*Cluster MANager*). CMAN est un gestionnaire distribué et de ce fait fonctionne sur chaque neoud.

CMAN est responsable du suivi du **Quorum**. Si le cluster n'a pas Quorum, toute activité est arrêtée. Le Quorum est déterminé par des messages échangés entre les noeuds via Ethernet ou à travers un **Disque Quorum**. Pour que le Quorum soit obtenu, deux cas sont possibles :

- dans le cas du Quorum via Ethernet - 50% des noeuds + 1 doivent être actifs,
- dans le cas du Quorum via le disque Quorum - selon des conditions spécifiées par l'utilisateur.

Par défaut, chaque noeud possède une vote. Par contre il est possible d'augmenter le nombre de votes pour un noeud spécifique.

CMAN assure également le suivi des adhésions au cluster en analysant les messages des autres noeuds et informe ensuite les autres composants de toute modification afin qu'ils prennent les actions adéquates. Par exemple, si un noeud rejoint le cluster et monte un système de fichier GFS déjà monté par d'autres noeuds, un journal supplémentaire est créé et un gestionnaire de verrouillage est démarré.

Si un noeud reste silencieux pendant trop longtemps, les noeuds qui s'entendent forment un groupe. Si ce groupe a plus de 50% des votes attendues, le groupe gagne et le cluster continue à fonctionner. Le groupe va ensuite appellé le composant **Fencing** qui déconnecte le noeud.

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

/etc/sysconfig/cman

```
# CMAN_CLUSTER_TIMEOUT -- amount of time to wait for joinging a cluster
#     before giving up.  If CMAN_CLUSTER_TIMEOUT is positive, then we will
#     wait CMAN_CLUSTER_TIMEOUT seconds before giving up and failing when
#     a cluster is not joined.  If CMAN_CLUSTER_TIMEOUT is zero, then
#     wait indefinately for a cluster join.  If CMAN_CLUSTER_TIMEOUT is
#     negative, do not check to see that the cluster has been joined
```

```
#CMAN_CLUSTER_TIMEOUT=60

# CMAN_QUORUM_TIMEOUT -- amount of time to wait for a quorate cluster on
#     startup quorum is needed by many other applications, so we may as
#     well wait here.  If CMAN_QUORUM_TIMEOUT is zero, quorum will
#     be ignored.
#CMAN_QUORUM_TIMEOUT=45

# CMAN_SHUTDOWN_TIMEOUT -- amount of time to wait for cman to become a
#     cluster member before calling cman_tool leave during shutdown.
#     The default is 60 seconds
#CMAN_SHUTDOWN_TIMEOUT=60

# CMAN_NOTIFYD_START - control the startup behaviour for cmannotifyd
# the variable can take 3 values:
# yes                   | will always start cmannotifyd
# no                    | will never start cmannotifyd
# conditional (default) | will start cmannotifyd only if scriptlets
#                         are found in /etc/cluster/cman-notify.d
#CMAN_NOTIFYD_START=conditional

# CMAN_SSHD_START - control sshd startup behaviour
# the variable can take 2 values:
# yes                   | cman will start sshd as early as possible
# no (default)          | cman will not start sshd
#CMAN_SSHD_START=no

# DLM_CONTROLD_OPTS -- allow extra options to be passed to dlm_controld daemon.
#DLM_CONTROLD_OPTS=""

# Allow tuning of DLM kernel config.
# do NOT change unless instructed to do so.
#DLM_LKBTBL_SIZE=""
#DLM_RSBTBL_SIZE=""
```

```
#DLM_DIRTBL_SIZE=""
#DLM_TCP_PORT=""

# FENCE_JOIN_TIMEOUT -- seconds to wait for fence domain join to
#     complete.  If the join hasn't completed in this time, fence_tool join
#     exits with an error, and this script exits with an error.  To wait
#     indefinitely set the value to -1.
#FENCE_JOIN_TIMEOUT=20

# FENCED_MEMBER_DELAY -- amount of time to delay fence_tool join to allow
#     all nodes in cluster.conf to become cluster members.  In seconds.
#FENCED_MEMBER_DELAY=45

# FENCE_JOIN -- boolean value used to control whether or not this node
#     should join the fence domain. If FENCE_JOIN is set to "no", then
#     the script will not attempt to the fence domain. If FENCE_JOIN is
#     set to "yes", then the script will attempt to join the fence domain.
#     If FENCE_JOIN is set to any other value, the default behavior is
#     to join the fence domain (equivalent to "yes").
#     When setting FENCE_JOIN to "no", it is important to also set
#     DLM_CONTROLD_OPTS="-f0" (at least) for correct operation.
#     Please note that clusters without fencing are not
#     supported by Red Hat except for MRG installations.
#FENCE_JOIN="yes"

# FENCED_OPTS -- allow extra options to be passed to fence daemon.
#FENCED_OPTS=""

# NETWORK_BRIDGE_SCRIPT -- script to use for xen network bridging.
#     This script must exist in the /etc/xen/scripts directory.
#     The default script is "network-bridge".
#NETWORK_BRIDGE_SCRIPT="network-bridge"

# CLUSTERNAME -- override clustername as specified in cluster.conf
```

```
#CLUSTERNAME=""

# NODENAME -- specify the nodename of this node. Default autodetected
#NODENAME=""

# CONFIG_LOADER -- select default config parser.
# This can be:
# xmlconfig       - read directly from cluster.conf and use ricci as default
#                   config propagation method. (default)
#CONFIG_LOADER=xmlconfig

# CONFIG_VALIDATION -- select default config validation behaviour
# This can be:
# FAIL - Use a very strict checking. The config will not be loaded if there
#        for any kind of warnings/errors.
# WARN - Same as FAIL, but will allow the config to load (this is temporary
#        the default behaviour)
# NONE - Disable config validation. Highly discouraged.
#CONFIG_VALIDATION=WARN

# CMAN_LEAVE_OPTS -- allows extra options to be passed to cman_tool when leave
#     operation is performed.
#CMAN_LEAVE_OPTS=""

# INITLOGLEVEL -- select how verbose the init script should be
# possible values:
# quiet           - only one line notification for start/stop operations
# terse (default) - show only required activity
# full            - show everything
#INITLOGLEVEL=terse
```

Important - La modification des directives **CMAN_CLUSTER_TIMEOUT** et **CMAN_QUORUM_TIMEOUT** permettent la configuration du timeout de cman ainsi que le timeout du quorum. Les valeurs par défaut sont indiquées dans le fichier lui-même.

Le Disque Quorum

Le daemon **qdiskd** fournit des heuristiques supplémentaires pour déterminer la santé des noeuds. Le mécanisme se base sur l'utilisation d'un périphérique de type bloc appelé le **Disque Quorum**. Les points clefs sont :

- chaque noeud doit avoir une vote,
- le nombre de noeuds est limité à 16 par cluster,
- le timeout de **qdiskd** doit être la moitié du timeout de **cman**,
- la méthode de fencing doit être **Power Fencing**,
- le périphérique bloc utilisé doit être partagé en mode **lecture/écriture simultané** pour tous les noeuds,
- la taille recommandée est de 10 Mo.

Concrètement, l'utilisation d'un Disque Quorum permet de configurer un quorum atypique, par exemple un quorum où un seul noeud fonctionne.

Pour cette raison le Disque Quorum est utilisé en règle générale dans le cas d'un cluster à peu de nœuds. Considérez le situation où une panne de réseau a lieu entre deux noeuds dans un cluster à deux noeuds. Dans ce cas chaque noeud considère que le cluster lui appartient et essaie de fencé l'autre noeud. Cette situation s'appelle **split brain**.

La seule solution à ce problème en utilisant un Quorum via Ethernet est de ne pas tenir compte du quorum en indiquant **<cman expected_votes="1" two_node="1"/>** dans le fichier de configuration **/etc/cluster/cluster.conf** de chaque noeud.

Important - Le fichier **/etc/cluster/cluster.conf** est étudié dans le détail plus tard dans de ce cours.

L'autre façon de palier à ce problème est d'utiliser un Disque Quorum. Dans notre cas, chaque noeud à une vote de 1 tandis que le Disque Quorum lui-même a aussi une vote de 1.

Important - Dans un cas d'un cluster avec 3 noeuds ayant chacun une vote par exemple, le Disque Quorum aura 2 votes, c'est-à-dire le nombre total des votes des noeuds moins 1. De cette façon, il suffit de 3 votes sur 5 pour que le quorum soit atteint, autrement dit le Disque Quorum et 1 neoud.

Revenons à notre cluster de deux noeuds. Lors de l'utilisation d'un Disque Quorum, le qdiskd sur chaque noeud évalue son état et inscrit les informations d'état dans une partie du Disque Quorum. Le service qdiskd sur l'autre noeud vérifie ensuite les informations du premier noeud et vice-versa. Si, après plusieurs tentatives, un service qdiskd sur un neoud ne peut pas mettre à jour ses informations d'état, ceci est détecté par l'autre service qdiskd qui demande à ce que le noeud problématique soit fenced.

Important - Dans la version 6 de Red Hat et de CentOS, le service **qdiskd** est géré directement par le service **cman**. Notez que vous pouvez obtenir plus d'informations sur le Disque Quorum en lisant le manuel **qdisk(5)**.

Gestionnaire du Verrouillage

La gestion du verrouillage est la tâche de **DLM** (*Distributed Lock Manager*). Comme son nom indique DLM est un gestionnaire distribué et de ce fait fonctionne sur chaque noeud. GFS utilise des verrous de DLM afin de synchroniser l'accès aux métadonnées sur un stockage partagé. CLVM utilise des verrous de DLM afin de synchroniser des mises à jour aux volumes logiques et groupes de volumes sur un stockage partagé.

Fencing

Fencing est géré par le deamon **fenced** et consiste en la déconnexion d'un noeud d'un stockage partagé du cluster afin de préserver l'intégrité des données.

Ils existe plusieurs types de Fencing dont :

- **Power Fencing** — utilise un contrôleur de courant pour éteindre un noeud,

- **Fibre Channel Switch Fencing** — désactive le port Fibre Channel connectant le stockage au noeud,
- **GNBD Fencing** — désactive l'accès à un serveur GNBD.

> **Important** - Certains **périphériques fence intégrés** sont capables d'arrêter un noeud en 4 ou 5 secondes. D'autres se reposent sur un arrêt géré par le système d'exploitation du noeud. Dans ce cas l'arrêt peut durer beaucoup plus longtemps. La liste des périphériques fence intégrés actuellement supportés par Red Hat se trouve à l'adresse : http://www.redhat.com/cluster_suite/hardware/

Un noeud peut être configuré avec plusieurs méthodes de Fencing en **cascade**. Si la première méthode échoue, la deuxième est appelée et ainsi de suite.

L'utilisation d'un périphérique fence intégré nécessite l'arrêt du service **ACPI Soft-Off**. L'arrêt de ce service peut être obtenu en utilisant la commande **chkconfig** :

```
[root@centos6 ~]# chkconfig --list acpid
acpid           0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
[root@centos6 ~]# chkconfig --del acpid
[root@centos6 ~]# chkconfig --list acpid
le service acpid prend en charge chkconfig, mais il n'est enregistré à aucun niveau (exécutez « chkconfig --add
acpid »)
```

> **Important** - Il existe aussi la possibilité de modifier le BIOS du noeud ou d'ajouter **acpi=off** à la ligne **kernel** du fichier **/boot/grub/grub.conf** afin d'obtenir le même résultat mais l'utilisation de la commande **chkconfig** reste la méthode conseillée.

Gestionnaire de la Configuration du Cluster

La gestion de la configuration du cluster est la tâche de **CSS** (*Cluster Configuration System*). CSS est un gestionnaire distribué et de ce fait fonctionne sur chaque neoud.

CSS assure que le fichier de configuration du cluster sur chaque noeud est à jour. Ce fichier - **/etc/cluster/cluster.conf** - et au format **XML** et décrit les caractéristiques suivantes :

- le **nom du cluster**,
- le **niveau de révision du fichier de configuration**,
- la ou les **méthode(s) de fencing**,
- le **nom de chaque noeud** du cluster ainsi que son **ID**,
- le **nombre de votes du Quorum**,
- le **périphérique de fencing**,
- les **ressources gérées**.

Gestionnaire des Services à Haute Disponibilité

La gestion des services à haute disponibilité du cluster est la tâche de **rgmanager**. Dans un cluster, une application peut être configurée avec d'autres ressources afin de créer un **Service à Haute Disponibilité**. Un service à haute disponibilité peut passer d'un nœud à un autre sans interruption. Un service à haute disponibilité est configuré dans le fichier de configuration de chaque nœud d'un **Domaine de Basculement** (*Failover Domain*), un sous-ensemble de nœuds d'un cluster.

> **Important** - Par défaut, tous les 10 secondes, **rgmanager** recherche dans l'arbre des ressources celles qui ont dépassé l'intervalle de vérification. Chaque agent de ressource spécifie l'intervalle de vérification pour la ressource concernée. Cette intervalle peut être modifiée en éditant le fichier **/etc/cluster/cluster.conf** en spécifiant une intervalle pour une ressource spécifique avec la balise **<action>** : **<action name="status" depth="*" interval="10" />**. Certains agents fournissent plusieurs **profondeurs** (*depths*) de vérification. Plus profonde est la vérification, plus complet est la vérification. Dans l'exemple précédent, l'intervalle concerne tous les profondeurs.

Outils d'administration du cluster

Conga

Conga est un ensemble intégré de composants qui fournit une configuration et une gestion des clusters et du stockage Red Hat. Conga permet :

- une gestion du cluster et du stockage via une interface Web,
- un déploiement automatisé des données du cluster et des paquets de support,
- une intégration avec des clusters existants,
- l'intégration du statut et des fichiers journaux du cluster,
- un contrôle des autorisations des utilisateurs.

Les principaux composants de Conga sont :

- **luci** - un serveur qui communique avec plusieurs clusters et ordinateurs via ricci,
- **ricci** - un agent démarré sur chaque ordinateur géré par Conga.

> **Important** - L'outil **system-config-cluster** contenant le **Cluster Configuration Tool** et le **Cluster Status Tool**, disponible dans RHEL 5, n'est pas disponible dans RHEL6.

En Ligne de Commande

RHCS propose également des outils en ligne de commande :

Commande	Description
ccs_tool	Outil système de configuration du cluster
cman_tool	Outil de gestion du cluster
fence_tool	Outil Fencing
clustat	Utilitaire de statut du cluster
clusvcadm	Utilitaire d'administration des services utilisateur du cluster

Installation du Matériel

Un cluster comprend généralement le matériel suivant :

- Noeuds - ordinateurs capable d'exécuter RHEL 6 et munis d'au moins **1 Go** de RAM,
- Un commutateur vers le réseau public pour permettre aux clients d'avoir accès au cluster,
- Un commutateur vers le réseau privé pour permettre aux neouds de se communiquer,
- Un commutateur d'alimentation pour le fencing,
- Un commutateur fibre channel pour le fencing,
- Du stockage partagé.

Installer le Logiciel du Module Red Hat High Availability

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

```
[root@centos6 ~]# yum -y install rgmanager lvm2-cluster gfs2-utils wget
```

Démarrer l'Agent ricci

Dans Red Hat 6 , l'agent **ricci** remplace l'agent **ccsd** disponible dans les versions précédentes. L'agent ricci 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 ricci est démarré :

```
[root@centos6 ~]# service ricci status
ricci est arrêté
```

Démarrez ricci puis configurez-le pour un démarrage automatique en utilisant la commande **chkconfig** :

```
[root@centos6 ~]# service ricci start
```

```
Démarrage de oddjobd :                                     [  OK  ]
generating SSL certificates...  done
Generating NSS database...  done
Démarrage de ricci :                                       [  OK  ]
[root@node1 ~]# chkconfig --list ricci
ricci           0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
[root@node1 ~]# chkconfig --levels 2345 ricci on
[root@node1 ~]# chkconfig --list ricci
ricci           0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
```

A partir de Red Hat Enterprise Linux 6.1, lors de la première propagation des modifications de la mise à jour de la configuration du cluster, ricci requiert un mot de passe. Pour fixer le mot de passe de ricci, utilisez la commende **passwd** :

```
[root@centos6 ~]# passwd ricci
Changement de mot de passe pour l'utilisateur ricci.
Nouveau mot de passe : ricci
MOT DE PASSE INCORRECT : trop court
MOT DE PASSE INCORRECT : est trop simple
Retapez le nouveau mot de passe : ricci
passwd : mise à jour réussie de tous les jetons d'authentification.
```

> **Important** - Dans le cas de notre exemple, le mot de passe est **ricci**. 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 ricci !

Pré-Configurer les Noeuds

Considérations Générales

En implémentant une solution de cluster avec Red Hat High Availability, il est important de savoir que :

- le nombre maximum de noeuds supportés par Red Hat High Availability est de **16**,
- Red Hat ne supporte pas de clusters multi-sites. Seuls les clusters mono-sites sont supportés,
- Red Hat ne supporte pas l'utilisation du système de fichiers GFS2 sur **un seul et unique noeud**,
- afin de maintenir l'intégrité des données, seul un noeud à la fois peut gérer un service donné et accéder aux données y associées,
- Red Hat High Availability peut être configuré dans des configurations matérielles qui ne sont pas **No-single-point-of-failure**,
- Red Hat High Availability est compatible avec les modes **0**, **1** et **2** du **Ethernet Channel Bonding** (l'agrégation de plusieurs interfaces réseaux), à savoir les modes **Round-Robin**, **Active-Passive** et **Balance xor**,
- Red Hat High Availability est compatible IPv4 et IPv6,
- Les composants utilisés dans le cluster tels les périphériques de fencing et switchs Fibre Channel doivent figurés dans la liste de compatibilité : **http://www.redhat.com/cluster_suite/hardware/**,
- Red Hat High Availability est compatible avec SELinux en mode **enforcing** avec une politique **targeted**,
- Les éventuelles machines virtuelles dans un cluster doivent être démarrées et arrêtées en utilisant la commande **rgmanager**. L'utilisation de la commande **virsh** pourrait démarrer la machine virtuelle dans plusieurs endroits créant ainsi une corruption de données. Le service **libvirt-guests** doit être désactivé sur chaque noeud où fonctionne rgmanager,
- Le service **cman** ne démarrera pas dans le cas où le service NetworkManager est lancé voire arrêté tout en étant configuré par la commande **chkconfig**.

Préparation des Machines Virtuelles

A partir de votre machine virtuelle **Redhat**, créez 3 clones complets et configurez-les ainsi :

Nom de la VM	**RAM**
node1	1 024 Mo
node2	1 024 Mo
node3	1 024 Mo

Instructions Particulières

- Lors de la création des clones, veillez à réinitialiser l'adresse MAC de la carte réseau,
- Lors du premier lancement de chaque clone éditez le fichier **/etc/udev/rules.d/70-persistent-net.rules** et supprimez la première entrée ayant

la valeur du champs **NAME="eth0"**. Editez ensuite la ligne restante en modifiant la valeur du champs **NAME="eth1"** en **NAME="eth0"**,
- Re-démarrez chaque machine virtuelle après avoir effectué les modifications,
- Vérifiez la configuration réseau à l'aide de la commande **ifconfig**.

Arrêtez ensuite les machines **node1**, **node2** et **node3**. Modifiez la configuration réseau des trois machines :

Adaptateur	**eth0**	**eth1**	**eth2**
Type de réseau	NAT	intnet	intnet

Démarrez les machines virtuelles **node1**, **node2** et **node3**.

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

Arrêtez et supprimez le service NetworkManager puis activez le service **network** :

```
[root@centos6 ~]# service NetworkManager stop
Arrêt du démon NetworkManager :                            [  OK  ]
[root@centos6 ~]# chkconfig --del NetworkManager
[root@centos6 ~]# chkconfig --list network
network         0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
[root@centos6 ~]# chkconfig --add network
[root@centos6 ~]# chkconfig --list network
network         0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
```

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

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
```

Éditez le fichier **/etc/sysconfig/network-scripts/ifcfg-eth0** :

ifcfg-eth0

```
DEVICE="eth0"
NM_CONTROLLED="no"
ONBOOT=yes
TYPE=Ethernet
BOOTPROTO=static
IPV6INIT=no
NETMASK=255.255.255.0
IPADDR=10.0.2.15
GATEWAY=10.0.2.2
USERCTL=yes
```

Éditez le fichier **/etc/sysconfig/network-scripts/ifcfg-eth1** :

ifcfg-eth1

```
DEVICE=eth1
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
```

Éditez le fichier **/etc/sysconfig/network-scripts/ifcfg-eth2** :

ifcfg-eth2

```
DEVICE=eth2
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
```

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

bonding.conf

```
alias bond0 bonding
```

Modifiez le fichier **/etc/sysconfig/network** :

network

```
NETWORKING=yes
```

```
HOSTNAME=node1.fenestros.loc
```

Modifiez le fichier **/etc/resolv.conf** :

resolv.conf

```
nameserver 8.8.8.8
nameserver 8.8.4.4
```

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

hosts

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.3.15   node1.fenestros.loc     node1
10.0.3.16   node2.fenestros.loc    node2
10.0.3.17   node3.fenestros.loc    node3
```

Saisissez les deux commandes suivantes :

```
[root@centos6 ~]# hostname node1.fenestros.loc
[root@centos6 ~]# service network start
```

Ouverture des Ports

Les ports indiqués dans le tableau suivant doivent être ouverts sur chaque noeud :

Port	Application
5404/udp, 5405/udp	corosync/cman
11111/tcp	ricci
21064/tcp	dlm
16851/tcp	modclusterd

Utilisez les commandes suivantes pour ouvrir les ports pour **cman** dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -m state --state NEW -m multiport -p udp -s 10.0.3.0/24 -d 10.0.3.0/24 --
dports 5404,5405 -j ACCEPT
[root@centos6 ~]# iptables -I INPUT -m addrtype --dst-type MULTICAST -m state --state NEW -m multiport -p udp -s
10.0.3.0/24 --dports 5404,5405 -j ACCEPT
```

Utilisez la commande suivante pour ouvrir les ports pour **ricci** dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -m state --state NEW -p tcp -s 10.0.3.0/24 -d 10.0.3.0/24 --dport 11111 -j
ACCEPT
```

Utilisez la commande suivante pour ouvrir les ports pour **dlm** dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -m state --state NEW -p tcp -s 10.0.3.0/24 -d 10.0.3.0/24 --dport 21064 -j
ACCEPT
```

Utilisez la commande suivante pour ouvrir les ports pour **modclusterd** dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -m state --state NEW -p tcp -s 10.0.3.0/24 -d 10.0.3.0/24 --dport 16851 -j
ACCEPT
```

Utilisez la commande suivante pour ouvrir les ports pour **igmp** (*Internet Group Management Protocol*) dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -p igmp -j ACCEPT
```

Utilisez la commande suivante pour ouvrir les ports pour **luci** dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -m state --state NEW -p tcp -s 10.0.3.0/24 -d 10.0.3.0/24 --dport 8084 -j
ACCEPT
```

Dernièrement, saisissez les commandes suivantes pour terminer la configuration d'iptables :

```
[root@centos6 ~]# service iptables save
iptables : Sauvegarde des règles du pare-feu dans /etc/sysc[  OK  ]tables :
[root@centos6 ~]# service iptables restart
iptables : Suppression des règles du pare-feu :            [  OK  ]
iptables : Configuration des chaînes sur la politique ACCEP[  OK  ]er
iptables : Déchargement des modules :                      [  OK  ]
iptables : Application des règles du pare-feu :            [  OK  ]
```

Vérifiez maintenant votre configuration d'iptables :

```
[root@centos6 ~]# service iptables status
Table : filter
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     tcp  --  10.0.3.0/24         10.0.3.0/24         state NEW tcp dpt:8084
2    ACCEPT     2    --  0.0.0.0/0           0.0.0.0/0
3    ACCEPT     tcp  --  10.0.3.0/24         10.0.3.0/24         state NEW tcp dpt:16851
4    ACCEPT     tcp  --  10.0.3.0/24         10.0.3.0/24         state NEW tcp dpt:21064
5    ACCEPT     tcp  --  10.0.3.0/24         10.0.3.0/24         state NEW tcp dpt:11111
6    ACCEPT     udp  --  10.0.3.0/24         0.0.0.0/0           ADDRTYPE match dst-type MULTICAST state NEW
multiport dports 5404,5405
7    ACCEPT     udp  --  10.0.3.0/24         10.0.3.0/24         state NEW multiport dports 5404,5405
8    ACCEPT     all  --  0.0.0.0/0           0.0.0.0/0           state RELATED,ESTABLISHED
9    ACCEPT     icmp --  0.0.0.0/0           0.0.0.0/0
10   ACCEPT     all  --  0.0.0.0/0           0.0.0.0/0
11   ACCEPT     tcp  --  0.0.0.0/0           0.0.0.0/0           state NEW tcp dpt:22
12   REJECT     all  --  0.0.0.0/0           0.0.0.0/0           reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
```

```
num  target     prot opt source               destination
1    REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
```

Arrêtez votre machine virtuelle. Créez deux clones : node10 et node100. Démarrez la machine node1.

Configuration du node2

Arrêtez et supprimez le service NetworkManager puis activez le service **network** :

```
[root@centos6 ~]# service NetworkManager stop
Arrêt du démon NetworkManager :                            [  OK  ]
[root@centos6 ~]# chkconfig --del NetworkManager
[root@centos6 ~]# chkconfig --list network
network         0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
[root@centos6 ~]# chkconfig --add network
[root@centos6 ~]# chkconfig --list network
network         0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
```

Important - Le service **cman** ne démarrera pas dans le cas où le service NetworkManager est lancé voire arrêté tout en étant configuré par la commande **chkconfig**.

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

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
```

Éditez le fichier **/etc/sysconfig/network-scripts/ifcfg-eth0** :

ifcfg-eth0

```
DEVICE="eth0"
NM_CONTROLLED="no"
ONBOOT=yes
TYPE=Ethernet
BOOTPROTO=static
IPV6INIT=no
NETMASK=255.255.255.0
IPADDR=10.0.2.15
GATEWAY=10.0.2.2
USERCTL=yes
```

Éditez le fichier **/etc/sysconfig/network-scripts/ifcfg-eth1** :

ifcfg-eth1

```
DEVICE=eth1
```

```
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
```

Éditez le fichier **/etc/sysconfig/network-scripts/ifcfg-eth2** :

ifcfg-eth2

```
DEVICE=eth2
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
```

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

bonding.conf

```
alias bond0 bonding
```

Modifiez le fichier **/etc/sysconfig/network** :

network

```
NETWORKING=yes
HOSTNAME=node2.fenestros.loc
```

Modifiez le fichier **/etc/resolv.conf** :

resolv.conf

```
nameserver 8.8.8.8
nameserver 8.8.4.4
```

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

hosts

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.3.15   node1.fenestros.loc     node1
10.0.3.16   node2.fenestros.loc    node2
10.0.3.17   node3.fenestros.loc    node3
```

Saisissez les deux commandes suivantes :

```
[root@centos6 ~]# hostname node2.fenestros.loc
[root@centos6 ~]# service network start
```

Ouverture des Ports

Les ports indiqués dans le tableau suivant doivent être ouverts sur chaque noeud :

Port	Application
5404/udp, 5405/udp	corosync/cman
11111/tcp	ricci
21064/tcp	dlm
16851/tcp	modclusterd

Utilisez les commandes suivantes pour ouvrir les ports pour **cman** dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -m state --state NEW -m multiport -p udp -s 10.0.3.0/24 -d 10.0.3.0/24 --
dports 5404,5405 -j ACCEPT
[root@centos6 ~]# iptables -I INPUT -m addrtype --dst-type MULTICAST -m state --state NEW -m multiport -p udp -s
10.0.3.0/24 --dports 5404,5405 -j ACCEPT
```

Utilisez la commande suivante pour ouvrir les ports pour **ricci** dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -m state --state NEW -p tcp -s 10.0.3.0/24 -d 10.0.3.0/24 --dport 11111 -j
ACCEPT
```

Utilisez la commande suivante pour ouvrir les ports pour **dlm** dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -m state --state NEW -p tcp -s 10.0.3.0/24 -d 10.0.3.0/24 --dport 21064 -j
ACCEPT
```

Utilisez la commande suivante pour ouvrir les ports pour **modclusterd** dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -m state --state NEW -p tcp -s 10.0.3.0/24 -d 10.0.3.0/24 --dport 16851 -j
ACCEPT
```

Utilisez la commande suivante pour ouvrir les ports pour **igmp** (*Internet Group Management Protocol*) dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -p igmp -j ACCEPT
```

Dernièrement, saisissez les commandes suivantes pour terminer la configuration d'iptables :

```
[root@centos6 ~]# service iptables save
iptables : Sauvegarde des règles du pare-feu dans /etc/sysc[  OK  ]tables :
[root@centos6 ~]# service iptables restart
iptables : Suppression des règles du pare-feu :            [  OK  ]
iptables : Configuration des chaînes sur la politique ACCEP[  OK  ]er
```

```
iptables : Déchargement des modules :                      [  OK  ]
iptables : Application des règles du pare-feu :            [  OK  ]
```

Vérifiez maintenant votre configuration d'iptables :

```
[root@centos6 ~]# service iptables status
Table : filter
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     2    --  0.0.0.0/0            0.0.0.0/0
2    ACCEPT     tcp  --  10.0.3.0/24          10.0.3.0/24         state NEW tcp dpt:16851
3    ACCEPT     tcp  --  10.0.3.0/24          10.0.3.0/24         state NEW tcp dpt:21064
4    ACCEPT     tcp  --  10.0.3.0/24          10.0.3.0/24         state NEW tcp dpt:11111
5    ACCEPT     udp  --  10.0.3.0/24          0.0.0.0/0           ADDRTYPE match dst-type MULTICAST state NEW
multiport dports 5404,5405
6    ACCEPT     udp  --  10.0.3.0/24          10.0.3.0/24         state NEW multiport dports 5404,5405
7    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0           state RELATED,ESTABLISHED
8    ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
9    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
10   ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           state NEW tcp dpt:22
11   REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination
1    REJECT     all  --  0.0.0.0/0            0.0.0.0/0           reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
```

Arrêtez votre machine virtuelle. Créez deux clones : node20 et node200. Démarrez la machine node1.

Configuration du node3

Arrêtez et supprimez le service NetworkManager puis activez le service **network** :

```
[root@centos6 ~]# service NetworkManager stop
Arrêt du démon NetworkManager :                            [  OK  ]
[root@centos6 ~]# chkconfig --del NetworkManager
[root@centos6 ~]# chkconfig --list network
network         0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
[root@centos6 ~]# chkconfig --add network
[root@centos6 ~]# chkconfig --list network
network         0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
```

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

ifcfg-bond0

```
DEVICE=bond0
USERCTL=no
BOOTPROTO=none
ONBOOT=yes
IPADDR=10.0.3.17
NETMASK=255.255.255.0
NETWORK=10.0.3.0
BONDING_OPTS="miimon=100 mode=balance-xor"
TYPE=Unknown
IPV6INIT=no
```

Éditez le fichier **/etc/sysconfig/network-scripts/ifcfg-eth0** :

ifcfg-eth0

```
DEVICE="eth0"
NM_CONTROLLED="no"
ONBOOT=yes
TYPE=Ethernet
BOOTPROTO=static
IPV6INIT=no
NETMASK=255.255.255.0
IPADDR=10.0.2.15
GATEWAY=10.0.2.2
USERCTL=yes
```

Éditez le fichier **/etc/sysconfig/network-scripts/ifcfg-eth1** :

ifcfg-eth1

```
DEVICE=eth1
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
USERCTL=no
```

Éditez le fichier **/etc/sysconfig/network-scripts/ifcfg-eth2** :

ifcfg-eth2

```
DEVICE=eth2
BOOTPROTO=none
ONBOOT=yes
MASTER=bond0
SLAVE=yes
```

```
USERCTL=no
```

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

bonding.conf

```
alias bond0 bonding
```

Modifiez le fichier **/etc/sysconfig/network** :

network

```
NETWORKING=yes
HOSTNAME=node3.fenestros.loc
```

Modifiez le fichier **/etc/resolv.conf** :

resolv.conf

```
nameserver 8.8.8.8
nameserver 8.8.4.4
```

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

hosts

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.3.15   node1.fenestros.loc     node1
```

```
10.0.3.16   node2.fenestros.loc    node2
10.0.3.17   node3.fenestros.loc    node3
```

Saisissez les deux commandes suivantes :

```
[root@centos6 ~]# hostname node3.fenestros.loc
[root@centos6 ~]# service network start
```

Ouverture des Ports

Les ports indiqués dans le tableau suivant doivent être ouverts sur chaque noeud :

Port	**Application**
5404/udp, 5405/udp	corosync/cman
11111/tcp	ricci
21064/tcp	dlm
16851/tcp	modclusterd

Utilisez les commandes suivantes pour ouvrir les ports pour **cman** dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -m state --state NEW -m multiport -p udp -s 10.0.3.0/24 -d 10.0.3.0/24 --
dports 5404,5405 -j ACCEPT
[root@centos6 ~]# iptables -I INPUT -m addrtype --dst-type MULTICAST -m state --state NEW -m multiport -p udp -s
10.0.3.0/24 --dports 5404,5405 -j ACCEPT
```

Utilisez la commande suivante pour ouvrir les ports pour **ricci** dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -m state --state NEW -p tcp -s 10.0.3.0/24 -d 10.0.3.0/24 --dport 11111 -j
ACCEPT
```

Utilisez la commande suivante pour ouvrir les ports pour **dlm** dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -m state --state NEW -p tcp -s 10.0.3.0/24 -d 10.0.3.0/24 --dport 21064 -j
ACCEPT
```

Utilisez la commande suivante pour ouvrir les ports pour **modclusterd** dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -m state --state NEW -p tcp -s 10.0.3.0/24 -d 10.0.3.0/24 --dport 16851 -j
ACCEPT
```

Utilisez la commande suivante pour ouvrir les ports pour **igmp** (*Internet Group Management Protocol*) dans **iptables** :

```
[root@centos6 ~]# iptables -I INPUT -p igmp -j ACCEPT
```

Dernièrement, saisissez les commandes suivantes pour terminer la configuration d'iptables :

```
[root@centos6 ~]# service iptables save
iptables : Sauvegarde des règles du pare-feu dans /etc/sysc[  OK  ]tables :
[root@centos6 ~]# service iptables restart
iptables : Suppression des règles du pare-feu :            [  OK  ]
iptables : Configuration des chaînes sur la politique ACCEP[  OK  ]er
iptables : Déchargement des modules :                      [  OK  ]
iptables : Application des règles du pare-feu :            [  OK  ]
```

Vérifiez maintenant votre configuration d'iptables :

```
[root@centos6 ~]# service iptables status
Table : filter
Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
1    ACCEPT     2    --  0.0.0.0/0            0.0.0.0/0
2    ACCEPT     tcp  --  10.0.3.0/24          10.0.3.0/24         state NEW tcp dpt:16851
3    ACCEPT     tcp  --  10.0.3.0/24          10.0.3.0/24         state NEW tcp dpt:21064
4    ACCEPT     tcp  --  10.0.3.0/24          10.0.3.0/24         state NEW tcp dpt:11111
5    ACCEPT     udp  --  10.0.3.0/24          0.0.0.0/0           ADDRTYPE match dst-type MULTICAST state NEW
```

```
multiport dports 5404,5405
6    ACCEPT     udp  --  10.0.3.0/24          10.0.3.0/24          state NEW multiport dports 5404,5405
7    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0            state RELATED,ESTABLISHED
8    ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0
9    ACCEPT     all  --  0.0.0.0/0            0.0.0.0/0
10   ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0            state NEW tcp dpt:22
11   REJECT     all  --  0.0.0.0/0            0.0.0.0/0            reject-with icmp-host-prohibited

Chain FORWARD (policy ACCEPT)
num  target     prot opt source               destination
1    REJECT     all  --  0.0.0.0/0            0.0.0.0/0            reject-with icmp-host-prohibited

Chain OUTPUT (policy ACCEPT)
num  target     prot opt source               destination
```

Arrêtez votre machine virtuelle. Créez deux clones : node30 et node300. Démarrez la machine node1.

Tester les Serveurs

Lancez **node1**, **node2** et **node3**. 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=48.8 ms

--- www.free.fr ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 91ms
rtt min/avg/max/mdev = 48.852/48.852/48.852/0.000 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=0.884 ms

--- 10.0.3.16 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 1ms
rtt min/avg/max/mdev = 0.884/0.884/0.884/0.000 ms
[root@node1 ~]# ping -c1 10.0.3.17
PING 10.0.3.17 (10.0.3.17) 56(84) bytes of data.
64 bytes from 10.0.3.17: icmp_seq=1 ttl=64 time=0.675 ms

--- 10.0.3.17 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.675/0.675/0.675/0.000 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=36.6 ms

--- www.free.fr ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 78ms
rtt min/avg/max/mdev = 36.650/36.650/36.650/0.000 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=0.359 ms

--- 10.0.3.15 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.359/0.359/0.359/0.000 ms
[root@node2 ~]# ping -c1 10.0.3.17
PING 10.0.3.17 (10.0.3.17) 56(84) bytes of data.
64 bytes from 10.0.3.17: icmp_seq=1 ttl=64 time=1.60 ms

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

```
rtt min/avg/max/mdev = 1.605/1.605/1.605/0.000 ms
```

```
[root@node3 ~]# 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=36.4 ms

--- www.free.fr ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 78ms
rtt min/avg/max/mdev = 36.436/36.436/36.436/0.000 ms
[root@node3 ~]# 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=0.649 ms

--- 10.0.3.16 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.649/0.649/0.649/0.000 ms
[root@node3 ~]# 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=0.587 ms

--- 10.0.3.15 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.587/0.587/0.587/0.000 ms
```

Démarrer le Service ricci si nécessaire

Dernièrement démarrez le service ricci sur chaque noeud :

```
[root@node1 ~]# service ricci status
ricci est arrêté
[root@node1 ~]# service ricci start
Démarrage de oddjobd :                                     [  OK  ]
generating SSL certificates...  done
```

```
Generating NSS database...  done
Démarrage de ricci :                                       [  OK  ]
```

```
[root@node2 ~]# service ricci status
ricci est arrêté
[root@node2 ~]# service ricci start
Démarrage de oddjobd :                                     [  OK  ]
generating SSL certificates...  done
Generating NSS database...  done
Démarrage de ricci :                                       [  OK  ]
```

```
[root@node3 ~]# service ricci status
ricci est arrêté
[root@node3 ~]# service ricci start
Démarrage de oddjobd :                                     [  OK  ]
generating SSL certificates...  done
Generating NSS database...  done
Démarrage de ricci :                                       [  OK  ]
```

Configurer un Cluster avec Conga

Introduction

La configuration en utilisant Conga (l'interface graphique de luci) consiste en :

- L'installation et le démarrage de luci sur un noeud,
- La création d'utilisateurs et des permissions,
- La création d'un cluster,
- La configuration des propriétés générales du cluster,
- La configuration du daemon fenced,
- La configuration de réseau,
- La configuration des périphériques fence (*Fence Devices*),

- La configuration des domaines de basculement (*Failover Domains*),
- La création de ressources globales,
- La création des services en cluster.

LAB #1 - L'Installation et le Démarrage de luci sur un Noeud

Installer luci sur node1.fenestros.loc

Téléchargez les paquets **python-beaker-1.3.1-7.el6.noarch.rpm**, **python-weberror-0.10.2-2.el6.noarch.rpm** et **luci-0.26.0-93.el6.centos.x86_64.rpm** :

```
[root@node1 ~]# wget http://ftp.riken.jp/Linux/centos/6/os/x86_64/Packages/python-beaker-1.3.1-7.el6.noarch.rpm
[root@node1 ~]# wget
http://ftp.riken.jp/Linux/centos/6/os/x86_64/Packages/python-weberror-0.10.2-2.el6.noarch.rpm
[root@node1 ~]# wget http://ftp.riken.jp/Linux/centos/6/os/x86_64/Packages/luci-0.26.0-93.el6.centos.x86_64.rpm
```

Installez les paquets :

```
[root@node1 ~]# yum -y localinstall python-beaker-1.3.1-7.el6.noarch.rpm --nogpgcheck
[root@node1 ~]# yum -y localinstall python-weberror-0.10.2-2.el6.noarch.rpm --nogpgcheck
[root@node1 ~]# yum -y localinstall luci-0.26.0-93.el6.centos.x86_64.rpm --exclude=python-beaker --
exclude=python-weberror --disablerepo=epel* --nogpgcheck
```

Démarrez le service luci :

```
[root@node1 ~]# service luci start
Adding following auto-detected host IDs (IP addresses/domain names), corresponding to `controller' address, to
the configuration of self-managed certificate `/var/lib/luci/etc/cacert.config' (you can change them by editing
`/var/lib/luci/etc/cacert.config', removing the generated certificate `/var/lib/luci/certs/host.pem' and
restarting luci):
    (none suitable found, you can still do it manually as mentioned above)
```

```
Generating a 2048 bit RSA private key
writing new private key to '/var/lib/luci/certs/host.pem'
Démarrage de saslauthd :                                   [  OK  ]
Start luci...                                              [  OK  ]
Point your web browser to https://controller:8084 (or equivalent) to access luci
```

Si luci ne démarre pas, créez les fichiers **/var/lib/luci/etc/luci.ini** et **/var/lib/luci/data/luci.db** :

```
[root@node1 ~]# touch /var/lib/luci/etc/luci.ini
[root@node1 ~]# touch /var/lib/luci/data/luci.db
```

Modifiez ensuite le propriétaire et le groupe de chaque fichier :

```
[root@node1 ~]# chown luci:luci /var/lib/luci/etc/luci.ini
[root@node1 ~]# chown luci:luci /var/lib/luci/data/luci.db
```

Démarrez le service luci :

```
[root@node1 ~]# service luci start
Adding following auto-detected host IDs (IP addresses/domain names), corresponding to `controller' address, to
the configuration of self-managed certificate `/var/lib/luci/etc/cacert.config' (you can change them by editing
`/var/lib/luci/etc/cacert.config', removing the generated certificate `/var/lib/luci/certs/host.pem' and
restarting luci):
    (none suitable found, you can still do it manually as mentioned above)

Generating a 2048 bit RSA private key
writing new private key to '/var/lib/luci/certs/host.pem'
Démarrage de saslauthd :                                   [  OK  ]
Start luci...                                              [  OK  ]
Point your web browser to https://controller:8084 (or equivalent) to access luci
```

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

```
# User serviceable configuration of luci
```

```
#
# This configuration file uses a rather uncommon structure because it is
# intended to be parseable by both shell and Python's ConfigParser.  If you
# want to change any configuration item listed (i.e. to explicitly redefine
# luci's defaults for this item), uncomment particular line and set your
# value.  Each configurable item is preceeded with descriptive comment.
#
# Please be aware that changing anything else may render this file defective
# when unexpected content found here!
#
# Remember to restart the luci service for the changes to take effect.


# ============================================================================
# INITSCRIPT CONFIGURATION
# Shell syntax (i.e. FOO=bar, BAZ="white-spaced string")
# ============================================================================
[ INITSCRIPT ]

# Change this to set log file; this file either must not exist (so it is
# initially created in ownership of user running initscript, presumably root,
# and then luci becomes its owner) or must be writeable for luci
# directly
#LOG_FILE="path_to/log_file"

# Uncomment this to override default behaviour of removing run time data
# (files maintained by middleware handling cache and sessions) everytime
# the luci service is stopped (and also on service start)
#KEEP_RUNTIME_DATA=1


INIT_CONFIG=`cat -E <<"#END#"

#
```

```
# From this point, everything is in Python's ConfigParser syntax
# (i.e. no quoted strings and the syntax is generally more relaxed)
#

# ================================================================================
# SERVER CONFIGURATION
# ================================================================================
[server:main]

# Change this to set the host IP (and, in turn, respective network interface)
# running luci binds at;  this is particularly useful when it should be bound
# only at a specific one according to its IP address (0.0.0.0 => any interface)
#host = 127.0.0.1

# Change this to set the port running luci binds at;  please note that you
# cannot use privileged ports (i.e. <1024) because it runs as a non-root user
#port = 4443

# Change this to force luci to use custom SSL certificate (given the path of
# PEM file containing both the certificate itself and respective private key),
# otherwise its self-signed certificate managed automatically by luci
# service is used instead;
# currently, this certificate is used only for https connection with web
# browser, connections with ricci instances still rely on self-managed cert
# Note: If this configuration item is active and no such file can be read,
#       starting luci service will fail
#ssl_pem = "path_to/ssl_cert_pem_file"

use=config:%(base_config)s

# ================================================================================
# APPLICATION CONFIGURATION
# ================================================================================
[app:main]
```

```
# Change this to override default number of seconds of inactivity after which
# the luci authenticated session will timeout
# (requires repoze.who >= 1.0.14)
#who.auth_tkt_timeout = 600

use=config:%(base_config)s

#END#`
```

Dernièrement configurez luci pour un démarrage automatique :

```
[root@node1 ~]# chkconfig --list luci
luci            0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
[root@node1 ~]# chkconfig --levels 2345 luci on
[root@node1 ~]# chkconfig --list luci
luci            0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
```

Lancez une nouvelle fenêtre de votre navigateur et saisissez l'adresse **http://localhost:8084**. Vous obtiendrez une fenêtre similaire à celle-ci :

LAB #2 - La Création d'Utilisateurs et des Permissions

Connectez-vous à Conga en utilisant le compte root de la machine :

Vous obtiendrez une fenêtre d'avertissement qui indique que cet outil ne doit être utilisé que par des personnes ayant les connaissances appropriées :

Cliquez sur le bouton **OK** de la fenêtre d'avertissement. Vous apercevrez la fenêtre principale de l'outil Conga :

Important - Notez qu'en cas d'inactivité de plus de 15 minutes, la session est fermée.

Cliquez sur le lien **Admin** dans le coins supérieur à droite. Vous obtiendrez l'interface pour gérer les utilisateurs et les permissions :

Cliquez sur le bouton **Add a user**. Vous obtiendrez la fenêtre de création d'un utilisateur :

Saisissez le nom **trainee** et cliquez sur le bouton **Submit** :

Vous apercevrez que l'utilisateur trainee a bien été créé :

Dans la partie inférieure de la fenêtre, vous aurez la possibilité d'attribuer des permissions à l'utilisateur trainee :

Dans la liste des permissions vous verrez :

- **Luci Administrator**,
  - Ce type d'utilisateur possède les pleins pouvoirs sur tous les clusters ainsi que les droits de modification des permissions de tout utilisateur à l'exception de root,
- **Can Create Clusters**,
  - Ce type d'utilisateur peut créer des clusters,

- **Can Import Existing Clusters**,
  - Ce type d'utilisateur peut ajouter un cluster existant à luci.

Cochez l'option **Luci Administrator** et cliquez sur le bouton **Submit** :

Vous obtiendrez la fenêtre suivante :

Notez qu'il n'est pas possible de modifier le mot de passe d'un utilisateur à partir de l'interface luci. Pour cette raison il est possible de créer un utilisateur, par exemple **toto**, sans pour autant que cet utilisateur puisse se connecter à luci. Pour comprendre pourquoi, sachez que luci se base sur PAM pour autoriser des connexions. Le fichier de configuration est **/etc/pam.d/luci** :

/etc/pam.d/luci

```
#%PAM-1.0
auth       include  password-auth
account    include  password-auth
password   include  password-auth
session    include  password-auth
```

Notez que ce fichier ne contient que des **include** vers le fichier **/etc/pam.d/password-auth** :

/etc/pam.d/password-auth

```
#%PAM-1.0
# This file is auto-generated.
# User changes will be destroyed the next time authconfig is run.
auth        required      pam_env.so
auth        sufficient    pam_unix.so nullok try_first_pass
```

```
auth        requisite     pam_succeed_if.so uid >= 500 quiet
auth        required      pam_deny.so

account     required      pam_unix.so
account     sufficient    pam_localuser.so
account     sufficient    pam_succeed_if.so uid < 500 quiet
account     required      pam_permit.so

password    requisite     pam_cracklib.so try_first_pass retry=3 type=
password    sufficient    pam_unix.so sha512 shadow nullok try_first_pass use_authtok
password    required      pam_deny.so

session     optional      pam_keyinit.so revoke
session     required      pam_limits.so
session     [success=1 default=ignore] pam_succeed_if.so service in crond quiet use_uid
session     required      pam_unix.so
```

Ce système indique donc que les utilisateurs de luci doivent être des utilisateurs **système**. La modification du mot de passe se fait donc avec les utilitaires système habituels.

Important - Les utilisateurs luci sont les utilisateurs système à qui vous confiez les autorisations spécifiques à luci.

LAB #3 - La Création d'un Cluster

Cliquez sur le bouton **Homebase** à gauche. Vous obtiendrez la fenêtre suivante :

Cliquez sur le bouton **Manage Clusters**. Vous obtiendrez la fenêtre suivante :

Cliquez sur le lien **Create**. Vous obtiendrez la fenêtre suivante :

Remplissez le formulaire ainsi :

Important - Vous avez appelé votre cluster **moncluster**. Le nom d'un cluster est limité à **15** caractères. Puisque le mot de passe de l'utilisateur ricci est identique dans chaque noeud, vous avez coché **Use the Same Password for All Nodes**. Pour ajouter successivement chaque noeud, vous avez cliqué sur le bouton **Add Another Node**. Afin que chaque noeud contienne les paquets nécessaires, vous avez cochez **Download Packages**.

Cliquez maintenant sur le bouton **Create Cluster**. Vous obtiendrez la fenêtre suivante :

Important - Regardez le processus de création du cluster.

A l'issu de la création du cluster, vous obtiendrez la fenêtre suivante :

Important - A partir de cette interface vous pouvez ajouter ou supprimer un noeud du cluster grâce aux boutons **Add** et **Delete**. Veuillez noter que la suppression d'un noeud est une action destructive qui ne peut pas être reversée.

La Configuration des Propriétés Générales du Cluster

Cliquez sur le bouton **Configure**. Vous obtiendrez la fenêtre suivante :

Le premier onglet concerne les propriétés générales du cluster. Vous ne pouvez pas modifier le nom du cluster. La valeur du champs **Configuration Version** est incrémentée automatiquement après chaque modification du cluster. Vous pouvez cependant modifier manuellement la valeur.

La Configuration du Daemon Fenced

Cliquez maintenant sur l'onglet **Fence Daemon**. Vous obtiendrez la fenêtre suivante :

La configuration par défaut du daemon **fenced** est définie ici. Des configurations spécifiques par périphérique peuvent être spécifiées en cliquant sur le bouton **Fence Devices**. Sur l'onglet, on constate la présence de deux paramètres :

- **Post Fail Delay** - Le nombre de secondes qu'attend le daemon fenced avant de réagir quand un noeud dans un **Domaine Fence** devient défaillant. La valeur par défaut est **0**,
- **Post Join Delay** - Le nombre de secondes qu'attend le daemon fenced avant de réagir quand un noeud rejoint un **Domaine Fence**. La valeur par défaut est **6**. La valeur conseillée est entre **20** et **30**.

La Configuration du Réseau

Cliquez maintenant sur l'onglet **Network**. Vous obtiendrez la fenêtre suivante :

> **Important** - Les noeuds dans un cluster communiquent entre eux en utilisant des adresses multicast (multidiffusion). De ce fait, tous les périphériques doivent être configurés pour permettre l'utilisation des adresses multicast **et** être compatibles avec le protocole **IGMP** (*Internet Group Management Protocol*). Depuis la version 6.2 de RHEL, la communication entre les noeuds peut se faire en utilisant de l'UDP Unicast. Par contre il est déconseillé d'utiliser l'UDP Unicast avec du GFS2.

Le paramètre **Network Transport Type** prend une de trois valeurs :

- **UDP Multicast and Let Cluster Choose the Multicast Address**,
  - IPv4 - l'adresse commence par **239.192.**. Les 16 derniers bits sont générés par le logiciel **Red Hat High Availability** en fonction de l'ID du cluster,
  - IPv6 - l'adresse commence par **FF15::**. Les 16 derniers bits sont générés par le logiciel **Red Hat High Availability** en fonction de l'ID du cluster,
- **UDP Multicast and Specify the Multicast Address Manually**,
  - Permet de spécifier une adresse multicast spécifique,
- **UDP Unicast (UDPU)**,
  - Permet de spécifier une adresse UDP unicast spécifique.

L'ID du cluster peut être visualisé en utilisant la commande **cman_tool** avec l'option **status** :

```
[root@node1 ~]# cman_tool status
Version: 6.2.0
Config Version: 1
Cluster Name: moncluster
Cluster Id: 59006
Cluster Member: Yes
Cluster Generation: 12
Membership state: Cluster-Member
Nodes: 3
```

```
Expected votes: 3
Total votes: 3
Node votes: 1
Quorum: 2
Active subsystems: 9
Flags:
Ports Bound: 0 11 177
Node name: node1.fenestros.loc
Node ID: 1
Multicast addresses: 239.192.230.101
Node addresses: 10.0.3.15
```

Important - Sauf cas exceptionnel, il est recommandé d'utiliser la valeur par défaut **UDP Multicast and Let Cluster Choose the Multicast Address**.

LAB #4 - La Configuration des Périphériques Fence

Cliquez maintenant sur le bouton **Fence Devices**. Vous obtiendrez la fenêtre suivante :

Cliquez sur le bouton **Add**. Vous obtiendrez la fenêtre suivante :

Dans la liste déroulante **–Select a Fence Device–** de la boîte de dialogue **Add Fence Device (Instance)**, choisissez **APC Power Switch**. Vous obtiendrez la fenêtre suivante :

Remplissez le formulaire en indiquant un nom, une adresse IP et les coordonnées de connexion :

Important - La documentation sur la configuration de chaque Périphérique Fence peut être consultée à **cette adresse**.

Configurez deux autres Périphériques Fence **APC Power Switch** ayant respectivement les adresses IP 10.0.3.51 et 10.0.3.52. Vous obtiendrez la fenêtre suivante :

Pour supprimer le Périphérique Fence **APC3**, cochez la case à gauche du nom et cliquez sur le bouton **Delete** :

Important - Notez que vous ne pouvez pas supprimer un Périphérique Fence si celui-ci est utilisé par un noeud.

Vous obtiendrez la fenêtre suivante :

Cliquez sur le bouton **Configure** puis sur l'onglet **General**. Vous noterez que le numéro de la **Configuration Version** a été incrémenté :

Configurer un Périphérique Fence pour un Noeud

Cliquez sur le bouton **Nodes**. Vous obtiendrez la fenêtre suivante :

Cliquez ensuite sur le lien **node2.fenestros.loc** dans la colonne **Node Name**. Vous obtiendrez la fenêtre suivante :

Descendez en bas de la fenêtre et cliquez sur le bouton **Add Fence Method** :

Vous verrez la boîte de dialogue **Add Fence Method to Node** :

Dans le champs **Method Name**, saisissez **APC1** et cliquez sur le bouton **Submit** :

Retournez à la section **Fence Devices** du **node2.fenestros.loc** et cliquez sur le bouton **Add Fence Instance** :

Dans la liste déroulante **–Select a Fence Device–** de la boîte de dialogue **Add Fence Device (Instance)**, choisissez **APC1 (APC Power Device)** :

Vous obtiendrez la fenêtre suivante :

Saisissez le numéro du port et cliquez sur le bouton **Submit** :

Vous obtiendrez la fenêtre suivante :

A l'aide de toute source, expliquez pourquoi la valeur du port est **1**.

Configurer un Périphérique Fence de Secours pour un Noeud

Afin de palier au danger d'une panne du premier Périphérique Fence, il est possible de configurer un Périphérique Fence de Secours. Descendez en bas de la fenêtre et cliquez de nouveau sur le bouton **Add Fence Method**. Saisissez un **Method Name** tel **Secours** et cliquez sur le bouton **Submit** :

Suivez la procédure expliquée ci-dessus pour ajouter le Périphérique Fence de Secours. A l'issu de la configuration vous obtiendrez la fenêtre suivante :

Consultez le fichier **/etc/cluster/cluster.conf** :

cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="15" name="moncluster">
```

```
    <clusternodes>
        <clusternode name="node1.fenestros.loc" nodeid="1"/>
        <clusternode name="node2.fenestros.loc" nodeid="2">
            <fence>
                <method name="APC1">
                    <device name="APC1" port="1"/>
                </method>
                <method name="Secours">
                    <device name="APC2" port="1"/>
                </method>
            </fence>
        </clusternode>
        <clusternode name="node3.fenestros.loc" nodeid="3"/>
    </clusternodes>
    <fencedevices>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.50" login="root" name="APC1" passwd="password"
power_wait="5"/>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.51" login="root" name="APC2" passwd="password"
power_wait="5"/>
    </fencedevices>
</cluster>
```

Important - Notez la présence de deux lignes dans la section <fence> de <clusternode name=“node2.fenestros.loc” nodeid=“2”> ainsi que les deux lignes dans la section <fencedevices>.

Configurer un Noeud avec une Alimentation Redondante

Dans le cas d'une alimentation redondante, il est important de comprendre que si les deux Périphériques Fence sont configurés dans deux **Fence**

Method différents, le noeud ne sera pas *Fenced*. Si le premier Fence Method est utilisé et l'alimention coupée, la deuxième alimentation prendra le relais !

Dans ce cas précis il convient de mettre les deux Périphériques Fence dans **la même Fence Method** en tant que deux instances distincts.

LAB #5 - La Configuration des Domaines de Basculement

Un **Domaine de Basculement** est un sous-ensemble de noeuds d'un cluster où les **membres** peuvent faire fonctionner un **service à haute disponibilité** en cas de défaillance du membre sur lequel le service a été originalement démarré. Un domaines de basculement possède plusieurs caractéristiques :

Caractéristique	**Description**
Unrestricted	Permet de désigner un sous-ensemble de membres préférés, mais qu'un service rattaché au domaine peut être démarré sur n'importe quel membre disponible.
Restricted	Permet de désigner un sous-ensemble de membres obligatoires. Si aucun membre désigné n'est disponible, le service ne peut pas être démarré.
Unordered	Un service peut être démarré sur n'importe quel membre du domaine disponible sans aucun ordre pré-défini.
Ordered	Un service peut être démarré sur n'importe quel membre du domaine dans un ordre pré-défini par des priorités.
Failback	Permet de basculer un service vers le membre sur lequel le service a été originalement démarré. Nécessite le caractéristique **Ordered**

Important - Par défaut un domaine de basculement possède les caractéristiques **Unrestricted** et **Unordered**.

Pour créer un domaine de basculement, cliquez sur le bouton **Failover Domains** puis sur le bouton **Add** :

Vous obtiendrez la fenêtre suivante :

Créez un domaine de basculement, appelé **Failover1**, contenant **node1.fenestros.loc** et **node2.fenestros.loc**, sans priorités, en cochant les cases respectives et en cliquant sur le bouton **Create** :

Vous obtiendrez la fenêtre suivante :

Pour modifier le domaine de basculement vous disposez de deux boutons, **Update Properties** et **Update Settings**. Cochez maintenant l'option **No Failback** et cliquez sur le bouton **Update Properties** :

Cliquez maintenant sur le bouton **Failover Domains**. A partir de cette interface vous pouvez ajouter un autre domaine de basculement ou supprimer un domaine existant. Cochez donc la case à gauche du nom **Failover1** et cliquez sur le bouton **Delete** pour le supprimer :

Re-créez le domaine de basculement **Failover1** en suivant le processus détaillé ci-dessus.

LAB #6 - La Création de Ressources Globales

Pour créer une ressource globale, cliquez sur le bouton **Resources** puis sur le bouton **Add** :

Dans la liste déroulante **–Select a Resource Type–** de la boîte de dialogue **Add Resource to Cluster**, choisissez **IP Address** :

Important - La documentation sur la configuration de chaque ressource peut être consultée à **cette adresse**.

Saisissez l'adresse IP **10.0.3.100** et **24** pour le nombre de bits dans le masque de sous-réseau puis cliquez sur le bouton **Submit** :

Vous obtiendrez la fenêtre suivante :

LAB #7 - La Création des Services en Cluster

Pour créer un service en cluster, cliquez sur le bouton **Service Groups** puis sur le bouton **Add** :

Vous verrez apparaître la fenêtre **Add Service Group to Cluster** :

Si cochée, l'option **Run Exclusive** met en place une politique qui empêche le service en cours de configuration de démarrer sur un noeud où d'autres services sont actifs.

Le menu déroulant **Recovery Mode** contient quatre choix :

Choix	Description
Relocate	En cas d'échec, le système essayer de démarrer le service sur un autre noeud.
Restart	En cas d'échec, le système essayera de re-démarrer le service sur le même noeud avant de démarrer le service sur un autre noeud.

Choix	Description
Restart-Disable	En cas d'échec, le service est redémarré mais en cas d'échec du redémarrage le service est désactivé au lieu d'être transférer vers un autre noeud.
Disable	En cas d'échec d'un composant d'un groupe de ressources, le système désactive le groupe de ressources.

Saisissez un **Service Name**, cochez **Automatically Start This Service**, choisissez le domaine de basculement **Failover1** dans la liste déroulante **Failover Domain** et cliquez sur le bouton **Add Resource** :

Vous verrez apparaître la boîte de dialogue **Add Resource to Service** :

Dans la liste déroulante **–Select a Resource Type–** de la boîte de dialogue **Add Resource to Service**, choisissez **10.0.3.100/24** :

Vous obtiendrez la fenêtre suivante. Cliquez sur le bouton **Submit** :

Fournissez la définition des options **Independant Subtree** et Non-Critical Resource.

Vous noterez que le statut du service est inconnu (*Unknown*). Cochez la case à gauche du service et cliquez sur le bouton **Start** :

Vous obtiendrez une fenêtre similaire à celle-ci :

En fonction du noeud sur lequel le service a été démarré, il convient de vérifier sa mise en place :

```
[root@node1 ~]# /sbin/ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
    link/ether 08:00:27:9b:b3:ce brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global eth0
    inet6 fe80::a00:27ff:fe9b:b3ce/64 scope link
       valid_lft forever preferred_lft forever
3: eth1: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master bond0 state UP qlen 1000
    link/ether 08:00:27:5d:b9:44 brd ff:ff:ff:ff:ff:ff
4: eth2: <BROADCAST,MULTICAST,SLAVE,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master bond0 state UP qlen 1000
    link/ether 08:00:27:5d:b9:44 brd ff:ff:ff:ff:ff:ff
5: bond0: <BROADCAST,MULTICAST,MASTER,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
    link/ether 08:00:27:5d:b9:44 brd ff:ff:ff:ff:ff:ff
    inet 10.0.3.15/24 brd 10.0.3.255 scope global bond0
    inet 10.0.3.100/24 scope global secondary bond0
    inet6 fe80::a00:27ff:fe5d:b944/64 scope link tentative dadfailed
       valid_lft forever preferred_lft forever
```

Important - Notez la présence de la ligne **inet 10.0.3.100/24 scope global secondary bond0**

Testez maintenant à partir des deux autres noeuds :

```
[root@node2 ~]# ping -c3 10.0.3.100
```

```
PING 10.0.3.100 (10.0.3.100) 56(84) bytes of data.
64 bytes from 10.0.3.100: icmp_seq=1 ttl=64 time=0.645 ms
64 bytes from 10.0.3.100: icmp_seq=2 ttl=64 time=0.569 ms
64 bytes from 10.0.3.100: icmp_seq=3 ttl=64 time=0.634 ms

--- 10.0.3.100 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.569/0.616/0.645/0.033 ms
```

```
[root@node3 ~]# ping -c3 10.0.3.100
PING 10.0.3.100 (10.0.3.100) 56(84) bytes of data.
64 bytes from 10.0.3.100: icmp_seq=1 ttl=64 time=1.84 ms
64 bytes from 10.0.3.100: icmp_seq=2 ttl=64 time=0.480 ms
64 bytes from 10.0.3.100: icmp_seq=3 ttl=64 time=0.559 ms

--- 10.0.3.100 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 0.480/0.962/1.847/0.626 ms
```

LAB #8 - Redémarrer un Noeud

Cliquez sur le bouton **Nodes** et cochez la case à gauche de **node1.fenestros.loc**. Cliquez ensuite sur le bouton **Reboot** :

Lors du redémarrage du noeud, vérifiez que vous pouvez accéder à l'interface **High Availability Management**. Sinon, vérifiez que tous les services nécessaires fonctionnent.

Gérer un Cluster avec Conga

LAB #9 - Sauvegarder et Restaurer la Configuration de luci

Sauvegarder la Configuration de luci

La configuration de luci est stocké dans une base de données qui se trouve dans le répertoire **/var/lib/luci/data** :

```
[root@node1 ~]# ls /var/lib/luci/data
luci.db
```

Il est à noter que :

- cette base de données ne contient **pas** la configuration du cluster qui est stocké dans **/etc/cluster/cluster.conf**,
- pour restaurer la base de données sur un **autre** noeud, il faut aussi sauvegarder et restaurer le certificat SSL, **/var/lib/luci/certs/host.pem** et le fichier **/var/lib/luci/etc/luci.ini**.

Pour sauvegarder la base de données de luci, arrêtez le service :

```
[root@node1 ~]# service luci stop
Stop luci...                                               [  OK  ]
```

Exécutez ensuite la commande **service luci backup-db** :

```
[root@node1 ~]# service luci backup-db
```

Par défaut les sauvegardes sont stockées dans le même répertoire que la base de données de luci :

```
[root@node1 ~]# ls /var/lib/luci/data
luci-backup20131012103330.db  luci.db
```

En utilisant la commande **service luci list-backups**, vous pouvez visualiser les sauvegardes dans **/var/lib/luci/data** :

```
[root@node1 ~]# service luci list-backups
/var/lib/luci/data/luci-backup20131012103330.db
```

IL est aussi possible de sauvegarder la base de données de luci ailleurs :

```
[root@node1 ~]# service luci backup-db /root/luci.db.backup
[root@node1 ~]# ls /root | grep luci
luci.db.backup
```

Par contre cette sauvegarde n'est pas visible à la commande **service luci list-backups** :

```
[root@node1 ~]# service luci list-backups
/var/lib/luci/data/luci-backup20131012103330.db
```

Ayant sauvegardé la base de données luci, démarrez le service :

```
[root@node1 ~]# service luci start
Start luci...                                              [  OK  ]
Point your web browser to https://node1.fenestros.loc:8084  (or equivalent) to access luci
```

Restaurer la Configuration de luci sur node2.fenestros.loc

Installez luci sur **node2.fenestros.loc**. Téléchargez les paquets **python-beaker-1.3.1-7.el6.noarch.rpm**, **python-weberror-0.10.2-2.el6.noarch.rpm** et **luci-0.26.0-93.el6.centos.x86_64.rpm** :

```
[root@node2 ~]# wget http://ftp.riken.jp/Linux/centos/6/os/x86_64/Packages/python-beaker-1.3.1-7.el6.noarch.rpm
[root@node2 ~]# wget
http://ftp.riken.jp/Linux/centos/6/os/x86_64/Packages/python-weberror-0.10.2-2.el6.noarch.rpm
[root@node2 ~]# wget http://ftp.riken.jp/Linux/centos/6/os/x86_64/Packages/luci-0.26.0-93.el6.centos.x86_64.rpm
```

Installez les paquets :

```
[root@node2 ~]# yum -y localinstall python-beaker-1.3.1-7.el6.noarch.rpm --nogpgcheck
[root@node2 ~]# yum -y localinstall python-weberror-0.10.2-2.el6.noarch.rpm --nogpgcheck
[root@node2 ~]# yum -y localinstall luci-0.26.0-93.el6.centos.x86_64.rpm --exclude=python-beaker --
exclude=python-weberror --disablerepo=epel* --nogpgcheck
```

Démarrez le service luci :

```
[root@node2 ~]# service luci start
Adding following auto-detected host IDs (IP addresses/domain names), corresponding to `controller' address, to
the configuration of self-managed certificate `/var/lib/luci/etc/cacert.config' (you can change them by editing
`/var/lib/luci/etc/cacert.config', removing the generated certificate `/var/lib/luci/certs/host.pem' and
restarting luci):
    (none suitable found, you can still do it manually as mentioned above)

Generating a 2048 bit RSA private key
writing new private key to '/var/lib/luci/certs/host.pem'
Démarrage de saslauthd :                                   [  OK  ]
Start luci...                                              [  OK  ]
Point your web browser to https://controller:8084 (or equivalent) to access luci
```

Si luci ne démarre pas, créez les fichiers **/var/lib/luci/etc/luci.ini** et **/var/lib/luci/data/luci.db** :

```
[root@node2 ~]# touch /var/lib/luci/etc/luci.ini
[root@node2 ~]# touch /var/lib/luci/data/luci.db
```

Modifiez ensuite le propriétaire et le groupe de chaque fichier :

```
[root@node2 ~]# chown luci:luci /var/lib/luci/etc/luci.ini
[root@node2 ~]# chown luci:luci /var/lib/luci/data/luci.db
```

Démarrez le service luci :

```
[root@node2 ~]# service luci start
Adding following auto-detected host IDs (IP addresses/domain names), corresponding to `controller' address, to
the configuration of self-managed certificate `/var/lib/luci/etc/cacert.config' (you can change them by editing
`/var/lib/luci/etc/cacert.config', removing the generated certificate `/var/lib/luci/certs/host.pem' and
restarting luci):
    (none suitable found, you can still do it manually as mentioned above)

Generating a 2048 bit RSA private key
writing new private key to '/var/lib/luci/certs/host.pem'
Démarrage de saslauthd :                                   [  OK  ]
Start luci...                                              [  OK  ]
Point your web browser to https://controller:8084 (or equivalent) to access luci
```

A partir de **node1.fenestros.loc**, transférez le **certificat SSL**, le fichier **/var/lib/luci/etc/luci.ini** ainsi que la **sauvegarde** vers **node2.fenestros.loc** :

```
[root@node1 ~]# scp /var/lib/luci/certs/host.pem /var/lib/luci/etc/luci.ini $(service luci list-backups)
root@node2:
The authenticity of host 'node2 (10.0.3.16)' can't be established.
RSA key fingerprint is 63:cf:95:55:ac:8e:d1:f8:aa:c4:44:46:da:aa:b7:2f.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'node2,10.0.3.16' (RSA) to the list of known hosts.
root@node2's password:
host.pem
100% 2868     2.8KB/s   00:00
luci.ini
100% 8554     8.4KB/s   00:00
luci-backup20131012103330.db
100% 4854     4.7KB/s   00:00
```

Restaurer le certificat et la sauvegarde :

```
[root@node2 ~]# ls
anaconda-ks.cfg  Desktop  host.pem  install.log  install.log.syslog  luci-backup20131012103330.db
```

```
[root@node2 ~]# cp host.pem /var/lib/luci/certs/
[root@node2 ~]# chown luci: /var/lib/luci/certs/host.pem
[root@node2 ~]# cp luci.ini /var/lib/luci/etc/
[root@node2 ~]# chown luci:luci /var/lib/luci/etc/luci.ini
[root@node2 ~]# service luci restore-db ~/luci-backup20131012103330.db
[root@node2 ~]# service luci start
Start luci...                                              [  OK  ]
Point your web browser to https://node2.fenestros.loc:8084 (or equivalent) to access luci
```

Lancez une nouvelle fenêtre de votre navigateur et saisissez l'adresse **http://localhost:8084**. Vous obtiendrez une fenêtre similaire à celle-ci :

Fermez le navigateur et arrêtez le service luci :

```
[root@node2 ~]# service luci stop
Stop luci...                                               [  OK  ]
```

Gérer les Services de Haute Disponibilité

Retournez à l'interface luci sur **node1.fenestros.loc**. Cliquez sur le bouton **Service Groups** :

Important - Notez que le service **AdresseIP** a été basculé vers **node2.fenestros.loc**.

En cochant la case à gauche du service, vous pouvez utiliser cette interface pour **Démarrer** (*Start*), **Redémarrer** (*Restart*), **Désactiver** (*Disable*) ou **Supprimer** (*Delete*) le service :

En cliquant sur le nom du service, vous accédez à une interface qui vous permet de basculer manuellement le service vers un autre noeud et/ou de **Démarrer** (*Start*), **Redémarrer** (*Restart*), **Désactiver** (*Disable*) ou **Supprimer** (*Delete*) le service grâce aux boutons d'action dans le coins supérieur droite :

LAB #10 - Gérer les Noeuds d'un Cluster

Causer un Noeud de Quitter ou de Joindre un Cluster

Cliquez sur le bouton **Nodes** :

Cochez la case à côté de **node2.fenestros.loc** et cliquez sur le bouton **Leave Cluster** :

Important - Si vous sélectionnez tous les noeuds d'un cluster et vous cliquez sur **Leave Cluster**, le cluster s'arrête.

Dans la boîte de dialogue **Confirm Action**, cliquez sur le bouton **Proceed** :

Vous obtiendrez la fenêtre suivante :

Cliquez sur le bouton **Nodes** pour rafraîchir la page :

Important - Notez que **node2.fenestros.loc** est en rouge et que le statut est devenu **Not a cluster member**.

Pour joindre le noeud au cluster, sélectionnez-le et cliquez sur le bouton **Join Cluster** :

Important - Si vous sélectionnez tous les noeuds d'un cluster arrêté et vous cliquez sur **Join Cluster**, le cluster redémarre.

Vous obtiendrez la fenêtre suivante :

Important - Notez la présence du message en haut à droite de la fenêtre.

Cliquez sur le bouton **Nodes** pour rafraîchir la page :

Cliquez maintenant sur le bouton **Service Groups** :

Important - Notez que le service a été basculé vers **node1.fenestros.loc** du fait du départ de **node2.fenestros.loc** du cluster.

Basculez manuellement le service AdresseIP vers **node2.fenestros.loc**

Supprimer un Membre d'un Cluster

Pour supprimer un membre d'un cluster, il faut d'abord l'arrêter. Cliquez sur la case à côté de **node3.fenestros.loc** puis cliquez sur **Leave Cluster** :

Dans la boîte de dialogue **Confirm Action**, cliquez sur le bouton **Proceed** :

Cliquez sur le bouton **Nodes** pour rafraîchir la page :

Important - Notez que **node2.fenestros.loc** est en rouge et que le statut est devenu **Not a cluster member**.

Cliquez sur la case à côté de **node3.fenestros.loc** puis cliquez sur **Delete** :

Dans la boîte de dialogue **Confirm Action**, cliquez sur le bouton **Proceed** :

La fenêtre se rafraîchit automatiquement :

Important - Pour supprimer un cluster, il convient de supprimer tous les noeuds faisant partie du cluster.

Ajouter un Membre à un Cluster en Cours d'Exécution

Pour rajouter **node3.fenestros.loc** au cluster en cours d'exécution, cliquez sur le bouton **Add**. Vous obtiendrez la fenêtre suivante :

Remplissez le formulaire ainsi et cliquez sur le bouton **Add Nodes** :

Observez le rajout du noeud :

Important - Notez que les noeuds du cluster sont arrêtés pendant le processus.

A l'issu du processus, vous obtiendrez la fenêtre suivante :

Ajouter un Cluster Existant à luci

Pour ajouter un cluster existant à l'interface de luci, cliquez sur le bouton **Manage Clusters** puis sur le bouton **Add** :

Dans la fenêtre **Add Existing Cluster**, il suffit de saisir le nom d'un noeud faisant partie du cluster :

Configurer et Gérer un Cluster avec La Commande ccs

Introduction

La configuration en utilisant la commande **ccs** consiste en :

- La préparation de tous les nœuds,
- La création d'un cluster,
- La configuration du daemon fenced,
- La configuration des périphériques fence (*Fence Devices*),
- La configuration des domaines de basculement (*Failover Domains*),
- La création de ressources globales,
- La création des services en cluster,
- La configuration du disque quorum,
- La configuration des propriétés générales du cluster,
- Propagation du fichier de configuration du cluster à tous les nœuds du cluster.

LAB #11 - Préparation de tous les nœuds

node1.fenestros.loc

Installer et Configurer ricci

Lancez la machine virtuelle **node10**. Celle-ci étant un clone de la machine **node1**, ricci est déjà installé.

Démarrez puis configurez ricci :

```
[root@node1 ~]# service ricci status
ricci est arrêté
[root@node1 ~]# service ricci start
Démarrage de oddjobd :                                     [  OK  ]
generating SSL certificates...  done
Generating NSS database...  done
Démarrage de ricci :                                       [  OK  ]
[root@node1 ~]# chkconfig --list ricci
ricci           0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
[root@node1 ~]# chkconfig --level 345 ricci on
[root@node1 ~]# chkconfig --list ricci
ricci           0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche6:arrêt
```

Créez maintenant un mot de passe pour l'utilisateur ricci :

```
[root@node1 ~]# passwd ricci
Changement de mot de passe pour l'utilisateur ricci.
Nouveau mot de passe :
MOT DE PASSE INCORRECT : trop court
MOT DE PASSE INCORRECT : est trop simple
Retapez le nouveau mot de passe :
```

```
passwd : mise à jour réussie de tous les jetons d'authentification.
```

Installer Cluster Configuration System

Installez le paquet ccs sur **node1.fenestros.loc** :

```
[root@node1 ~]# yum install ccs
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
 * atomic: www2.atomicorp.com
 * base: holmes.umflint.edu
 * epel: fedora-epel.mirror.lstn.net
 * extras: mirror.cs.vt.edu
 * rpmforge: repoforge.spinellicreations.com
 * updates: centos.someimage.com
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package ccs.i686 0:0.16.2-69.el6_5.1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================
 Package      Arch          Version                      Repository        Size
================================================================================
Installing:
 ccs          i686          0.16.2-69.el6_5.1            updates           49 k

Transaction Summary
================================================================================
Install       1 Package(s)
```

```
Total download size: 49 k
Installed size: 287 k
Is this ok [y/N]: y
```

Modifiez /etc/hosts

Créez une entrée pour chaque nœud dans le fichier **/etc/hosts** :

hosts

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.3.15   node1.fenestros.loc
10.0.3.16   node2.fenestros.loc
10.0.3.17   node3.fenestros.loc
```

node2.fenestros.loc

Installer et Configurer ricci

Lancez la machine virtuelle **node20**. Celle-ci étant un clone de la machine **node2**, ricci est déjà installé.

Démarrez puis configurez ricci :

```
[root@node2 ~]# service ricci status
ricci est arrêté
[root@node2 ~]# service ricci start
Démarrage de oddjobd :                                     [  OK  ]
generating SSL certificates...  done
```

```
Generating NSS database...  done
Démarrage de ricci :                                       [  OK  ]
[root@node2 ~]# chkconfig --level 345 ricci on
[root@node2 ~]# chkconfig --list ricci
ricci           0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche6:arrêt
```

Créez maintenant un mot de passe pour l'utilisateur ricci :

```
[root@node2 ~]# passwd ricci
Changement de mot de passe pour l'utilisateur ricci.
Nouveau mot de passe :
MOT DE PASSE INCORRECT : trop court
MOT DE PASSE INCORRECT : est trop simple
Retapez le nouveau mot de passe :
passwd : mise à jour réussie de tous les jetons d'authentification.
```

Modifiez /etc/hosts

Créez une entrée pour chaque nœud dans le fichier **/etc/hosts** :

hosts

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.3.15   node1.fenestros.loc
10.0.3.16   node2.fenestros.loc
10.0.3.17   node3.fenestros.loc
```

node3.fenestros.loc

Installer et Configurer ricci

Lancez la machine virtuelle **node30**. Celle-ci étant un clone de la machine **node3**, ricci est déjà installé.

Démarrez puis configurez ricci :

```
[root@node3 ~]# service ricci status
ricci est arrêté
[root@node3 ~]# service ricci start
Démarrage de oddjobd :                                     [  OK  ]
generating SSL certificates...  done
Generating NSS database...  done
Démarrage de ricci :                                       [  OK  ]
[root@node3 ~]# chkconfig --level 345 ricci on
[root@node3 ~]# chkconfig --list ricci
ricci           0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche6:arrêt
```

Créez maintenant un mot de passe pour l'utilisateur ricci :

```
[root@node3 ~]# passwd ricci
Changement de mot de passe pour l'utilisateur ricci.
Nouveau mot de passe :
MOT DE PASSE INCORRECT : trop court
MOT DE PASSE INCORRECT : est trop simple
Retapez le nouveau mot de passe :
passwd : mise à jour réussie de tous les jetons d'authentification.
```

Modifiez /etc/hosts

Créez une entrée pour chaque nœud dans le fichier **/etc/hosts** :

hosts

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.3.15   node1.fenestros.loc
10.0.3.16   node2.fenestros.loc
10.0.3.17   node3.fenestros.loc
```

LAB #12 - Création d'un Cluster

Sur **node1.fenestros.loc** saisissez la commande suivante pour créer un cluster dénommé **moncluster** :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --createcluster moncluster
node1.fenestros.loc password: ricci
```

> **Important** - Vous avez appelé votre cluster **moncluster**. Le nom d'un cluster est limité à **15** caractères. Le mot de passe ne sera pas visible en clair. Si le fiochier cluster.conf existe déjà, il sera écrasé.

Ajoutez maintenant chaque nœud, numéroté de 1 à 3, au cluster précedemment créé, chacun ayant une vote :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --addnode node1.fenestros.loc --votes 1 --nodeid 1
Node node1.fenestros.loc added.
[root@node1 ~]# ccs -h node1.fenestros.loc --addnode node2.fenestros.loc --votes 1 --nodeid 2
Node node2.fenestros.loc added.
[root@node1 ~]# ccs -h node1.fenestros.loc --addnode node3.fenestros.loc --votes 1 --nodeid 3
Node node3.fenestros.loc added.
```

Le commandes ci-dessus crée le fichier **/etc/cluster/cluster.conf** sur **node1.fenestros.loc** :

cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="4" name="moncluster">
    <fence_daemon/>
    <clusternodes>
        <clusternode name="node1.fenestros.loc" nodeid="1" votes="1"/>
        <clusternode name="node2.fenestros.loc" nodeid="2" votes="1"/>
        <clusternode name="node3.fenestros.loc" nodeid="3" votes="1"/>
    </clusternodes>
    <cman/>
    <fencedevices/>
    <rm>
        <failoverdomains/>
        <resources/>
    </rm>
</cluster>
```

Pour vérifier la configuration des nœuds dans le cluster, utilisez la commande suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --lsnodes
node1.fenestros.loc: votes=1, nodeid=1
node2.fenestros.loc: votes=1, nodeid=2
node3.fenestros.loc: votes=1, nodeid=3
```

Vous devez maintenant propagé le fichier cluster.conf aux autres nœuds :

```
[root@node1 ~]# ccs -f /etc/cluster/cluster.conf -h node2.fenestros.loc --setconf
node2.fenestros.loc password: ricci
[root@node1 ~]# ccs -f /etc/cluster/cluster.conf -h node3.fenestros.loc --setconf
node3.fenestros.loc password: ricci
```

Important - Le mot de passe ne sera pas visible en clair.

Pour vérifier si tous les nœuds sont synchroniser, utilisez la commande suivante :

```
[root@node1 ~]# ccs -f /etc/cluster/cluster.conf --checkconf
All nodes in sync.
```

Vous pouvez aussi synchroniser et activer le fichier cluster.conf en utilisant les commandes suivantes :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --sync --activate
[root@node1 ~]# ccs -h node2.fenestros.loc --sync --activate
[root@node1 ~]# ccs -h node3.fenestros.loc --sync --activate
```

Dans ce cas, utilisez la comande suivante pour vérifier la configuration :

```
[root@node1 ~]# ccs -h node2.fenestros.loc --checkconf
All nodes in sync.
```

Configurez les Services Cluster

Démarrez les cluster sur chaque machine virtuelle :

```
[root@node1 ~]# chkconfig --level 345 cman on
[root@node1 ~]# chkconfig --level 345 rgmanager on
[root@node1 ~]# reboot
```

```
[root@node2 ~]# chkconfig --level 345 cman on
[root@node1 ~]# chkconfig --level 345 rgmanager on
[root@node2 ~]# reboot
```

```
[root@node3 ~]# chkconfig --level 345 cman on
[root@node1 ~]# chkconfig --level 345 rgmanager on
[root@node3 ~]# reboot
```

Qaund les machines virtuelles ont redémarrés, vérifiez l'existance du cluster :

```
[root@node1 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M     12   2013-12-14 21:35:54  node1.fenestros.loc
   2   M     16   2013-12-14 21:36:34  node2.fenestros.loc
   3   M     20   2013-12-14 21:36:51  node3.fenestros.loc
```

LAB #13 - Configuration du Daemon Fenced

Pour configurer le daemon Fenced, il convient d'utiliser la commande suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --setfencedaemon post_fail_delay=5 post_join_delay=25
```

Rappelez-vous que les deux paramètres du daemon sont :

- **Post Fail Delay** - Le nombre de secondes qu'attend le daemon fenced avant de réagir quand un nœud dans un **Domaine Fence** devient défaillant. La valeur par défaut est **0**,
- **Post Join Delay** - Le nombre de secondes qu'attend le daemon fenced avant de réagir quand un nœud rejoint un **Domaine Fence**. La valeur par défaut est **6**. La valeur conseillée est entre **20** et **30**.

LAB #14 - Configuration des Périphériques Fence

Pour ajouter le périphérique fence APC1, utilisez la commande suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --addfencedev APC1 agent=fence_apc ipaddr=10.0.3.50 login=root
passwd=password
```

Configurez deux autres Périphériques Fence APC Power Switch ayant respectivement les adresses IP 10.0.3.51 et 10.0.3.52 :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --addfencedev APC2 agent=fence_apc ipaddr=10.0.3.51 login=root
```

```
passwd=password
[root@node1 ~]# ccs -h node1.fenestros.loc --addfencedev APC3 agent=fence_apc ipaddr=10.0.3.52 login=root
passwd=password
```

Pour supprimer le Périphérique Fence APC3, utilisez la commande suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --rmfencedev APC3
```

Contrôlez ensuite votre fichier **/etc/cluster/cluster.conf** :

```
[root@node1 ~]# cat /etc/cluster/cluster.conf
<?xml version="1.0"?>
<cluster config_version="12" name="moncluster">
    <fence_daemon post_fail_delay="5" post_join_delay="25"/>
    <clusternodes>
        <clusternode name="node1.fenestros.loc" nodeid="1" votes="1"/>
        <clusternode name="node2.fenestros.loc" nodeid="2" votes="1"/>
        <clusternode name="node3.fenestros.loc" nodeid="3" votes="1"/>
    </clusternodes>
    <cman/>
    <fencedevices>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.50" login="root" name="APC1" passwd="password"/>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.51" login="root" name="APC2" passwd="password"/>
    </fencedevices>
    <rm>
        <failoverdomains/>
        <resources/>
    </rm>
</cluster>
```

Important - Notez que ce fichier ne fait pas référence à l'APC3.

Une autre façon de voir le périphériques fence configurés pour le cluster est d'utiliser la command ccs :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --lsfencedev
APC1: passwd=password, login=root, ipaddr=10.0.3.50, agent=fence_apc
APC2: passwd=password, login=root, ipaddr=10.0.3.51, agent=fence_apc
```

Pour voir la liste des **agents fence** *disponibles* pour le cluster, utilisez la commande suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --lsfenceopts
fence_apc - Fence agent for APC over telnet/ssh
fence_apc_snmp - Fence agent for APC over SNMP
fence_bladecenter - Fence agent for IBM BladeCenter
fence_bladecenter_snmp - Fence agent for IBM BladeCenter over SNMP
fence_brocade - Fence agent for Brocade over telnet
fence_cisco_mds - Fence agent for Cisco MDS
fence_cisco_ucs - Fence agent for Cisco UCS
fence_drac - fencing agent for Dell Remote Access Card
fence_drac5 - Fence agent for Dell DRAC CMC/5
fence_eaton_snmp - Fence agent for Eaton over SNMP
fence_egenera - I/O Fencing agent for the Egenera BladeFrame
fence_eps - Fence agent for ePowerSwitch
fence_hpblade - Fence agent for HP BladeSystem
fence_ibmblade - Fence agent for IBM BladeCenter over SNMP
fence_idrac - Fence agent for IPMI over LAN
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 over LAN
fence_ilo4 - Fence agent for IPMI over LAN
fence_ilo_mp - Fence agent for HP iLO MP
fence_imm - Fence agent for IPMI over LAN
fence_intelmodular - Fence agent for Intel Modular
fence_ipdu - Fence agent for iPDU over SNMP
fence_ipmilan - Fence agent for IPMI over LAN
```

```
fence_kdump - Fence agent for use with kdump
fence_pcmk - Helper that presents a RHCS-style interface to stonith-ng for CMAN based clusters
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_sanbox2 - Fence agent for QLogic SANBox2 FC switches
fence_sanlock - Fence agent for watchdog and shared storage
fence_scsi - fence agent for SCSI-3 persistent reservations
fence_virsh - Fence agent for virsh
fence_virt - Fence agent for virtual machines
fence_vmware - Fence agent for VMWare
fence_vmware_soap - Fence agent for VMWare over SOAP API
fence_wti - Fence agent for WTI
fence_xvm - Fence agent for virtual machines
```

Pour voir les **options fence** pour un **agent fence** donné, utilisez la commande suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --lsfenceopts fence_apc
fence_apc - Fence agent for APC over telnet/ssh
  Required Options:
  Optional Options:
    option: No description available
    action: Fencing Action
    ipaddr: IP Address or Hostname
    login: Login Name
    passwd: Login password or passphrase
    passwd_script: Script to retrieve password
    secure: SSH connection
    port: Physical plug number or name of virtual machine
    identity_file: Identity file for ssh
    switch: Physical switch number on device
    inet4_only: Forces agent to use IPv4 addresses only
    inet6_only: Forces agent to use IPv6 addresses only
    ipport: TCP port to use for connection with device
```

```
    cmd_prompt: Force command prompt
    verbose: Verbose mode
    debug: Write debug information to given file
    version: Display version information and exit
    help: Display help and exit
    separator: Separator for CSV created by operation list
    power_timeout: Test X seconds for status change after ON/OFF
    shell_timeout: Wait X seconds for cmd prompt after issuing command
    login_timeout: Wait X seconds for cmd prompt after login
    power_wait: Wait X seconds after issuing ON/OFF
    delay: Wait X seconds before fencing is started
    retry_on: Count of attempts to retry power on
```

Important - La documentation sur la configuration de chaque Périphérique Fence peut être consultée à **cette adresse**.

Pour configurer une **Méthode Fence** appelé **APC1** à **node2.fenetros.loc**, la commande est la suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --addmethod APC1 node2.fenestros.loc
Method APC1 added to node2.fenestros.loc.
```

Pour ajouter une **Instance Fence** à la **Méthode fence**, utilisez la commande suivante :

```
ccs -h node1.fenestros.loc --addfenceinst APC1 node2.fenestros.loc APC1 port=2
```

Synchronisez maintenant les nœuds :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --sync --activate
[root@node1 ~]# ccs -h node2.fenestros.loc --sync --activate
[root@node1 ~]# ccs -h node3.fenestros.loc --sync --activate
[root@node1 ~]# ccs -h node2.fenestros.loc --checkconf
```

Pour supprimer une méthode fence, la commande ccs est utilisée avec l'option **–rmmethod** :

```
[root@node1 ~]# ccs -h node1.fenestros.loc  --rmmethod APC1 node2.fenestros.loc
Method APC1 removed from node2.fenestros.loc.
```

LAB #15 - La Configuration des Domaines de Basculement

Pour créer un domaine de basculement identique à l'exemple précédement configuré en utilisant Conga, la commande est la suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --addfailoverdomain Failover1 ordered
```

Pour inclure **node1.fenestros.loc** et **node2.fenestros.loc** dans **Failover1**, les commandes sont :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --addfailoverdomainnode Failover1 node1.fenestros.loc 1
[root@node1 ~]# ccs -h node1.fenestros.loc --addfailoverdomainnode Failover1 node2.fenestros.loc 2
```

Important - Le chiffre à la fin de chaque ligne est la **priorité** du mode **ordered**.

Pour voir les domaines de basculement configurés, utilisez la commande suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --lsfailoverdomain
Failover1: restricted=0, ordered=1, nofailback=0
  node1.fenestros.loc: priority=1
  node2.fenestros.loc: priority=2
```

Important - Pour supprimer un domaine de basculement, la commande ccs peur être utilisée avec l'option **–rmfailoverdomain**.

LAB #16 - La Création de Ressources Globales

Pour voir la liste des ressources globales *disponibles* dans le cluster, utilisez la commande suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --lsresourceopts
service - Defines a service (resource group).
ASEHAagent - Sybase ASE Failover Instance
SAPDatabase - Manages any SAP database (based on Oracle, MaxDB, or DB2)
SAPInstance - SAP instance resource agent
apache - Defines an Apache web server
clusterfs - Defines a cluster file system mount.
fs - Defines a file system mount.
ip - This is an IP address.
lvm - LVM Failover script
mysql - Defines a MySQL database server
named - Defines an instance of named server
netfs - Defines an NFS/CIFS file system mount.
nfsclient - Defines an NFS client.
nfsexport - This defines an NFS export.
nfsserver - This defines an NFS server resource.
openldap - Defines an Open LDAP server
oracledb - Oracle 10g/11g Failover Instance
orainstance - Oracle 10g Failover Instance
oralistener - Oracle 10g Listener Instance
postgres-8 - Defines a PostgreSQL server
samba - Dynamic smbd/nmbd resource agent
script - LSB-compliant init script as a clustered resource.
tomcat-6 - Defines a Tomcat server
vm - Defines a Virtual Machine
```

Pour mettre en place une adresse IP en tant que resource globale, il convient d'utiliser la commande suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --addresource ip address=10.0.3.100/24
```

Pour voir la liste des resources actuellement configurées, la commande ccs est utilisée avec l'opton **–lsservices** :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --lsservices
resources:
  ip: address=10.0.3.100/24
```

LAB #17 - La Création des Services en Cluster

Pour voir la liste des services *disponibles* dans le cluster, utilisez la commande suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --lsserviceopts
service - Defines a service (resource group).
ASEHAagent - Sybase ASE Failover Instance
SAPDatabase - Manages any SAP database (based on Oracle, MaxDB, or DB2)
SAPInstance - SAP instance resource agent
apache - Defines an Apache web server
clusterfs - Defines a cluster file system mount.
fs - Defines a file system mount.
ip - This is an IP address.
lvm - LVM Failover script
mysql - Defines a MySQL database server
named - Defines an instance of named server
netfs - Defines an NFS/CIFS file system mount.
nfsclient - Defines an NFS client.
nfsexport - This defines an NFS export.
nfsserver - This defines an NFS server resource.
openldap - Defines an Open LDAP server
oracledb - Oracle 10g/11g Failover Instance
orainstance - Oracle 10g Failover Instance
oralistener - Oracle 10g Listener Instance
postgres-8 - Defines a PostgreSQL server
samba - Dynamic smbd/nmbd resource agent
script - LSB-compliant init script as a clustered resource.
```

```
tomcat-6 - Defines a Tomcat server
vm - Defines a Virtual Machine
```

Pour créer un service en cluster appelé **ip_address** utilisant le domaine de basculement **Failover1** avec un **Recovery Mode** de relocate, utilisez la commande suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --addservice ip_address domain=Failover1 recovery=relocate
```

Il existe quatre types de **Recovery Mode** :

Choix	**Description**
Relocate	En cas d'échec, le système essaie de démarrer le service sur un autre nœud.
Restart	En cas d'échec, le système essayera de re-démarrer le service sur le même nœud avant de démarrer le service sur un autre nœud.
Restart-Disable	En cas d'échec, le service est redémarré mais en cas d'échec du redémarrage le service est désactivé au lieu d'être transférer vers un autre nœud.
Disable	En cas d'échec d'un composant d'un groupe de ressources, le système désactive le groupe de ressources.

Pour ajouter la ressource **ip** à ce service, il convient maintenant d'utiliser l'option **–addsubservice** :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --addsubservice ip_address ip address=10.0.3.100/24
```

Vérifiez la mise en place :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --lsservices
service: name=ip_address, domain=Failover1, recovery=relocate
  ip: address=10.0.3.100/24
resources:
  ip: address=10.0.3.100/24
```

Les options utilisées dans cette commande peuvent être consultées avec la commande suivante :

```
[root@node1 ~]# ccs -f node1.fenestros.loc --lsserviceopts ip
ip - This is an IP address.
```

```
  Required Options:
    address: IP Address
  Optional Options:
    family: Family
    monitor_link: Monitor NIC Link
    nfslock: Enable NFS lock workarounds
    sleeptime: Amount of time (seconds) to sleep.
    disable_rdisc: Disable updating of routing using RDISC protocol
    prefer_interface: Network interface
    __independent_subtree: Treat this and all children as an independent subtree.
    __enforce_timeouts: Consider a timeout for operations as fatal.
    __max_failures: Maximum number of failures before returning a failure to a status check.
    __failure_expire_time: Amount of time before a failure is forgotten.
    __max_restarts: Maximum number restarts for an independent subtree before giving up.
    __restart_expire_time: Amount of time before a failure is forgotten for an independent subtree.
```

Supprimer des Services et Ressources en Cluster

Pour supprimer un service, utilisez l'option **–rmservice** de la commande **ccs** :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --lsservices
service: name=ip_address, domain=Failover1, recovery=relocate
  ip: address=10.0.3.100/24
resources:
  ip: address=10.0.3.100/24
[root@node1 ~]# ccs -h node1.fenestros.loc --rmservice ip_address
[root@node1 ~]# ccs -h node1.fenestros.loc --lsservices
resources:
  ip: address=10.0.3.100/24
```

Pour supprimer une ressource, utilisez l'option **–rmresource** de la commande **ccs** :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --rmresource ip
Unable to find matching resource: ip, []
[root@node1 ~]# ccs -h node1.fenestros.loc --rmresource ip address=10.0.3.100/24
[root@node1 ~]# ccs -h node1.fenestros.loc --lsservices
resources:
```

Important - N'oubliez pas de spécifier les options de la ressource que vous souhiatez supprimer, sinon la commande échoue.

LAB #18 - Configuration des Propriétés Générales du Cluster

Version de la Configuration du Cluster

La version de la configuration du cluster peut être gérer grâce à la commande ccs :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --getversion
29
[root@node1 ~]# ccs -h node1.fenestros.loc --setversion 30
[root@node1 ~]# ccs -h node1.fenestros.loc --getversion
30
```

Adresse de la multidiffusion

L'adresse de multidiffusion peut être spécifiée avec la commande ccs, Par exemple :

```
ccs -h node1.fenestros.loc --setmulticast multicastaddress
```

Cluster à Deux Nœuds

Afin d'éviter un problème de quorum dans le cas d'un cluster à deux nœuds, utilisez la commande suivante :

```
ccs -h node1.fenestros.loc --setcman two_node=1 expected_votes=1
```

Journalisation

Pour activer le déboggage du cluster, utilisez la commande suivante :

```
ccs -h node1.fenestros.loc --setlogging debug=on
```

Les journaux se trouyvent par défaut dans le répertoire **/var/log/cluster** :

```
[root@node1 ~]# ls /var/log/cluster
corosync.log  dlm_controld.log  fenced.log  gfs_controld.log  rgmanager.log
```

Dernièrement, propagez maintenant le fichier de configuration du cluster vers les deux autres nœuds :

```
[root@node1 ~]# ccs -h node2.fenestros.loc --sync --activate
[root@node1 ~]# ccs -h node3.fenestros.loc --sync --activate
[root@node1 ~]# ccs -h node2.fenestros.loc --checkconf
All nodes in sync.
```

LAB #19 - Gérer un Cluster avec La Commande ccs

Causer à un Nœud de Quitter un Cluster

Pour enlever un nœud d'un cluster, il convient d'utiliser la commande suivante :

```
[root@node1 ~]# ccs -h node2.fenestros.loc --stop
[root@node1 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M     40   2013-12-15 17:42:50  node1.fenestros.loc
   2   X     44                        node2.fenestros.loc
   3   M     44   2013-12-15 17:42:50  node3.fenestros.loc
```

Important - Pour supprimer un complètement nœud, il convient d'utiliser l'option **–rmnode** de la commande ccs.

Causer à un Nœud de Joindre un Cluster

Pour enlever un nœud d'un cluster, il convient d'utiliser la commande suivante :

```
[root@node1 ~]# ccs -h node2.fenestros.loc --start
[root@node1 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M     40   2013-12-15 17:42:50  node1.fenestros.loc
   2   M     52   2013-12-15 17:44:58  node2.fenestros.loc
   3   M     44   2013-12-15 17:42:50  node3.fenestros.loc
```

Arrêter un cluster

Pour arrêter un cluster, utilisez la commande suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --stopall
Stopped node2.fenestros.loc
Stopped node3.fenestros.loc
Stopped node1.fenestros.loc
```

Démarrer un cluster

Pour démarrer un cluster, utilisez la commande suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --startall
Started node2.fenestros.loc
Started node3.fenestros.loc
Started node1.fenestros.loc
```

Diagnostiquer des Problèmes de Configuration

Pour diagnostiquer des problèmes au niveau de la configuration, utilisez la commande suivante :

```
[root@node1 ~]# ccs -h node1.fenestros.loc --checkconf
All nodes in sync.
```

ou la commande suivante :

```
[root@node1 ~]# ccs -f /etc/cluster/cluster.conf --checkconf
All nodes in sync.
```

Configurer et Gérer un Cluster avec des Outils de Ligne de Commande

Introduction

La configuration en utilisant la ligne de commande consiste en :

- La préparation de tous les nœuds,
- La création d'un cluster,

- La configuration du daemon fenced,
- La configuration des périphériques fence (*Fence Devices*),
- La configuration des domaines de basculement (*Failover Domains*),
- La création de ressources globales,
- La création des services en cluster,
- Propagation du fichier de configuration du cluster à tous les nœuds du cluster.

LAB #20 - Préparation de tous les nœuds

node1.fenestros.loc

Installer et Configurer ricci

Lancez la machine virtuelle **node100**. Celle-ci étant un clone de la machine **node1**, ricci est déjà installé.

Démarrez puis configurez ricci :

```
[root@node1 ~]# service ricci status
ricci est arrêté
[root@node1 ~]# service ricci start
Démarrage de oddjobd :                                     [  OK  ]
generating SSL certificates...  done
Generating NSS database...  done
Démarrage de ricci :                                       [  OK  ]
[root@node1 ~]# chkconfig --list ricci
ricci           0:arrêt    1:arrêt    2:arrêt    3:arrêt    4:arrêt    5:arrêt    6:arrêt
[root@node1 ~]# chkconfig --level 345 ricci on
[root@node1 ~]# chkconfig --list ricci
ricci           0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche6:arrêt
```

Créez maintenant un mot de passe pour l'utilisateur ricci :

```
[root@node1 ~]# passwd ricci
Changement de mot de passe pour l'utilisateur ricci.
Nouveau mot de passe :
MOT DE PASSE INCORRECT : trop court
MOT DE PASSE INCORRECT : est trop simple
Retapez le nouveau mot de passe :
passwd : mise à jour réussie de tous les jetons d'authentification.
```

Modifiez /etc/hosts

Créez une entrée pour chaque nœud dans le fichier **/etc/hosts** :

hosts

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.3.15   node1.fenestros.loc
10.0.3.16   node2.fenestros.loc
10.0.3.17   node3.fenestros.loc
```

node2.fenestros.loc

Installer et Configurer ricci

Lancez la machine virtuelle **node200**. Celle-ci étant un clone de la machine **node2**, ricci est déjà installé.

Démarrez puis configurez ricci :

```
[root@node2 ~]# service ricci status
```

```
ricci est arrêté
[root@node2 ~]# service ricci start
Démarrage de oddjobd :                                     [  OK  ]
generating SSL certificates...  done
Generating NSS database...  done
Démarrage de ricci :                                       [  OK  ]
[root@node2 ~]# chkconfig --level 345 ricci on
[root@node2 ~]# chkconfig --list ricci
ricci           0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche6:arrêt
```

Créez maintenant un mot de passe pour l'utilisateur ricci :

```
[root@node2 ~]# passwd ricci
Changement de mot de passe pour l'utilisateur ricci.
Nouveau mot de passe :
MOT DE PASSE INCORRECT : trop court
MOT DE PASSE INCORRECT : est trop simple
Retapez le nouveau mot de passe :
passwd : mise à jour réussie de tous les jetons d'authentification.
```

Modifiez /etc/hosts

Créez une entrée pour chaque nœud dans le fichier **/etc/hosts** :

hosts

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.3.15   node1.fenestros.loc
10.0.3.16   node2.fenestros.loc
10.0.3.17   node3.fenestros.loc
```

node3.fenestros.loc

Installer et Configurer ricci

Lancez la machine virtuelle **node300**. Celle-ci étant un clone de la machine **node3**, ricci est déjà installé.

Démarrez puis configurez ricci :

```
[root@node3 ~]# service ricci status
ricci est arrêté
[root@node3 ~]# service ricci start
Démarrage de oddjobd :                                     [  OK  ]
generating SSL certificates...  done
Generating NSS database...  done
Démarrage de ricci :                                       [  OK  ]
[root@node3 ~]# chkconfig --level 345 ricci on
[root@node3 ~]# chkconfig --list ricci
ricci           0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche6:arrêt
```

Créez maintenant un mot de passe pour l'utilisateur ricci :

```
[root@node3 ~]# passwd ricci
Changement de mot de passe pour l'utilisateur ricci.
Nouveau mot de passe :
MOT DE PASSE INCORRECT : trop court
MOT DE PASSE INCORRECT : est trop simple
Retapez le nouveau mot de passe :
passwd : mise à jour réussie de tous les jetons d'authentification.
```

Modifiez /etc/hosts

Créez une entrée pour chaque nœud dans le fichier **/etc/hosts** :

hosts

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.3.15   node1.fenestros.loc
10.0.3.16   node2.fenestros.loc
10.0.3.17   node3.fenestros.loc
```

LAB #21 - Création d'un Cluster

Sur **chaque noeud** utilisez le fichier suivant pour créer **/etc/cluster/cluster.conf** :

cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="1" name="moncluster">
   <clusternodes>
     <clusternode name="node1.fenestros.loc" nodeid="1" votes="1"/>
     <clusternode name="node2.fenestros.loc" nodeid="2" votes="1"/>
     <clusternode name="node3.fenestros.loc" nodeid="3" votes="1"/>
   </clusternodes>
   <fencedevices>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

Configurez les Services Cluster

Configurez les services cluster sur chaque machine virtuelle :

```
[root@node1 ~]# chkconfig --level 2345 cman on
[root@node1 ~]# reboot
```

```
[root@node2 ~]# chkconfig --level 2345 cman on
[root@node2 ~]# reboot
```

```
[root@node3 ~]# chkconfig --level 2345 cman on
[root@node3 ~]# reboot
```

Quand les machines virtuelles ont redémarrés, vérifiez l'existance du cluster :

```
[root@node1 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M     16   2013-12-15 19:26:19  node1.fenestros.loc
   2   M     20   2013-12-15 19:26:19  node2.fenestros.loc
   3   M     24   2013-12-15 19:26:27  node3.fenestros.loc
```

LAB #22 - Configuration du Daemon Fenced

La configuration Daemon Fenced se fait par l'édition du fichier **/etc/cluster/cluster.conf** :

cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="1" name="moncluster">
   <clusternodes>
     <clusternode name="node1.fenestros.loc" nodeid="1" votes="1"/>
     <clusternode name="node2.fenestros.loc" nodeid="2" votes="1"/>
```

```
    <clusternode name="node3.fenestros.loc" nodeid="3" votes="1"/>
  </clusternodes>
  <fencedevices>
  </fencedevices>
  <rm>
  </rm>
</cluster>
```

Par exemple :

cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="2" name="moncluster">
<fence_daemon post_fail_delay="5" post_join_delay="25"/>
  <clusternodes>
    <clusternode name="node1.fenestros.loc" nodeid="1" votes="1"/>
    <clusternode name="node2.fenestros.loc" nodeid="2" votes="1"/>
    <clusternode name="node3.fenestros.loc" nodeid="3" votes="1"/>
  </clusternodes>
  <fencedevices>
  </fencedevices>
  <rm>
  </rm>
</cluster>
```

Rappelez-vous que les deux paramètres du daemon sont :

- **Post Fail Delay** - Le nombre de secondes qu'attend le daemon fenced avant de réagir quand un nœud dans un **Domaine Fence** devient défaillant. La valeur par défaut est **0**,
- **Post Join Delay** - Le nombre de secondes qu'attend le daemon fenced avant de réagir quand un nœud rejoint un **Domaine Fence**. La valeur par défaut est **6**. La valeur conseillée est entre **20** et **30**.

Validez le fichier sur le schéma du cluster (/usr/share/cluster/cluster.rng) :

```
[root@node1 ~]# ccs_config_validate
Configuration validates
```

LAB #23 - Configuration des Périphériques Fence

La configuration d'un périphérique fence se fait par l'édition du fichier **/etc/cluster/cluster.conf** :

cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="2" name="moncluster">
<fence_daemon post_fail_delay="5" post_join_delay="25"/>
   <clusternodes>
     <clusternode name="node1.fenestros.loc" nodeid="1" votes="1"/>
     <clusternode name="node2.fenestros.loc" nodeid="2" votes="1"/>
     <clusternode name="node3.fenestros.loc" nodeid="3" votes="1"/>
   </clusternodes>
   <fencedevices>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

Par exemple :

cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="3" name="moncluster">
```

```
<fence_daemon post_fail_delay="5" post_join_delay="25"/>
   <clusternodes>
     <clusternode name="node1.fenestros.loc" nodeid="1" votes="1"/>
     <clusternode name="node2.fenestros.loc" nodeid="2" votes="1"/>
     <clusternode name="node3.fenestros.loc" nodeid="3" votes="1"/>
   </clusternodes>
   <fencedevices>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.50" login="root" name="APC1" passwd="password"/>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.51" login="root" name="APC2" passwd="password"/>
   </fencedevices>
   <rm>
   </rm>
</cluster>
```

Validez le fichier sur le schéma du cluster (/usr/share/cluster/cluster.rng) :

```
[root@node1 ~]# ccs_config_validate
Configuration validates
```

Une méthode est configurée de la même manière :

cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="4" name="moncluster">
<fence_daemon post_fail_delay="5" post_join_delay="25"/>
   <clusternodes>
     <clusternode name="node1.fenestros.loc" nodeid="1" votes="1">
         <fence>
            <method name="APC1">
              <device name="APC1" port="1"/>
             </method>
         </fence>
```

```
    </clusternode>
    <clusternode name="node2.fenestros.loc" nodeid="2" votes="1"/>
    <clusternode name="node3.fenestros.loc" nodeid="3" votes="1"/>
  </clusternodes>
  <fencedevices>
      <fencedevice agent="fence_apc" ipaddr="10.0.3.50" login="root" name="APC1" passwd="password"/>
      <fencedevice agent="fence_apc" ipaddr="10.0.3.51" login="root" name="APC2" passwd="password"/>
  </fencedevices>
  <rm>
  </rm>
</cluster>
```

Validez le fichier sur le schéma du cluster (/usr/share/cluster/cluster.rng) :

```
[root@node1 ~]# ccs_config_validate
Configuration validates
```

LAB #23 - La Configuration des Domaines de Basculement

Pour créer un domaine de basculement identique à l'exemple précédement configuré en utilisant la commande css, éditez de nouveau le fichier **/etc/cluster/cluster.conf** :

cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="5" name="moncluster">
<fence_daemon post_fail_delay="5" post_join_delay="25"/>
  <clusternodes>
    <clusternode name="node1.fenestros.loc" nodeid="1" votes="1">
        <fence>
           <method name="APC1">
```

```
              <device name="APC1" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node2.fenestros.loc" nodeid="2" votes="1"/>
     <clusternode name="node3.fenestros.loc" nodeid="3" votes="1"/>
   </clusternodes>
   <fencedevices>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.50" login="root" name="APC1" passwd="password"/>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.51" login="root" name="APC2" passwd="password"/>
   </fencedevices>
   <rm>
        <failoverdomains>
            <failoverdomain name="Failover1" nofailback="0" ordered="1" restricted="0">
                <failoverdomainnode name="node1.fenestros.loc" priority="1"/>
                <failoverdomainnode name="node2.fenestros.loc" priority="2"/>
            </failoverdomain>
        </failoverdomains>
   </rm>
</cluster>
```

Validez le fichier sur le schéma du cluster (/usr/share/cluster/cluster.rng) :

```
[root@node1 ~]# ccs_config_validate
Configuration validates
```

LAB #24 - La Création de Ressources Globales

La création d'une ressource globale s'effectue en éditant la section **<rm>** du fichier **/etc/cluster/cluster.conf** :

cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="6" name="moncluster">
<fence_daemon post_fail_delay="5" post_join_delay="25"/>
   <clusternodes>
     <clusternode name="node1.fenestros.loc" nodeid="1" votes="1">
         <fence>
            <method name="APC1">
              <device name="APC1" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node2.fenestros.loc" nodeid="2" votes="1"/>
     <clusternode name="node3.fenestros.loc" nodeid="3" votes="1"/>
   </clusternodes>
   <fencedevices>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.50" login="root" name="APC1" passwd="password"/>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.51" login="root" name="APC2" passwd="password"/>
   </fencedevices>
   <rm>
        <failoverdomains>
            <failoverdomain name="Failover1" nofailback="0" ordered="1" restricted="0">
                <failoverdomainnode name="node1.fenestros.loc" priority="1"/>
                <failoverdomainnode name="node2.fenestros.loc" priority="2"/>
            </failoverdomain>
        </failoverdomains>
        <resources>
            <ip address="10.0.3.100/24" monitor_link="yes" sleeptime="10"/>
        </resources>
   </rm>
</cluster>
```

Validez le fichier sur le schéma du cluster (/usr/share/cluster/cluster.rng) :

```
[root@node1 ~]# ccs_config_validate
Configuration validates
```

Important - Par défaut, tous les 10 secondes, **rgmanager** recherche dans l'arbre des ressources celles qui ont dépassé l'intervalle de vérification. Chaque agent de ressource spécifie l'intervalle de vérification pour la ressource concernée. Cette intervalle peut être modifiée en éditant le fichier **/etc/cluster/cluster.conf** en spécifiant une intervalle pour une ressource spécifique avec la balise **<action>** : **<action name="status" depth="*" interval="10" />**. Certains agents fournissent plusieurs **profondeurs** (*depths*) de vérification. Plus profonde est la vérification, plus complet est la vérification. Dans l'exemple précédent, l'intervalle concerne tous les profondeurs.

LAB #25 - La Création des Services en Cluster

La création d'un service globale s'effectue en éditant la section **<rm>** du fichier **/etc/cluster/cluster.conf** :

cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="7" name="moncluster">
<fence_daemon post_fail_delay="5" post_join_delay="25"/>
   <clusternodes>
     <clusternode name="node1.fenestros.loc" nodeid="1" votes="1">
         <fence>
            <method name="APC1">
              <device name="APC1" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node2.fenestros.loc" nodeid="2" votes="1"/>
     <clusternode name="node3.fenestros.loc" nodeid="3" votes="1"/>
   </clusternodes>
```

```
   <fencedevices>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.50" login="root" name="APC1" passwd="password"/>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.51" login="root" name="APC2" passwd="password"/>
   </fencedevices>
   <rm>
        <failoverdomains>
            <failoverdomain name="Failover1" nofailback="0" ordered="1" restricted="0">
                <failoverdomainnode name="node1.fenestros.loc" priority="1"/>
                <failoverdomainnode name="node2.fenestros.loc" priority="2"/>
            </failoverdomain>
        </failoverdomains>
        <resources>
            <ip address="10.0.3.100/24" monitor_link="yes" sleeptime="10"/>
        </resources>
        <service autostart="0" domain="Failover1" exclusive="0" name="ip_address" recovery="relocate">
            <ip address="10.0.3.100/24" monitor_link="yes" sleeptime="10"/>
        </service>
   </rm>
</cluster>
```

Validez le fichier sur le schéma du cluster (/usr/share/cluster/cluster.rng) :

```
[root@node1 ~]# ccs_config_validate
Configuration validates
```

Rappelez-vous qu'il existe quatre types de **Recovery Mode** :

Choix	**Description**
Relocate	En cas d'échec, le système essaie de démarrer le service sur un autre nœud.
Restart	En cas d'échec, le système essayera de re-démarrer le service sur le même nœud avant de démarrer le service sur un autre nœud.
Restart-Disable	En cas d'échec, le service est redémarré mais en cas d'échec du redémarrage le service est désactivé au lieu d'être transférer vers un autre nœud.

Choix	Description
Disable	En cas d'échec d'un composant d'un groupe de ressources, le système désactive le groupe de ressources.

LAB #26 - Propagation du Fichier de Configuration

Utilisez la commande **cman_tool version -r** pour propager le fichier de configuration :

```
[root@node1 ~]# cman_tool version -r
You have not authenticated to the ricci daemon on node1.fenestros.loc
Password: ricci
You have not authenticated to the ricci daemon on node2.fenestros.loc
Password: ricci
You have not authenticated to the ricci daemon on node3.fenestros.loc
Password: ricci
```

Arrêter et Démarrer le Cluster

Lors de l'arrêt du cluster, les services doivent être arrêtés dans un ordre précis :

- rgmanager - si vous utilisez les services high-availability,
- gfs2 - si vous utilisez Red Hat GFS2,
- clvmd - si CLVM a été utilisé pour créer des volumes clusterisés,
- cman.

Bien évidement lors du démarrage, les services doivent être démarrés dans l'ordre inverse :

- cman,
- clvmd - si CLVM a été utilisé pour créer des volumes clusterisés,
- gfs2 - si vous utilisez Red Hat GFS2,
- rgmanager - si vous utilisez les services high-availability.

LAB #27 - Supprimer un Noeud

Pour supprimer un noeud, éditez le fichier **/etc/cluster/cluster.conf** :

cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="7" name="moncluster">
   <cman two_node="1" expected_votes="1"/>
    <fence_daemon post_fail_delay="5" post_join_delay="25"/>
    <clusternodes>
        <clusternode name="node1.fenestros.loc" nodeid="1" votes="1">
            <fence>
                <method name="APC1">
                    <device name="APC1" port="1"/>
                </method>
            </fence>
        </clusternode>
        <clusternode name="node2.fenestros.loc" nodeid="2" votes="1"/>
    </clusternodes>
    <fencedevices>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.50" login="root" name="APC1" passwd="password"/>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.51" login="root" name="APC2" passwd="password"/>
    </fencedevices>
    <rm>
        <failoverdomains>
            <failoverdomain name="Failover1" nofailback="0" ordered="1" restricted="0">
                <failoverdomainnode name="node1.fenestros.loc" priority="1"/>
                <failoverdomainnode name="node2.fenestros.loc" priority="2"/>
            </failoverdomain>
        </failoverdomains>
        <resources>
            <ip address="10.0.3.100/24" monitor_link="yes" sleeptime="10"/>
```

```
        </resources>
        <service autostart="0" domain="Failover1" exclusive="0" name="ip_address" recovery="relocate">
            <ip address="10.0.3.100/24" monitor_link="yes" sleeptime="10"/>
        </service>
    </rm>
</cluster>
```

Validez le fichier sur le schéma du cluster (/usr/share/cluster/cluster.rng) :

```
[root@node1 ~]# ccs_config_validate
Configuration validates
```

Utilisez la commande **cman_tool version -r** pour propager le fichier de configuration :

```
[root@node1 ~]# cman_tool version -r
```

Utilisez la commande **clustat** pour visualiser le statut du cluster :

```
[root@node1 ~]# clustat
Cluster Status for moncluster @ Tue Dec 17 18:14:05 2013
Member Status: Quorate

 Member Name                                                     ID   Status
 ------ ----                                                     ---- ------
 node1.fenestros.loc                                                 1 Online, Local
 node2.fenestros.loc                                                 2 Online
 node3.fenestros.loc                                                 3 Online, Estranged
```

Éteignez node3 puis utilisez la commande clustat de nouveau :

```
[root@node1 ~]# clustat
Cluster Status for moncluster @ Tue Dec 17 18:15:03 2013
```

```
Member Status: Quorate

 Member Name                                                     ID   Status
 ------ ----                                                     ---- ------
 node1.fenestros.loc                                                 1 Online, Local
 node2.fenestros.loc                                                 2 Online
```

Utilisez maintenant la commande **cman_tool** :

```
[root@node1 ~]# cman_tool nodes
Node  Sts   Inc   Joined               Name
   1   M    136   2013-12-17 16:59:41  node1.fenestros.loc
   2   M    140   2013-12-17 16:59:42  node2.fenestros.loc
   3   X    144                        node3.fenestros.loc
```

La colonne **Sts** peut contenir une de trois valeurs :

- M Le noeud est un membre du cluster,
- X Le noeud n'est pas un membre du cluster,
- d Accès au cluster est interdit pour ce noeud.

```
Puisque le compte des neouds est passé de deux à trois, vous devez redémarrez les noeuds un et deux.
```

Une fois les machines redémarrées, la commande clustat démontre un quorum à deux noeuds :

```
[root@node1 ~]# clustat
Cluster Status for moncluster @ Tue Dec 17 18:30:37 2013
Member Status: Quorate

 Member Name                             ID   Status
 ------ ----                             ---- ------
 node1.fenestros.loc                         1 Online, Local
 node2.fenestros.loc                         2 Online
```

LAB #28 - Ajouter un Noeud

Pour rajouter node3 de nouveau, rétablissez le fichier **/etc/cluster/cluster.conf** dans son état d'origine :

cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="8" name="moncluster">
<fence_daemon post_fail_delay="5" post_join_delay="25"/>
   <clusternodes>
     <clusternode name="node1.fenestros.loc" nodeid="1" votes="1">
         <fence>
            <method name="APC1">
              <device name="APC1" port="1"/>
             </method>
         </fence>
     </clusternode>
     <clusternode name="node2.fenestros.loc" nodeid="2" votes="1"/>
     <clusternode name="node3.fenestros.loc" nodeid="3" votes="1"/>
   </clusternodes>
   <fencedevices>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.50" login="root" name="APC1" passwd="password"/>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.51" login="root" name="APC2" passwd="password"/>
   </fencedevices>
   <rm>
        <failoverdomains>
            <failoverdomain name="Failover1" nofailback="0" ordered="1" restricted="0">
                <failoverdomainnode name="node1.fenestros.loc" priority="1"/>
                <failoverdomainnode name="node2.fenestros.loc" priority="2"/>
            </failoverdomain>
        </failoverdomains>
        <resources>
            <ip address="10.0.3.100/24" monitor_link="yes" sleeptime="10"/>
```

```
        </resources>
        <service autostart="0" domain="Failover1" exclusive="0" name="ip_address" recovery="relocate">
            <ip address="10.0.3.100/24" monitor_link="yes" sleeptime="10"/>
        </service>
   </rm>
</cluster>
```

Démarrez node3 puis utilisez la commande **cman_tool version -r** pour propager le fichier de configuration :

```
[root@node1 ~]# cman_tool version -r
```

Démarrez maintenant le Cluster Service Manager :

```
[root@node1 ~]# service rgmanager start
Starting Cluster Service Manager:                          [  OK  ]
```

Activez rgmanager sur les trois neouds et redémarrez les machines virtuelles :

```
[root@node1 ~]# chkconfig --level 2345 rgmanager on
[root@node1 ~]# reboot
```

```
[root@node2 ~]# chkconfig --level 2345 rgmanager on
[root@node1 ~]# reboot
```

```
[root@node3 ~]# chkconfig --level 2345 rgmanager on
[root@node1 ~]# reboot
```

Une fois les machines redémarrées, utilisez la commande **clustat** pour visualiser le statut du cluster :

```
[root@node1 ~]# clustat
Cluster Status for moncluster @ Tue Dec 17 19:26:46 2013
Member Status: Quorate
```

```
Member Name                             ID   Status
------ ----                             ---- ------
node1.fenestros.loc                         1 Online, Local, rgmanager
node2.fenestros.loc                         2 Online, rgmanager
node3.fenestros.loc                         3 Online, rgmanager

Service Name                   Owner (Last)                   State
------- ----                   ----- ------                   -----
service:ip_address             node1.fenestros.loc            started
```

L'état du service peut être un des états suivants :

Etat	**Description**
Started	Le service est disponible.
Recovering	Le service est en attente de démarrage sur un autre nœud.
Disabled	Le service a été désactivé.
Stopped	Un état temporaire généralement visible juste avant la transition du service vers un autre noeud.
Failed	Le service est mort.
Uninitialized	Cet état peut apparaître dans certains cas lors du démarrage.

LAB #29 - Gérer les Services avec la Commande clusvcadm

La commande **clusvcadm** permet de :

- Activer un service,
- Désactiver un service,
- Arrêter un service,
- Geler un service,
- Dégeler un service,
- Migrer un service de machines virtuelle,
- Déplacer un service,
- Redémarrer un service.

Lancez la commande clustat :

```
[root@node1 ~]# clustat
Cluster Status for moncluster @ Tue Dec 17 21:57:24 2013
Member Status: Quorate

 Member Name                             ID   Status
 ------ ----                             ---- ------
 node1.fenestros.loc                         1 Online, Local, rgmanager
 node2.fenestros.loc                         2 Online, rgmanager
 node3.fenestros.loc                         3 Online, rgmanager

 Service Name                   Owner (Last)                   State
 ------- ----                   ----- ------                   -----
 service:ip_address             node1.fenestros.loc            started
```

Utilisez la commande clusvcadm pour arrêter le service ip_address :

```
[root@node1 ~]# clusvcadm -s ip_address
Local machine stopping service:ip_address...Success
[root@node1 ~]# clustat
Cluster Status for moncluster @ Tue Dec 17 22:00:11 2013
Member Status: Quorate

 Member Name                             ID   Status
 ------ ----                             ---- ------
 node1.fenestros.loc                         1 Online, Local, rgmanager
 node2.fenestros.loc                         2 Online, rgmanager
 node3.fenestros.loc                         3 Online, rgmanager

 Service Name                   Owner (Last)                   State
 ------- ----                   ----- ------                   -----
 service:ip_address             (node1.fenestros.loc)          stopped
```

Utilisez la commande clusvcadm pour démarrer le service ip_address :

```
[root@node1 ~]# clusvcadm -e ip_address
Local machine trying to enable service:ip_address...Success
service:ip_address is now running on node1.fenestros.loc
```

Utilisez la commande clusvcadm pour déplacer le service ip_address vers node2 :

```
[root@node1 ~]# clusvcadm -r ip_address
Trying to relocate service:ip_address...Success
service:ip_address is now running on node2.fenestros.loc
[root@node1 ~]# clustat
Cluster Status for moncluster @ Tue Dec 17 22:04:29 2013
Member Status: Quorate

 Member Name                             ID   Status
 ------ ----                             ---- ------
 node1.fenestros.loc                         1 Online, Local, rgmanager
 node2.fenestros.loc                         2 Online, rgmanager
 node3.fenestros.loc                         3 Online, rgmanager

 Service Name                   Owner (Last)                   State
 ------- ----                   ----- ------                   -----
 service:ip_address             node2.fenestros.loc            started
```

LAB #30 - CLVM et GFS2

La mutualisation du stockage s'obtient en utilisant un **SAN** (*Storage Area Network*) ou un **NAS** (*Network Attached Storage*). La différence entre les deux réside dans la façon que le client voit le stockage :

- le SAN est vu par le client comme un disque interne. Les ressources d'un SAN sont mutualisées permettant ainsi une gestion centralisée du stockage, une évolution du stockage ainsi que des fonctions de réplication,
- le NAS est partagé sur le réseau en utilisant un protocole tel **CIFS**, **NFS** et/ou **AppleShare**.

Le SAN utilise un réseau en fibre optique et un des protocoles suivants :

- **Fibre Channel**,
- **FCoE** ou *Fibre Channel over Ethernet*,
- **iSCSI**.

Le vocabulaire du SAN est résumé dans le tableau suivant :

Terme	**Description**
Fabric	L'ensemble d'un SAN
Director	Un switch fibre channel
GLM (*Global Link Module*)	Le raccordement entre le réseau fibre et le réseau ethernet dans le cas de l'utilisation du FCoE
HBA (*Host Bus Adaptors*)	Les adaptateurs fibre dans les clients
FA (*Fibre Adaptors*)	Les ports fibres d'un baie de stockage
Zoning	Le découpage du SAN en sous-réseaux, uniquement accessible à certains clients
mapping	La présentation de certains volumes logiques de la baie sur un ou plusieurs FA spécifiques
masking	La restriction d'accès à certains volumes logiques
multipathing	Le regroupement de plusieurs chemins vers le même SAN

Vous allez simuler un SAN avec iSCSI en utilisant une machine virtuelle. Arrêtez les machines virtuelles **node100**, **node200** et **node300**. Créez un clône de **node300**, appelé **san**. Attachez un disque supplémentaire, appelé **sdb** d'une taille de 512 Mo au controleur SATA de la nouvelle machine virtuelle **san**.

Démarrez la machine virtuelle **san**. Supprimez les services inutiles :

```
[root@node3 ~]# chkconfig --del iptables
[root@node3 ~]# chkconfig --del rgmanager
[root@node3 ~]# chkconfig --del ricci
[root@node3 ~]# chkconfig --del modclusterd
[root@node3 ~]# chkconfig --del clvmd
[root@node3 ~]# chkconfig --del cman
```

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

ifcfg-bond0

```
DEVICE=bond0
USERCTL=no
BOOTPROTO=none
ONBOOT=yes
IPADDR=10.0.3.18
NETMASK=255.255.255.0
NETWORK=10.0.3.0
BONDING_OPTS="miimon=100 mode=balance-xor"
TYPE=Unknown
IPV6INIT=no
```

Modifiez le fichier **/etc/sysconfig/network** :

network

```
NETWORKING=yes
HOSTNAME=san.fenestros.loc
```

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

hosts

```
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.0.3.15   node1.fenestros.loc     node1
10.0.3.16   node2.fenestros.loc    node2
10.0.3.17   node3.fenestros.loc    node3
```

```
10.0.3.18   san.fenestros.loc    san
```

Installez maintenant le paquet **scsi-target-utils** :

```
[root@node3 ~]# yum install scsi-target-utils
Loaded plugins: fastestmirror, refresh-packagekit, security
Loading mirror speeds from cached hostfile
 * atomic: mir01.syntis.net
 * base: mirrors.atosworldline.com
 * epel: epel.mirrors.ovh.net
 * extras: mirrors.atosworldline.com
 * rpmforge: repoforge.cu.be
 * updates: centos.mirror.crcrepairs.com
Setting up Install Process
Resolving Dependencies
--> Running transaction check
---> Package scsi-target-utils.i686 0:1.0.24-12.el6_5 will be installed
--> Processing Dependency: perl(Config::General) for package: scsi-target-utils-1.0.24-12.el6_5.i686
--> Running transaction check
---> Package perl-Config-General.noarch 0:2.52-1.el6 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

================================================================================================================
================================================
 Package                                        Arch                                Version
Repository                           Size
================================================================================================================
================================================
Installing:
 scsi-target-utils                              i686                                1.0.24-12.el6_5
updates                             172 k
```

```
Installing for dependencies:
 perl-Config-General                          noarch                          2.52-1.el6
base                                72 k

Transaction Summary
================================================================================================================
================================================================
Install       2 Package(s)

Total download size: 244 k
Installed size: 581 k
Is this ok [y/N]: y
```

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

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

```
[root@node3 ~]# tgtd --help
Usage: tgtd [OPTION]
Target framework daemon, version 1.0.24
  -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@node3 ~]# tgtadm --help
Usage: tgtadm [OPTION]
Linux SCSI Target Framework Administration Utility, version 1.0.24

  --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 specific
                        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> --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.
                        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>
                        delete the specific account having <name> from specific
                        target.
  --control-port <port> use control port <port>
  --help                display this help and exit

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

Configurez maintenant le service **tgtd** avec **chkconfig** :

```
[root@node3 ~]# chkconfig tgtd on
[root@node3 ~]# chkconfig --list tgtd
tgtd            0:arrêt    1:arrêt    2:marche    3:marche    4:marche    5:marche    6:arrêt
```

Important - Redémarrez la machine virtuelle **san**.

Créez maintenant une cible à laquelle vous pouvez rajouter le disque :

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

Consultez la configuration actuelle :

```
[root@san ~]# 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
            Backing store type: null
            Backing store path: None
            Backing store flags:
    Account information:
    ACL information:
```

Ajoutez maintenant le disque /dev/sdb au premier LUN :

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

Consultez la configuration :

```
[root@san ~]# 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
            Backing store type: null
            Backing store path: None
            Backing store flags:
        LUN: 1
            Type: disk
            SCSI ID: IET     00010001
            SCSI SN: beaf11
            Size: 537 MB, Block size: 512
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: No
            Backing store type: rdwr
            Backing store path: /dev/sdb
            Backing store flags:
    Account information:
    ACL information:
```

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

```
[root@san ~]# netstat -apn | grep 3260
tcp        0      0 0.0.0.0:3260                0.0.0.0:*                   LISTEN      1844/tgtd
tcp        0      0 :::3260                     :::*                        LISTEN      1844/tgtd
```

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

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

Constatez la modification de la section ACL :

```
[root@san ~]# 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
            Backing store type: null
            Backing store path: None
            Backing store flags:
        LUN: 1
            Type: disk
            SCSI ID: IET     00010001
            SCSI SN: beaf11
            Size: 537 MB, Block size: 512
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: No
            Backing store type: rdwr
            Backing store path: /dev/sdb
            Backing store flags:
    Account information:
```

```
    ACL information:
        ALL
```

Important - Notez la présence de la ligne **ACL information: ALL**.

Dernièrement configurez LUN1 en mode r/w :

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

Vérifiez votre configuration :

```
[root@san ~]# 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
            Backing store type: null
            Backing store path: None
            Backing store flags:
        LUN: 1
```

```
            Type: disk
            SCSI ID: IET     00010001
            SCSI SN: beaf11
            Size: 537 MB, Block size: 512
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: No
            Backing store type: rdwr
            Backing store path: /dev/sdb
            Backing store flags:
    Account information:
    ACL information:
        ALL
```

Important - Notez la présence de la ligne **Readonly: No**.

Pour rendre la configuration persistante re-créez le fichier **/etc/tgt/targets.conf** :

```
[root@san ~]# mv /etc/tgt/targets.conf /etc/tgt/targets.old
[root@san ~]# tgt-admin --dump > /etc/tgt/targets.conf
[root@san ~]# cat /etc/tgt/targets.conf
default-driver iscsi

<target target>
    backing-store /dev/sdb
</target>
```

Démarrez les machines virtuelles **node100**, **node200** et **node300**.

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

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

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

```
[root@node3 ~]# yum install iscsi-initiator-utils
```

Lancez maintenant la découverte des cibles sur le serveur iscsi **san** :

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

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

```
[root@node3 ~]# iscsiadm --mode discovery --type sendtargets --portal 10.0.3.18
Démarrage de iscsid :                                      [  OK  ]
10.0.3.18:3260,1 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.
```

```
[root@node3 ~]# 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.
```

Modifiez la directive **locking_type** dans le fichier **/etc/lvm/lvm.conf** sur chaque noeud :

```
...
    # Type of locking to use. Defaults to local file-based locking (1).
    # Turn locking off by setting to 0 (dangerous: risks metadata corruption
    # if LVM2 commands get run concurrently).
    # Type 2 uses the external shared library locking_library.
    # Type 3 uses built-in clustered locking.
    # Type 4 uses read-only locking which forbids any operations that might
    # change metadata.
    locking_type = 3
...
```

A partir de node1, créez un volume logique sur la cible iscsi :

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

[root@node1 ~]# vgcreate -cy vg0 /dev/sdb
  Clustered volume group "vg0" successfully created

[root@node1 ~]# lvcreate -L 240 -n lv0 vg0
  Logical volume "lv0" created
```

Vérifiez que le service clvmd sur chaque noeud voit le volume logique :

```
[root@node1 ~]# service clvmd status
clvmd (pid  4047) en cours d'exécution...
Clustered Volume Groups: vg0
Active clustered Logical Volumes: lv0
```

```
[root@node2 ~]# service clvmd status
clvmd (pid  1898) en cours d'exécution...
Clustered Volume Groups: vg0
Active clustered Logical Volumes: lv0

[root@node3 ~]# service clvmd status
clvmd (pid  2003) en cours d'exécution...
Clustered Volume Groups: vg0
Active clustered Logical Volumes: lv0
```

Vérifiez ensuite que la volume logique est visible à partir de tous les noeuds :

```
[root@node1 ~]# vgs
  VG   #PV #LV #SN Attr   VSize   VFree
  vg0    1   2   0 wz--nc 508,00m 256,00m
[root@node1 ~]# lvs
  LV   VG   Attr       LSize   Pool Origin Data%  Move Log Cpy%Sync Convert
  lv0  vg0  -wi-a----- 240,00m
[root@node1 ~]#

[root@node2 ~]# vgs
  VG   #PV #LV #SN Attr   VSize   VFree
  vg0    1   2   0 wz--nc 508,00m 256,00m
[root@node2 ~]# lvs
  LV   VG   Attr      LSize   Pool Origin Data%  Move Log Cpy%Sync Convert
  lv0  vg0  -wi-a---- 240,00m
[root@node2 ~]#

[root@node3 ~]# vgs
  VG   #PV #LV #SN Attr   VSize   VFree
  vg0    1   2   0 wz--nc 508,00m 256,00m
[root@node3 ~]# lvs
  LV   VG   Attr      LSize   Pool Origin Data%  Move Log Cpy%Sync Convert
  lv0  vg0  -wi-a---- 240,00m
```

```
[root@node3 ~]#
```

Important - Notez l'attribut **c** dans la sortie de la commande **vgs**. Ceci implique que la vg est clusterisé.

Notez que lors de la création du système de fichiers gfs2 en utilisant les paramètres par défaut celle-ci renvoie une erreur **Not enough space available on device**. Cette erreur est causée par le fait que la taille par défaut des trois journaux est supérieur à la taille de **pseudo-iscsi** :

```
[root@node1 ~]# mkfs.gfs2 -p lock_dlm -t moncluster:mydata -j 3 /dev/mapper/vg0-lv0
This will destroy any data on /dev/mapper/vg0-lv0.
It appears to contain: symbolic link to `../dm-0'

Are you sure you want to proceed? [y/n] y

Not enough space available on device
```

Créez donc un système de fichiers gfs2 ayant 3 journaux de 32Mo chacun :

```
[root@node1 ~]# mkfs.gfs2 -p lock_dlm -t moncluster:mydata -j 3 -J 32 /dev/vg0/lv0
This will destroy any data on /dev/vg0/lv0.
It appears to contain: symbolic link to `../dm-0'

Are you sure you want to proceed? [y/n] y

Device:                    /dev/vg0/lv0
Blocksize:                 4096
Device Size                0,23 GB (61440 blocks)
Filesystem Size:           0,23 GB (61437 blocks)
Journals:                  3
Resource Groups:           1
Locking Protocol:          "lock_dlm"
Lock Table:                "moncluster:mydata"
```

```
UUID:                   c0f648d8-66f3-c1bc-d7b6-953693931de8
```

Montez ensuite le volume logique dans chaque noeud :

```
[root@node1 ~]# mkdir /mydata
[root@node1 ~]# mount -t gfs2 -o acl,noatime /dev/mapper/vg0-lv0 /mydata

[root@node2 ~]# mkdir /mydata
[root@node2 ~]# mount -t gfs2 -o acl,noatime /dev/mapper/vg0-lv0 /mydata

[root@node3 ~]# mkdir /mydata
[root@node3 ~]# mount -t gfs2 -o acl,noatime /dev/mapper/vg0-lv0 /mydata
```

Testez maintenant le système de fichiers gfs2 en créant un fichier à partir de node01 :

```
[root@node1 ~]# touch /mydata/node1_test
```

Visualisez ce fichier à partir de node2 et node3 :

```
[root@node2 ~]# ls /mydata
node1_test

[root@node3 ~]# ls /mydata
node1_test
```

Lancez la commande suivante sur node1 :

```
[root@node1 ~]# vi /mydata/node1_test
```

Lancez la même commande sur node2 :

```
[root@node2 ~]# vi /mydata/node1_test
```

Notez que vous obtenez un message similaire au massage suivant sur node2 :

```
E325: ATTENTION
Found a swap file by the name "/mydata/.node1_test.swp"
          owned by: root   dated: Sun May 25 10:11:38 2014
         file name: /mydata/node1_test
          modified: no
         user name: root   host name: node1.fenestros.loc
        process ID: 4587
While opening file "/mydata/node1_test"
             dated: Sun May 25 10:11:05 2014

(1) Another program may be editing the same file.
    If this is the case, be careful not to end up with two
    different instances of the same file when making changes.
    Quit, or continue with caution.

(2) An edit session for this file crashed.
    If this is the case, use ":recover" or "vim -r /mydata/node1_test"
    to recover the changes (see ":help recovery").
    If you did this already, delete the swap file "/mydata/.node1_test.swp"
    to avoid this message.
"/mydata/node1_test" 0L, 0C
Press ENTER or type command to continue
```

Activez le service gfs2 :

```
[root@node1 ~]# chkconfig --levels 345 gfs2 on
[root@node1 ~]# chkconfig | grep gfs2
gfs2            0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche    6:arrêt

[root@node2 ~]# chkconfig --levels 345 gfs2 on
[root@node2 ~]# chkconfig | grep gfs2
gfs2            0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche    6:arrêt
```

```
[root@node3 ~]# chkconfig --levels 345 gfs2 on
[root@node3 ~]# chkconfig | grep gfs2
gfs2            0:arrêt    1:arrêt    2:arrêt    3:marche    4:marche    5:marche    6:arrêt
```

Dernièrement, modifiez le fichier **/etc/fstab** sur chaque noeud :

/etc/fstab

```
#
# /etc/fstab
# Created by anaconda on Fri May  3 13:33:42 2013
#
# Accessible filesystems, by reference, are maintained under '/dev/disk'
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info
#
UUID=b9f29672-c84e-4d3b-b132-189758a084eb /                       ext4    defaults        1 1
UUID=01baf03d-df0d-479b-b3e4-81ce63b8dec3 /boot                   ext4    defaults        1 2
UUID=2646a33a-65f3-4501-9ced-9459435fd774 swap                    swap    defaults        0 0
tmpfs                   /dev/shm                tmpfs   defaults        0 0
devpts                  /dev/pts                devpts  gid=5,mode=620  0 0
sysfs                   /sys                    sysfs   defaults        0 0
/dev/mapper/vg0-lv0     /mydata                 gfs2    noatime,acl     0 0
proc                    /proc                   proc    defaults        0 0
```

LAB #31 - Utilisation d'un Disque Quorum

Arrêtez la machine virtuelle **san**. Attachez un disque supplémentaire appelé **sdc** d'une taille de 11 Mo au controleur SATA de la nouvelle machine virtuelle **san**. Démarrez la machine virtuelle **san**.

Dans la machine virtuelle **san** ajoutez le disque **/dev/sdc** au deuxième LUN :

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

Consultez la configuration :

```
[root@san ~]# 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
            Backing store type: null
            Backing store path: None
            Backing store flags:
        LUN: 1
            Type: disk
            SCSI ID: IET     00010001
            SCSI SN: beaf11
            Size: 537 MB, Block size: 512
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: 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: 12 MB, Block size: 512
            Online: Yes
            Removable media: No
            Prevent removal: No
            Readonly: No
            Backing store type: rdwr
            Backing store path: /dev/sdc
            Backing store flags:
    Account information:
    ACL information:
        ALL
```

Vérifiez que le cluster est en état de fonctionnement sur les noeuds :

```
[root@node1 ~]# clustat
Cluster Status for moncluster @ Mon May 26 11:50:24 2014
Member Status: Quorate

 Member Name                                                     ID   Status
 ------ ----                                                     ---- ------
 node1.fenestros.loc                                                 1 Online, Local, rgmanager
 node2.fenestros.loc                                                 2 Online, rgmanager
 node3.fenestros.loc                                                 3 Online, rgmanager

 Service Name                                                     Owner (Last)
State
 ------- ----                                                     ----- ------
```

```
-----
 service:ip_address                                              (none)
disabled
```

Déconnectez-vous et re-connectez-vous au SAN à partir de chaque noeud :

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

```
[root@node2 ~]# iscsiadm --mode node --targetname target --logout
[root@node2 ~]# iscsiadm --mode node --targetname target --login
```

```
[root@node3 ~]# iscsiadm --mode node --targetname target --logout
[root@node3 ~]# iscsiadm --mode node --targetname target --login
```

Vérifiez la présence du disque sdc sur chaque noeud :

```
[root@node1 ~]# cat /proc/partitions
major minor  #blocks  name

   8        0   20971520 sda
   8        1     102400 sda1
   8        2    5120000 sda2
   8        3    2048000 sda3
   8       16     524288 sdb
   8       32      11264 sdc
 253        0     245760 dm-0
```

```
[root@node2 ~]# cat /proc/partitions
major minor  #blocks  name

   8        0   20971520 sda
   8        1     102400 sda1
   8        2    5120000 sda2
```

```
   8        3    2048000 sda3
   8       16     524288 sdb
   8       32      11264 sdc
 253        0     245760 dm-0
```

```
[root@node3 ~]# cat /proc/partitions
major minor  #blocks  name

   8        0   20971520 sda
   8        1     102400 sda1
   8        2    5120000 sda2
   8        3    2048000 sda3
   8       16     524288 sdb
   8       32      11264 sdc
 253        0     245760 dm-0
```

Préparez maintenant le Disque Quorum sur **/dev/sdc** grâce à la commande **mkqdisk** :

```
[root@node1 ~]# mkqdisk -c /dev/sdc -l qdisk
mkqdisk v3.0.12.1

Writing new quorum disk label 'qdisk' to /dev/sdc.
WARNING: About to destroy all data on /dev/sdc; proceed [N/y] ? y
Initializing status block for node 1...
Initializing status block for node 2...
Initializing status block for node 3...
Initializing status block for node 4...
Initializing status block for node 5...
Initializing status block for node 6...
Initializing status block for node 7...
Initializing status block for node 8...
Initializing status block for node 9...
Initializing status block for node 10...
Initializing status block for node 11...
```

```
Initializing status block for node 12...
Initializing status block for node 13...
Initializing status block for node 14...
Initializing status block for node 15...
Initializing status block for node 16...
```

Important - Notez la présence de **status blocks** pour les 16 neouds. L'option **-l** indique le **label** (étiquette) associé au disque.

Il n'y a pas de filesystem classique sur le Disque Quorum :

```
[root@node1 ~]# file -sL /dev/sda
/dev/sda: x86 boot sector; GRand Unified Bootloader, stage1 version 0x3, boot drive 0x80, 1st sector stage2
0x76ea, GRUB version 0.94; partition 1: ID=0x83, active, starthead 32, startsector 2048, 204800 sectors;
partition 2: ID=0x83, starthead 223, startsector 206848, 10240000 sectors; partition 3: ID=0x82, starthead 72,
startsector 10446848, 4096000 sectors, code offset 0x48
[root@node1 ~]# file -sL /dev/sda1
/dev/sda1: Linux rev 1.0 ext4 filesystem data (needs journal recovery) (extents) (huge files)
[root@node1 ~]# file -sL /dev/sdc
/dev/sdc: data
[root@node1 ~]# file -sL /dev/sdc1
/dev/sdc1: cannot open `/dev/sdc1' (No such file or directory)
```

Visualisez le Disque Quorum sur les trois noeuds :

```
[root@node1 ~]# mkqdisk -L
mkqdisk v3.0.12.1

/dev/block/8:32:
/dev/disk/by-id/scsi-1IET_00010002:
/dev/disk/by-path/ip-10.0.3.18:3260-iscsi-target-lun-2:
/dev/sdc:
```

```
    Magic:                eb7a62c2
    Label:                qdisk
    Created:              Mon May 26 11:54:21 2014
    Host:                 node1.fenestros.loc
    Kernel Sector Size:   512
    Recorded Sector Size: 512
```

```
[root@node2 ~]# mkqdisk -L
mkqdisk v3.0.12.1

/dev/block/8:32:
/dev/disk/by-id/scsi-1IET_00010002:
/dev/disk/by-path/ip-10.0.3.18:3260-iscsi-target-lun-2:
/dev/sdc:
    Magic:                eb7a62c2
    Label:                qdisk
    Created:              Mon May 26 11:54:21 2014
    Host:                 node1.fenestros.loc
    Kernel Sector Size:   512
    Recorded Sector Size: 512
```

```
[root@node3 ~]# mkqdisk -L
mkqdisk v3.0.12.1

/dev/block/8:32:
/dev/disk/by-id/scsi-1IET_00010002:
/dev/disk/by-path/ip-10.0.3.18:3260-iscsi-target-lun-2:
/dev/sdc:
    Magic:                eb7a62c2
    Label:                qdisk
    Created:              Mon May 26 11:54:21 2014
    Host:                 node1.fenestros.loc
    Kernel Sector Size:   512
```

```
    Recorded Sector Size: 512
```

> Il est déconseillé d'utiliser un volume logique clusterisé pour le Disque Quorum. La raison est simple : pour qur le quorum soit effectif, le Disque Quorum doit être visible. Or pour que le Disque Quorum soit visible, le service clvmd doit démarrer. Sachant que le service clvmd dépende du service cman ce dernier s'arrête.

Modifiez maintenant le fichier **/etc/cluster/cluster.conf** sur node1.fenestros.loc en ajoutant les deux lignes suivantes :

```
...
        <totem token="135000"/>
    <cman expected_votes="5"/>
        <quorumd interval="3" label="qdisk" tko="23" votes="2"/>
...
```

La première ligne indique le timeout pour cman est de 135 seconds :

- soit 1,5 fois le timeout de qdisk,
- ceci implique que si le qdisk défaillant se trouve sur le maître, dans ce cas node1.fenestros.loc, cman donnera une instruction au démon fenced de redémarrer le noeud.

La deuxième ligne porte à la connaissance de cman le nombre de votes attendues, soit 5 - 1 pour chaque noeud et 2 pour le Disque Quorum.

La troisième ligne configure le Disque Quorum :

- **interval** indique que qdiskd va vérifier tous les **3** secondes,
- **tko** (*Technical Knock Out*) indique que qdiskd permettra **23** tentatives en échec avant de réagir, soit 3*23=69 secondes,
- **label** identifie le Disque Quorum,
- **votes** fixe le nombre de votes pour le Disque Quorum lui-même.

> **Important** - Dans cette configuration, si le service qdiskd sur un noeud n'a pas mis à jour ses informations d'état à l'issu de 69 secondes, un autre qdiskd va demander qu'il soit fenced.

Vous obtiendrez donc :

/etc/cluster/cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="12" name="moncluster">
        <totem token="135000"/>
    <cman expected_votes="5"/>
        <quorumd interval="3" label="qdisk" tko="23" votes="2"/>
    <fence_daemon post_fail_delay="5" post_join_delay="25"/>
    <clusternodes>
        <clusternode name="node1.fenestros.loc" nodeid="1" votes="1">
            <fence>
                <method name="APC1">
                    <device name="APC1" port="1"/>
                </method>
            </fence>
        </clusternode>
        <clusternode name="node2.fenestros.loc" nodeid="2" votes="1"/>
        <clusternode name="node3.fenestros.loc" nodeid="3" votes="1"/>
    </clusternodes>
    <fencedevices>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.50" login="root" name="APC1" passwd="password"/>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.51" login="root" name="APC2" passwd="password"/>
    </fencedevices>
    <rm>
        <failoverdomains>
            <failoverdomain name="Failover1" nofailback="0" ordered="1" restricted="0">
                <failoverdomainnode name="node1.fenestros.loc" priority="1"/>
                <failoverdomainnode name="node2.fenestros.loc" priority="2"/>
            </failoverdomain>
        </failoverdomains>
        <resources>
            <ip address="10.0.3.100/24" monitor_link="yes" sleeptime="10"/>
```

```
        </resources>
        <service autostart="0" domain="Failover1" exclusive="0" name="ip_address" recovery="relocate">
            <ip address="10.0.3.100/24" monitor_link="yes" sleeptime="10"/>
        </service>
    </rm>
</cluster>
```

Validez le fichier sur le schéma du cluster (/usr/share/cluster/cluster.rng) :

```
[root@node1 ~]# ccs_config_validate
Configuration validates
```

Utilisez la commande **cman_tool version -r** pour propager le fichier de configuration :

```
[root@node1 ~]# cman_tool version -r
```

Constatez maintenant l'état du cluster :

```
[root@node1 ~]# cman_tool status
Version: 6.2.0
Config Version: 12
Cluster Name: moncluster
Cluster Id: 59006
Cluster Member: Yes
Cluster Generation: 688
Membership state: Cluster-Member
Nodes: 3
Expected votes: 3
Total votes: 3
Node votes: 1
Quorum: 2
Active subsystems: 9
Flags:
```

```
Ports Bound: 0 11 177
Node name: node1.fenestros.loc
Node ID: 1
Multicast addresses: 239.192.230.101
Node addresses: 10.0.3.15
```

Créez le script **/root/cluster.sh** sur chaque noeud :

/root/cluster.sh

```
#!/bin/sh
for i in ricci rgmanager gfs2 clvmd cman ; do
      service $i stop
done
for i in cman clvmd gfs2 rgmanager ricci ; do
      service $i start
done
```

```
[root@node1 ~]# vi /root/cluster.sh
[root@node1 ~]# chmod u+x /root/cluster.sh

[root@node2 ~]# vi /root/cluster.sh
[root@node2 ~]# chmod u+x /root/cluster.sh

[root@node3 ~]# vi /root/cluster.sh
[root@node3 ~]# chmod u+x /root/cluster.sh
```

Exécutez le script sur chaque neoud **en même temps** :

<coe> [root@node1 ~]# ./cluster.sh

[root@node2 ~]# ./cluster.sh

[root@node3 ~]# ./cluster.sh </code>

Sur chaque noeud, vous devez voir une sortie similaire à celle-ci :

```
[root@node1 ~]# ./cluster.sh
Shutting down ricci:                                       [  OK  ]
Stopping Cluster Service Manager:                          [  OK  ]
Unmounting GFS2 filesystem (/mydata):                      [  OK  ]
Deactivating clustered VG(s):   0 logical volume(s) in volume group "vg0" now active
  clvmd not running on node node3.fenestros.loc
                                                           [  OK  ]
Signaling clvmd to exit                                    [  OK  ]
clvmd terminated                                           [  OK  ]
Stopping cluster:
   Leaving fence domain...                                 [  OK  ]
   Stopping gfs_controld...                                [  OK  ]
   Stopping dlm_controld...                                [  OK  ]
   Stopping fenced...                                      [  OK  ]
   Stopping qdiskd...                                      [  OK  ]
   Stopping cman...                                        [  OK  ]
   Waiting for corosync to shutdown:                       [  OK  ]
   Unloading kernel modules...                             [  OK  ]
   Unmounting configfs...                                  [  OK  ]
Starting cluster:
   Checking if cluster has been disabled at boot...        [  OK  ]
   Checking Network Manager...                             [  OK  ]
   Global setup...                                         [  OK  ]
   Loading kernel modules...                               [  OK  ]
   Mounting configfs...                                    [  OK  ]
   Starting cman...                                        [  OK  ]
   Starting qdiskd...                                      [  OK  ]
   Waiting for quorum...                                   [  OK  ]
   Starting fenced...                                      [  OK  ]
   Starting dlm_controld...                                [  OK  ]
```

```
   Tuning DLM kernel config...                                [  OK  ]
   Starting gfs_controld...                                   [  OK  ]
   Unfencing self...                                          [  OK  ]
   Joining fence domain...                                    [  OK  ]
Starting clvmd:
Activating VG(s):   1 logical volume(s) in volume group "vg0" now active
                                                              [  OK  ]
Mounting GFS2 filesystem (/mydata):                           [  OK  ]
Starting Cluster Service Manager:                             [  OK  ]
Démarrage de ricci :                                          [  OK  ]
```

```
[root@node2 ~]# ./cluster.sh
Shutting down ricci:                                          [  OK  ]
Stopping Cluster Service Manager:                             [  OK  ]
Unmounting GFS2 filesystem (/mydata):                         [  OK  ]
Deactivating clustered VG(s):   0 logical volume(s) in volume group "vg0" now active
  clvmd not running on node node3.fenestros.loc
                                                              [  OK  ]
Signaling clvmd to exit                                       [  OK  ]
clvmd terminated                                              [  OK  ]
Stopping cluster:
   Leaving fence domain...                                    [  OK  ]
   Stopping gfs_controld...                                   [  OK  ]
   Stopping dlm_controld...                                   [  OK  ]
   Stopping fenced...                                         [  OK  ]
   Stopping qdiskd...                                         [  OK  ]
   Stopping cman...                                           [  OK  ]
   Waiting for corosync to shutdown:                          [  OK  ]
   Unloading kernel modules...                                [  OK  ]
   Unmounting configfs...                                     [  OK  ]
Starting cluster:
   Checking if cluster has been disabled at boot...           [  OK  ]
   Checking Network Manager...                                [  OK  ]
   Global setup...                                            [  OK  ]
```

```
   Loading kernel modules...                                  [  OK  ]
   Mounting configfs...                                       [  OK  ]
   Starting cman...                                           [  OK  ]
   Starting qdiskd...                                         [  OK  ]
   Waiting for quorum...                                      [  OK  ]
   Starting fenced...                                         [  OK  ]
   Starting dlm_controld...                                   [  OK  ]
   Tuning DLM kernel config...                                [  OK  ]
   Starting gfs_controld...                                   [  OK  ]
   Unfencing self...                                          [  OK  ]
   Joining fence domain...                                    [  OK  ]
Starting clvmd:
Activating VG(s):   1 logical volume(s) in volume group "vg0" now active
                                                              [  OK  ]
Mounting GFS2 filesystem (/mydata):                           [  OK  ]
Starting Cluster Service Manager:                             [  OK  ]
Démarrage de ricci :                                          [  OK  ]
```

```
[root@node3 ~]# ./cluster.sh
Shutting down ricci:                                          [  OK  ]
Stopping Cluster Service Manager:                             [  OK  ]
Unmounting GFS2 filesystem (/mydata):                         [  OK  ]
Deactivating clustered VG(s):   0 logical volume(s) in volume group "vg0" now active
                                                              [  OK  ]
Signaling clvmd to exit                                       [  OK  ]
clvmd terminated                                              [  OK  ]
Stopping cluster:
   Leaving fence domain...                                    [  OK  ]
   Stopping gfs_controld...                                   [  OK  ]
   Stopping dlm_controld...                                   [  OK  ]
   Stopping fenced...                                         [  OK  ]
   Stopping qdiskd...                                         [  OK  ]
   Stopping cman...                                           [  OK  ]
   Waiting for corosync to shutdown:                          [  OK  ]
```

```
   Unloading kernel modules...                                [  OK  ]
   Unmounting configfs...                                     [  OK  ]
Starting cluster:
   Checking if cluster has been disabled at boot...           [  OK  ]
   Checking Network Manager...                                [  OK  ]
   Global setup...                                            [  OK  ]
   Loading kernel modules...                                  [  OK  ]
   Mounting configfs...                                       [  OK  ]
   Starting cman...                                           [  OK  ]
   Starting qdiskd...                                         [  OK  ]
   Waiting for quorum...                                      [  OK  ]
   Starting fenced...                                         [  OK  ]
   Starting dlm_controld...                                   [  OK  ]
   Tuning DLM kernel config...                                [  OK  ]
   Starting gfs_controld...                                   [  OK  ]
   Unfencing self...                                          [  OK  ]
   Joining fence domain...                                    [  OK  ]
Starting clvmd:
Activating VG(s):   1 logical volume(s) in volume group "vg0" now active
                                                              [  OK  ]
Mounting GFS2 filesystem (/mydata):                           [  OK  ]
Starting Cluster Service Manager:                             [  OK  ]
Démarrage de ricci :                                          [  OK  ]
```

Constatez maintenant l'état du cluster :

```
[root@node1 ~]# cman_tool status
Version: 6.2.0
Config Version: 12
Cluster Name: moncluster
Cluster Id: 59006
Cluster Member: Yes
Cluster Generation: 752
Membership state: Cluster-Member
```

```
Nodes: 3
Expected votes: 5
Quorum device votes: 2
Total votes: 5
Node votes: 1
Quorum: 3
Active subsystems: 11
Flags:
Ports Bound: 0 11 177 178
Node name: node1.fenestros.loc
Node ID: 1
Multicast addresses: 239.192.230.101
Node addresses: 10.0.3.15
```

> **Important** - Notez la présence de la ligne **Quorum device votes: 2**.

Remettez en place le service ip_address et basculez-le sur node2.fenestros.loc :

```
[root@node1 ~]# clusvcadm -e ip_address
```

```
[root@node1 ~]# clusvcadm -r ip_address
```

Mettez en place maintenant une heuristique.

> **Important** - Une heuristique est une règle qui définisse des conditions dans lesquelles le nœud doit être considéré comme n'appartenant plus au cluster.

Dans notre cas nous allons mettre en place une heuristique qui vérifie la connectivité du noeud au réseau. Cette règle prend la forme d'un ping à la

passerelle.

Pour faire ceci, il convient de changer la ligne :

```
...
        <quorumd interval="3" label="qdisk" tko="23" votes="2"/>
...
```

en

```
...
    <quorumd interval="3" label="qdisk" tko="23" votes="2">
        <heuristic program="ping -c3 -t2 10.0.2.2"/>
    </quorumd>
...
```

Editez donc votre fichier **/etc/cluster/cluster.conf** :

/etc/cluster/cluster.conf

```
<?xml version="1.0"?>
<cluster config_version="13" name="moncluster">
    <totem token="135000"/>
    <cman expected_votes="5"/>
    <quorumd interval="3" label="qdisk" tko="23" votes="2">
        <heuristic program="ping -c3 -t2 10.0.2.2"/>
    </quorumd>
    <fence_daemon post_fail_delay="5" post_join_delay="25"/>
    <clusternodes>
        <clusternode name="node1.fenestros.loc" nodeid="1">
            <fence>
                <method name="APC1">
                    <device name="APC1" port="1"/>
                </method>
```

```
            </fence>
        </clusternode>
        <clusternode name="node2.fenestros.loc" nodeid="2"/>
        <clusternode name="node3.fenestros.loc" nodeid="3"/>
    </clusternodes>
    <fencedevices>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.50" login="root" name="APC1" passwd="password"/>
        <fencedevice agent="fence_apc" ipaddr="10.0.3.51" login="root" name="APC2" passwd="password"/>
    </fencedevices>
    <rm>
        <failoverdomains>
            <failoverdomain name="Failover1" ordered="1">
                <failoverdomainnode name="node1.fenestros.loc" priority="1"/>
                <failoverdomainnode name="node2.fenestros.loc" priority="2"/>
            </failoverdomain>
        </failoverdomains>
        <resources>
            <ip address="10.0.3.100/24" monitor_link="yes" sleeptime="10"/>
        </resources>
        <service autostart="0" domain="Failover1" name="ip_address" recovery="relocate">
            <ip address="10.0.3.100/24" monitor_link="yes" sleeptime="10"/>
        </service>
    </rm>
</cluster>
```

Important - Cette modification peut également être faite en utilisant **luci** via les onglets **Configure** puis **qdisk**.

Propagez votre fichier de configuration aux autres neouds et ré-exécutez le script **/root/cluster.sh** sur chaque neoud.

Sur la machine virtuelle **node2.fenestros.loc**, arrêtez le réseau :

```
[root@node2 ~]# service network stop
Arrêt de l'interface bond0 :                               [  OK  ]
Arrêt de l'interface eth0 :                                [  OK  ]
Arrêt de l'interface loopback :                            [  OK  ]
```

Consultez le journal **/var/log/messages** sur la machine virtuelle **node1.fenestros.loc** :

```
...
May 26 15:10:49 node1 qdiskd[15144]: Writing eviction notice for node 2
May 26 15:10:52 node1 qdiskd[15144]: Node 2 evicted
May 26 15:14:00 node1 corosync[15094]:   [QUORUM] Members[2]: 1 3
May 26 15:14:00 node1 rgmanager[15608]: State change: node2.fenestros.loc DOWN
May 26 15:14:00 node1 corosync[15094]:   [TOTEM ] A processor joined or left the membership and a new membership
was formed.
May 26 15:14:00 node1 corosync[15094]:   [CPG   ] chosen downlist: sender r(0) ip(10.0.3.15) ; members(old:3
left:1)
May 26 15:14:00 node1 corosync[15094]:   [MAIN  ] Completed service synchronization, ready to provide service.
May 26 15:14:01 node1 kernel: dlm: closing connection to node 2
May 26 15:14:01 node1 rgmanager[15608]: Taking over service service:ip_address from down member
node2.fenestros.loc
May 26 15:14:01 node1 kernel: GFS2: fsid=moncluster:mydata.1: jid=0: Trying to acquire journal lock...
May 26 15:14:06 node1 fenced[15360]: fencing node node2.fenestros.loc
May 26 15:14:06 node1 fenced[15360]: fence node2.fenestros.loc dev 0.0 agent none result: error no method
May 26 15:14:06 node1 fenced[15360]: fence node2.fenestros.loc failed
May 26 15:14:09 node1 fenced[15360]: fencing node node2.fenestros.loc
May 26 15:14:09 node1 fenced[15360]: fence node2.fenestros.loc dev 0.0 agent none result: error no method
May 26 15:14:09 node1 fenced[15360]: fence node2.fenestros.loc failed
May 26 15:14:12 node1 fenced[15360]: fencing node node2.fenestros.loc
May 26 15:14:12 node1 fenced[15360]: fence node2.fenestros.loc dev 0.0 agent none result: error no method
May 26 15:14:12 node1 fenced[15360]: fence node2.fenestros.loc failed
...
```

Important - Notez que qdiskd enlève node2.fenestros.loc du cluster. Le service rgmanager prend possession du service ip_address. Constatez la tentative de fencing qui n'aboutit pas parce que le périphérique fence n'existe pas réellement. Notez donc que le service ip_address ne bascule pas sur node1.fenestros.loc car un pré-requis du basculement est que node2.fenestros.loc soit fenced.

Arrêtez maintenant le service réseau sur **node3.fenestros.loc** :

```
[root@node3 ~]# service network stop
Arrêt de l'interface bond0 :                               [  OK  ]
Arrêt de l'interface eth0 :                                [  OK  ]
Arrêt de l'interface loopback :                            [  OK  ]
```

Consultez ensuite l'état du cluster à partir du **node1.fenestros.loc** :

```
[root@node1 ~]# clustat
Service states unavailable: Temporary failure; try again
Cluster Status for moncluster @ Mon May 26 15:31:52 2014
Member Status: Quorate

 Member Name                                                     ID   Status
 ------ ----                                                     ---- ------
 node1.fenestros.loc                                                 1 Online, Local
 node2.fenestros.loc                                                 2 Offline
 node3.fenestros.loc                                                 3 Offline
 /dev/block/8:32                                                     0 Online, Quorum Disk
```

Important - Notez que le cluster est toujours actif grâce au Disque Quorum.

Red Hat High Availability Cluster sous CentOS 7

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 machine virtuelle **CentOS**, créez 2 clones complets et configurez-les ainsi :

Nom de la VM	**RAM**
node1.i2tch.loc	512 Mo
node2.i2tch.loc	512 Mo

Important - Lors de la création des clones, veillez à réinitialiser l'adresse MAC de la carte réseau.

Modifiez la configuration réseau des deux clones :

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

Important - Dans Virtual Box > Paramètres de node2.i2tch.loc > Réseau > Carte 1 > Redirection de ports, Modifiez le port hôte ssh en 4022.

Démarrez les machines virtuelles **node1.i2tch.loc** et **node2.i2tch.loc** et modifiez les noms d'hôtes ainsi :

```
[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 noeud :

```
[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
```

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
```

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 #32 - 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 #33 - 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 #34 - 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 #35 - 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 #36 - Mise en place d'un Serveur Apache Actif/Passif

Création du Stockage Partagé

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

Ajoutez un disque supplémentaire de type **vdi** et d'une taille de **8Go** au contrôleur SATA.

Démarrez la machine virtuelle.

Connectez-vous à la VM via putty sur localhost:6022.

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

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

```
    SetHandler server-status
    Require local
</Location>
```

```
[root@node2 ~]# vi /etc/httpd/conf/httod.conf
[root@node2 ~]# cat /etc/httpd/conf/httod.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@node1 ~]# vi /etc/lvm/lvm.conf
[root@node1 ~]# cat /etc/lvm/lvm.conf | grep my_vg
    volume_list = [ ]
<code>

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

<code>
[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**.
