→ Convenciones:

```
# En todos los nodos como 'sudo su'.
[root@srv1 ~]# Solo en servidor 'srv1'→ como 'sudo su'.
[root@srv2 ~]# Solo en servidor 'srv2'→ como 'sudo su'.
```

362.3 Clustered File Systems (weight: 4)

Weight	4
Description	Candidates should be able to install, maintain and troubleshoot GFS2 and OCFS2 filesystems. This includes awareness of other clustered filesystems available on Linux.

Key Knowledge Areas:

- Understand the principles of cluster file systems and distributed file systems
- Understand the Distributed Lock Manager
- Create, maintain and troubleshoot GFS2 file systems in a cluster
- Create, maintain and troubleshoot OCFS2 file systems in a cluster
- Awareness of the O2CB cluster stack
- Awareness of other commonly used clustered file systems, such as AFS and Lustre

Partial list of the used files, terms and utilities:

- mkfs.gfs2
- mount.gfs2
- fsck.gfs2
- gfs2_grow
- gfs2_edit
- gfs2_jadd
- mkfs.ocfs2
- mount.ocfs2
- fsck.ocfs2
- tunefs.ocfs2
- mounted.ocfs2
- o2info
- o2image

→ → Conceptos teóricos preliminares ==> GFS2

Conceptos ⇒ GFS2

Sistema de archivos de Cluster simétrico de 64 bits. El módulo del kernel **gfs2.ko** implementa el Stma. de archivos GFS2 y se carga en los nodos del cluster. Para mantener la coherencia entre los nodos del cluster, el control de la caché lo proporciona la máquina de estado ⇒ **glock.**

Conceptos ⇒ glock

Estructura de datos que contiene el DLM y la caché en una sola máquina de estado. Cada glock tiene una relación 1:1 con un único bloqueo DLM y proporciona almacenamiento en caché de este estado de bloqueo para que las operaciones repetitivas realizadas desde un único nodo del stma. no tengan que llenar repetitivamente el DLM evitando un tráfico de red innecesario.

⇒ **2 categorias de glocks:**

-> Los que almacenan en caché los metadatos.

-> Los que NO los almacenan.

Los glocks de modo y grupo de recursos almacenan en caché los metadatos. Los otros tipos de glocks NO. El **ínodo-glock** tambien participa en el almacenamiento en caché de los datos, ademas de los metadatos, y contiene la lógica mas compleja de todos los glocks.

Componentes GFS2

⇒ Nodos GFS2.

⇒ Número de Stmas. de Archivos.

⇒ Nombre del Sistema de Archivos.

⇒ Journals.

⇒ Dispositivos de Almacenamiento y particiones para crear volúmenes lógicos -> **lvmlockd**.

⇒ Protocolo de Tiempo -> NTP/PTP ⇒ chrony

Consideraciones GFS2

⇒ NO se soporta en un único nodo. (Máximo 16 nodos).

⇒ Tamaño del Stma. de Archivos -> Mejor que sea PEQUEÑO.

⇒ Tamaño del Bloque por defecto -> 4K.

⇒ Tamaño del diario (journal) por defecto -> 128MB.

⇒ Tamaño y número de recursos. Al crear el FS (mkfs.gfs2), se divide en porciones uniformes -> grupo de recursos, que van desde 32MB hasta 2GB. (El valor predeterminado puede anularse con la opción **-r** de mkfs.gfs2.

ACTUALIZACIONES ⇒ ATIME

Cada ínodo de archivos/directorios, tiene 3 marcas de tiempo asociadas:

⇒ **atime (ls -lu)** (Por defecto). Hora del último acceso. Una hora o marca de tiempo de acceso es la

última vez que un archivo fue leído, leído por uno de los procesos directamente o mediante comandos y scripts.

⇒ **mtime (ls -l)** Hora de modificación es la hora del último cambio en el contenido del archivo. 'Modificación' significa que algo dentro del archivo fue modificado o eliminado, o que se agregaron nuevos datos.

⇒ **ctime (ls -lc)** Marca de tiempo modificada que se refiere a los cambios realizados en el atributo de un archivo, como la propiedad, el permiso de acceso. Es el momento en el que cambiaron los metadatos relacionados con el archivo.

Para ver la hora de modificación, la hora de acceso y la hora de cambio de un archivo en particular (todo a la vez):

```
# stat fichero.txt
  Fichero: fichero.txt
  Tamaño: 5102          Bloques: 16         Bloque E/S: 4096   fichero regular
Device: 253,6   Inode: 235083337   Links: 1
Acceso: (0644/-rw-r--r--)  Uid: ( 1000/    labs)   Gid: ( 1000/    labs)
Contexto: unconfined_u:object_r:admin_home_t:s0
      Acceso: 2021-12-08 10:14:24.004826310 +0100
Modificación: 2021-12-08 10:00:03.467476863 +0100
      Cambio: 2021-12-08 10:11:13.706847090 +0100
    Creación: 2021-12-08 10:00:03.467476863 +0100
```

Reducción del tiempo de actualización atime (/etc/fstab):

atime, puede requerir una cantidad de tráfico significativa de escritura y de bloqueo innecesaria degradando el rendimiento -> Mejor reducir o desactivar la frecuencia de actualización:

⇒ **relatime** Actualiza si la actualización anterior es más antigua que mtime o ctime.

⇒ **noatime** Desactiva actualizaciones para archivos y directorios.

⇒ **nodiratime** Desactiva solo para directorios. (Mejor noatime).

RECOMENDACIÓN: Montar el FS con las opciones: noatime,acl

→ Instalamos pacemaker básico → CentOS 7.x

Instalamos pacemaker básico - Requisitos:

⇒ Sincronizar Máquinas -> chrony.

⇒ ssh-copy-id entre Máquinas.

⇒ hosts:

```
192.168.10.161 gfs2-01.cadilinea.lan      gfs2-01
192.168.10.162 gfs2-02.cadilinea.lan      gfs2-02
192.168.10.170 gfs2-target.cadilinea.lan  gfs2-target
```

```
# yum install pacemaker pcs
```

```
# firewall-cmd --permanent --add-service=high-availability

# firewall-cmd --reload

# systemctl enable --now pcsd.service

#  passwd hacluster

gfs2-01 ~ # pcs cluster auth gfs2-01 gfs2-02 -u hacluster
gfs2-01 ~ # pcs cluster setup --name HA_gfs2 gfs2-01 gfs2-02
gfs2-01 ~ # pcs cluster start --all
gfs2-01 ~ # pcs cluster enable --all
gfs2-01 ~ # pcs property set stonith-enabled=false
gfs2-01 ~ # pcs property set no-quorum-policy=ignore
gfs2-01 ~ # pcs property set default-resource-stickiness="INFINITY"
gfs2-01 ~ # pcs resource create ClusterIP ocf:heartbeat:IPaddr2 ip=192.168.10.150
cidr_netmask=24 op monitor interval=30s

gfs2-01 ~ # pcs property
Cluster Properties:
 cluster-infrastructure: corosync
 cluster-name: HA_gfs2
 dc-version: 1.1.23-1.el7_9.1-9acf116022
 default-resource-stickiness: INFINITY
 have-watchdog: false
 no-quorum-policy: ignore
 stonith-enabled: false

gfs2-01 ~ # pcs status
Cluster name: HA_gfs2
Stack: corosync
Current DC: gfs2-02 (version 1.1.23-1.el7_9.1-9acf116022) - partition with quorum
Last updated: Mon May 10 11:23:51 2021
Last change: Mon May 10 11:22:21 2021 by root via cibadmin on gfs2-01

2 nodes configured
1 resource instance configured

Online: [ gfs2-01 gfs2-02 ]

Full list of resources:

 ClusterIP      (ocf::heartbeat:IPaddr2):       Started gfs2-01

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

→ Instalamos crmsh → crm => No funciona en todas las versiones →Apuesta actual → pcs

 # vim /etc/yum.repos.d/ha-clustering.repo

[network_ha-clustering_Stable]

name=Stable High Availability/Clustering packages (CentOS_CentOS-7)

type=rpm-md

baseurl=https://download.opensuse.org/repositories/network:/ha-clustering:/Stable/CentOS_CentOS-7/

gpgcheck=1

gpgkey=https://download.opensuse.org/repositories/network:/ha-clustering:/Stable/CentOS_CentOS-7/repodata/repomd.xml.key

enabled=1

 # yum update

 # yum install crm*

→ Crear el cercado → fencing

→ En el host → KVM → Únicamente → redhat00 → Es el anfitrión KVM

```
redhat00 ~ # yum install fence-virtd fence-virtd-libvirt fence-virtd-multicast
redhat00 ~ # mkdir /etc/cluster
redhat00 ~ # dd if=/dev/urandom of=/etc/cluster/fence_xvm.key bs=4k count=1

redhat00 ~ # firewall-cmd --get-active-zones
libvirt
  interfaces: br10
public
  interfaces: virbr0 eno1 eno2 team0 vnet0 vnet1

redhat00 ~ # fence_virtd -c
Module search path [/usr/lib64/fence-virt/]:

Available backends:
    libvirt 0.3
Available listeners:
    vsock 0.1
    multicast 1.2

Listener modules are responsible for accepting requests
from fencing clients.

Listener module [multicast]:
```

The multicast listener module is designed for use environments
where the guests and hosts may communicate over a network using
multicast.

The multicast address is the address that a client will use to
send fencing requests to fence_virtd.

Multicast IP Address [**225.0.0.12**]:

Using ipv4 as family.

Multicast IP Port [**1229**]:

Setting a preferred interface causes fence_virtd to listen only
on that interface. Normally, it listens on all interfaces.
In environments where the virtual machines are using the host
machine as a gateway, this *must* be set (typically to virbr0).
Set to 'none' for no interface.

Interface [br10]: **br10**

The key file is the shared key information which is used to
authenticate fencing requests. The contents of this file must
be distributed to each physical host and virtual machine within
a cluster.

Key File [**/etc/cluster/fence_xvm.key**]:

Backend modules are responsible for routing requests to
the appropriate hypervisor or management layer.

Backend module [**libvirt**]:

The libvirt backend module is designed for single desktops or
servers. Do not use in environments where virtual machines
may be migrated between hosts.

Libvirt URI [qemu:///system]:

Configuration complete.

```
=== Begin Configuration ===
fence_virtd {
        listener = "multicast";
        backend = "libvirt";
        module_path = "/usr/lib64/fence-virt/";
```

```
}

listeners {
        multicast {
                key_file = "/etc/cluster/fence_xvm.key";
                address = "225.0.0.12";
                interface = "br10";
                family = "ipv4";
                port = "1229";
        }

}

backends {
        libvirt {
                uri = "qemu:///system";
        }

}

=== End Configuration ===
Replace /etc/fence_virt.conf with the above [y/N]? y
```

redhat00 ~ # systemctl enable --now fence_virtd.service

redhat00 ~ # firewall-cmd --permanent --add-port=1229/udp
redhat00 ~ # firewall-cmd --reload

redhat00 ~ # ssh 192.168.10.161 mkdir /etc/cluster
redhat00 ~ # ssh 192.168.10.162 mkdir /etc/cluster

redhat00 ~ # scp /etc/cluster/fence_xvm.key root@192.168.10.161:/etc/cluster/
redhat00 ~ # scp /etc/cluster/fence_xvm.key root@192.168.10.162:/etc/cluster/

redhat00 ~ # firewall-cmd --get-active-zones
libvirt
  interfaces: **br10**
public
  interfaces: virbr0 eno1 eno2 team0 vnet0 vnet1

redhat00 ~ # firewall-cmd --add-port=1229/udp --permanent --zone=libvirt
redhat00 ~ # firewall-cmd --add-port=1229/tcp --permanent --zone=libvirt
redhat00 ~ # firewall-cmd --reload
redhat00 ~ # firewall-cmd --list-all --zone=libvirt
libvirt (active)
  target: ACCEPT
  icmp-block-inversion: no

```
  interfaces: br10
  sources:
  services: dhcp dhcpv6 dns ssh tftp
  ports: 1229/udp 1229/tcp
  protocols: icmp ipv6-icmp
  masquerade: no
  forward-ports:
  source-ports:
  icmp-blocks:
  rich rules:
        rule priority="32767" reject
```

→ En los nodos → gfs2-01 / gfs2-02

```
# yum install fence-virt fence-agents-all sbd
# firewall-cmd --add-port=1229/{tcp,udp} --permanent
# firewall-cmd --reload

# pcs property set stonith-enabled=true
# fence_xvm -o list | grep gfs2
gfs2-01                  e6c35ad6-828c-4ffe-a1d8-e3ae0852b1e3 on
gfs2-02                  3e27d173-087f-4081-8b86-de1230ff3ada on

# pcs resource describe stonith:fence_xvm
stonith:fence_xvm - Fence agent for virtual machines

fence_xvm is an I/O Fencing agent which can be used withvirtual machines.

Resource options:
  debug: Specify (stdin) or increment (command line) debug level
  ip_family: IP Family ([auto], ipv4, ipv6)
  multicast_address: Multicast address (default=225.0.0.12 / ff05::3:1)
  ipport: TCP, Multicast, or VMChannel IP port (default=1229)
  retrans: Multicast retransmit time (in 1/10sec; default=20)
  auth: Authentication (none, sha1, [sha256], sha512)
  hash: Packet hash strength (none, sha1, [sha256], sha512)
  key_file: Shared key file (default=/etc/cluster/fence_xvm.key)
  port: Virtual Machine (domain name) to fence
  use_uuid: Treat [domain] as UUID instead of domain name. This is provided for compatibility
with older fence_xvmd installations.
  action: Fencing action (null, off, on, [reboot], status, list, list-status, monitor, validate-all,
metadata)
  timeout: Fencing timeout (in seconds; default=30)
  delay: Fencing delay (in seconds; default=0)
  domain: Virtual Machine (domain name) to fence (deprecated; use port)

Default operations:
```

```
 null: interval=0s
 on: interval=0s
 off: interval=0s
 reboot: interval=0s
 metadata: interval=0s
 monitor: interval=60s
 list: interval=0s
 stop: interval=0s timeout=20s
 start: interval=0s timeout=20s
```

→ **Método 1**

```
gfs2-01 ~ # pcs stonith create fence_gfs2-01 fence_xvm   key_file="/etc/cluster/fence_xvm.key"
domain="gfs2-01"   pcmk_host_list="gfs2-01.cadilinea.lan" pcmk_off_action=yes

gfs2-01 ~ # pcs stonith create fence_gfs2-02 fence_xvm   key_file="/etc/cluster/fence_xvm.key"
domain="gfs2-02"   pcmk_host_list="gfs2-02.cadilinea.lan" pcmk_off_action=yes
```

→ ==> **Método 2** → **Preferido**

```
gfs2-01 ~ # pcs stonith create fence_all fence_xvm   key_file="/etc/cluster/fence_xvm.key"
pcmk_host_map="gfs2-01.cadilinea.lan:gfs2-01,gfs2-02.cadilinea.lan:gfs2-02"
pcmk_host_list="gfs2-01,gfs2-02"   pcmk_host_check=static-list
```

→ **Prueba de fencing**

```
gfs2-01 ~ # pcs stonith
 fence_gfs2-01  (stonith:fence_xvm):    Started gfs2-02
 fence_gfs2-02  (stonith:fence_xvm):    Started gfs2-01

gfs2-01 ~ # fence_xvm -o list | grep gfs2
gfs2-01                  e6c35ad6-828c-4ffe-a1d8-e3ae0852b1e3 on
gfs2-02                  3e27d173-087f-4081-8b86-de1230ff3ada on

gfs2-01 ~ # fence_xvm -o off -H 3e27d173-087f-4081-8b86-de1230ff3ada
gfs2-01 ~ # pcs cluster status
Cluster Status:
 Stack: corosync
 Current DC: gfs2-01 (version 1.1.23-1.el7_9.1-9acf116022) - partition with quorum
 Last updated: Mon May 10 20:27:51 2021
 Last change: Mon May 10 20:26:48 2021 by root via cibadmin on gfs2-01
 2 nodes configured
 3 resource instances configured

PCSD Status:
  gfs2-01: Online
  gfs2-02: Offline
```

```
gfs2-01 ~ # fence_xvm -o on -H 3e27d173-087f-4081-8b86-de1230ff3ada
gfs2-01 ~ # pcs cluster status
Cluster Status:
 Stack: corosync
 Current DC: gfs2-01 (version 1.1.23-1.el7_9.1-9acf116022) - partition with quorum
 Last updated: Mon May 10 20:29:05 2021
 Last change: Mon May 10 20:26:48 2021 by root via cibadmin on gfs2-01
 2 nodes configured
 3 resource instances configured

PCSD Status:
  gfs2-01: Online
  gfs2-02: Online
```

→ Desde → gfs2-01 / gfs2-02

```
# crm
crm(live)# configure
crm(live)configure# show
node 1: gfs2-01
node 2: gfs2-02
primitive ClusterIP IPaddr2 \
        params cidr_netmask=24 ip=192.168.10.150 \
        op monitor interval=30s \
        op start interval=0s timeout=20s \
        op stop interval=0s timeout=20s
primitive fence_gfs2-01 stonith:fence_xvm \
        params domain=gfs2-01 key_file="/etc/cluster/fence_xvm.key" pcmk_host_list=gfs2-
01.cadilinea.lan pcmk_off_action=yes \
        op monitor interval=60s
primitive fence_gfs2-02 stonith:fence_xvm \
        params domain=gfs2-02 key_file="/etc/cluster/fence_xvm.key" pcmk_host_list=gfs2-
02.cadilinea.lan pcmk_off_action=yes \
        op monitor interval=60s
property cib-bootstrap-options: \
        have-watchdog=false \
        dc-version=1.1.23-1.el7_9.1-9acf116022 \
        cluster-infrastructure=corosync \
        cluster-name=HA_gfs2 \
        no-quorum-policy=ignore \
        default-resource-stickiness=INFINITY \
        stonith-enabled=true
```

→ Instalamos GFS2 administrado por pacemaker → (CentOS-7)

=> Partimos de esta configuración ahora:

gfs2-01 ~ # pcs resource create ClusterIP ocf:heartbeat:IPaddr2 ip=192.168.10.150 cidr_netmask=24 op monitor interval=30s

gfs2-01 ~ # pcs status
Cluster name: HA_gfs2
Stack: corosync
Current DC: gfs2-02 (version 1.1.23-1.el7_9.1-9acf116022) - partition with quorum
Last updated: Tue Dec 14 19:31:49 2021
Last change: Tue Dec 14 19:25:46 2021 by root via cibadmin on gfs2-01

2 nodes configured
1 resource instance configured

Online: [gfs2-01 gfs2-02]

Full list of resources:

ClusterIP (ocf::heartbeat:IPaddr2): Started gfs2-01

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

⇒ Empezamos por el fencing (Utilizamos ahora el preferido).

gfs2-01 ~ # pcs stonith create fence_all fence_xvm key_file="/etc/cluster/fence_xvm.key" pcmk_host_map="gfs2-01.cadilinea.lan:gfs2-01,gfs2-02.cadilinea.lan:gfs2-02" pcmk_host_list="gfs2-01,gfs2-02" pcmk_host_check=static-list

gfs2-01 ~ # pcs resource
ClusterIP (ocf::heartbeat:IPaddr2): Started gfs2-01
gfs2-01 ~ # pcs stonith
fence_all (stonith:fence_xvm): Started gfs2-02

gfs2-01 ~ # man pengine
…
no-quorum-policy = enum [stop]
What to do when the cluster does not have quorum

What to do when the cluster does not have quorum Allowed values: stop, **freeze**, ignore,

suicide

→ **Notas importantes:**

El valor de **no-quorum-policy** se predetermina a **stop**, indicando que cuando el quorum se pierde, todos los recursos en la partición restante se detendrán inmediatamente. Por lo general, este valor predeterminado es el más seguro y la opción más óptima, pero a diferencia de la mayoría de los recursos, GFS2 requiere qurorum para funcionar. Cuando el quorum se pierde, la aplicación que usa GFS2 se monta y no puede detenerse correctamente. Cualquier intento para detener estos recursos sin quorum fallará, lo cual resulta en el cercado de todo el clúster cada vez que se pierda el quorum. Para afrontar esta situación, establezca el valor **no-quorum-policy=freeze** cuando GFS2 esté en uso. Esto significa que cuando el quorum se pierde, la partición restante no hará nada hasta que se recupere el quorum.

pcs property set no-quorum-policy=freeze

Después de asegurarse de que el tipo que está viendo está configurado a **3** en el archivo **/etc/lvm/lvm.conf** para soportar el bloqueo en clúster, cree el LV en clúster y de formato al volúmen con un sistema de archivos GFS2. Verifique si crea suficientes diarios para cada uno de los nodos en su clúster.

⇒ **Ajustamos stonith:**

```
gfs2-01 ~ # pcs property set no-quorum-policy=freeze
gfs2-01 ~ # pcs property set stonith-timeout=120s
gfs2-01 ~ # pcs property list --all|grep stonith
 stonith-action: reboot
 stonith-enabled: true
 stonith-max-attempts: 10
 stonith-timeout: 120s
 stonith-watchdog-timeout: (null)

gfs2-01 ~ # pcs property list --defaults
Cluster Properties:
 batch-limit: 0
 cluster-delay: 60s
 cluster-ipc-limit: 500
 cluster-name: (null)
 cluster-recheck-interval: 15min
 concurrent-fencing: true
 crmd-finalization-timeout: 30min
 crmd-integration-timeout: 3min
 crmd-transition-delay: 0s
 dc-deadtime: 20s
 default-action-timeout: (null)
 default-resource-stickiness: (null)
 election-timeout: 2min
```

```
 enable-acl: false
 enable-startup-probes: true
 fence-reaction: stop
 have-watchdog: false
 is-managed-default: (null)
 load-threshold: 80%
 maintenance-mode: false
 migration-limit: -1
 no-quorum-policy: stop
 node-action-limit: 0
 node-health-base: 0
 node-health-green: 0
 node-health-red: -INFINITY
 node-health-strategy: none
 node-health-yellow: 0
 notification-agent: /dev/null
 notification-recipient:
 pe-error-series-max: -1
 pe-input-series-max: 4000
 pe-warn-series-max: 5000
 placement-strategy: default
 remove-after-stop: false
 shutdown-escalation: 20min
 shutdown-lock: false
 shutdown-lock-limit: 0
 start-failure-is-fatal: true
 startup-fencing: true
 stonith-action: reboot
 stonith-enabled: true
 stonith-max-attempts: 10
 stonith-timeout: 60s
 stonith-watchdog-timeout: (null)
 stop-all-resources: false
 stop-orphan-actions: true
 stop-orphan-resources: true
 symmetric-cluster: true

gfs2-01 ~ # pcs stonith show --full
 Resource: fence_all (class=stonith type=fence_xvm)
  Attributes: key_file=/etc/cluster/fence_xvm.key pcmk_host_check=static-list
pcmk_host_list=gfs2-01,gfs2-02 pcmk_host_map=gfs2-01.cadilinea.lan:gfs2-01,gfs2-
02.cadilinea.lan:gfs2-02
  Operations: monitor interval=60s (fence_all-monitor-interval-60s)

gfs2-01 ~ # stonith_admin --reboot gfs2-02
gfs2-01 ~ # pcs stonith
 fence_all       (stonith:fence_xvm):  Started gfs2-01
```

⇒ **Paqueteria necesaria GFS2:**

```
gfs2-01 ~ # pcs resource defaults resource-stickiness=200
gfs2-01 ~ # pcs resource defaults
resource-stickiness=200

# yum install lvm2-cluster lvm2-lock gfs2-utils dlm

# cat /etc/lvm/lvm.conf |grep locking_type |grep -v "#"
    locking_type = 1
```

⇒ **Habilitamos locking cluster, y deshabilitamos metadatos.**

```
# lvmconf --enable-cluster
# cat /etc/lvm/lvm.conf |grep locking_type |grep -v "#"
   locking_type = 3

# cat /etc/lvm/lvm.conf |grep use_lvmetad |grep -v "#"
   use_lvmetad = 0

# systemctl disable lvm2-lvmetad.service lvm2-lvmetad.socket
# systemctl stop lvm2-lvmetad.service lvm2-lvmetad.socket

# shutdown -r 0

gfs2-01 ~ # pcs stonith show fence_gfs2-0{1,2}
 Resource: fence_gfs2-01 (class=stonith type=fence_xvm)
  Attributes: domain=gfs2-01 key_file=/etc/cluster/fence_xvm.key pcmk_host_list=gfs2-
01.cadilinea.lan pcmk_off_action=yes
  Operations: monitor interval=60s (fence_gfs2-01-monitor-interval-60s)
 Resource: fence_gfs2-02 (class=stonith type=fence_xvm)
  Attributes: domain=gfs2-02 key_file=/etc/cluster/fence_xvm.key pcmk_host_list=gfs2-
02.cadilinea.lan pcmk_off_action=yes
  Operations: monitor interval=60s (fence_gfs2-02-monitor-interval-60s)

gfs2-01 ~ # pcs resource create dlm ocf:pacemaker:controld op monitor interval=30s on-
fail=fence clone interleave=true ordered=true

gfs2-01 ~ # pcs resource create clvmd ocf:heartbeat:clvm op monitor interval=30s on-
fail=fence clone interleave=true ordered=true
```

→ Restricciones → **constraints (loc)**

```
gfs2-01 ~ # pcs constraint location ClusterIP prefers gfs2-01=200
gfs2-01 ~ # pcs constraint location fence_all prefers gfs2-01=200
gfs2-01 ~ # pcs constraint location ClusterIP prefers gfs2-02=100
gfs2-01 ~ # pcs constraint location fence_all prefers gfs2-02=100
```

```
gfs2-01 ~ # pcs constraint order  start dlm-clone then clvmd-clone
gfs2-01 ~ # pcs constraint colocation add clvmd-clone with dlm-clone

gfs2-01 ~ # pcs property set no-quorum-policy=freeze

gfs2-01 ~ # pcs status
Cluster name: HA_gfs2
Stack: corosync
Current DC: gfs2-01 (version 1.1.23-1.el7_9.1-9acf116022) - partition with quorum
Last updated: Tue Dec 14 20:54:33 2021
Last change: Tue Dec 14 20:47:36 2021 by root via cibadmin on gfs2-01

2 nodes configured
6 resource instances configured

Online: [ gfs2-01 gfs2-02 ]

Full list of resources:

 ClusterIP      (ocf::heartbeat:IPaddr2):       Started gfs2-01
 fence_all      (stonith:fence_xvm):  Started gfs2-01
 Clone Set: dlm-clone [dlm]
     Started: [ gfs2-01 gfs2-02 ]
 Clone Set: clvmd-clone [clvmd]
     Started: [ gfs2-01 gfs2-02 ]

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

→ Creamos el cluster lógico → iSCSI → gfs2-target (CentOS 7.x)

```
gfs2-target ~ # yum install targetcli
gfs2-target ~ # lsblk /dev/vd[b-z]
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
vdb  252:16   0  20G  0 disk
vdc  252:32   0  20G  0 disk
vdd  252:48   0  20G  0 disk

gfs2-target ~ # targetcli
targetcli shell version 2.1.53
Copyright 2011-2013 by Datera, Inc and others.
For help on commands, type 'help'.
/> backstores/block create name=gfs2 dev=/dev/vdb
Created block storage object gfs2 using /dev/vdb.
```

```
/> iscsi/ create iqn.2022-01.lan.cadilinea:target
Created target iqn.2022-01.lan.cadilinea:target.
Created TPG 1.
Global pref auto_add_default_portal=true
Created default portal listening on all IPs (0.0.0.0), port 3260.
/> iscsi/iqn.2022-01.lan.cadilinea:target/tpg1/acls create iqn.2022-01.lan.cadilinea:gfs2
Created Node ACL for iqn.2022-01.lan.cadilinea:gfs2
/> iscsi/iqn.2022-01.lan.cadilinea:target/tpg1/ set attribute authentication=0
Parameter authentication is now '0'.
/> iscsi/iqn.2022-01.lan.cadilinea:target/tpg1/ set attribute generate_node_acls=1
Parameter generate_node_acls is now '1'.
/> iscsi/iqn.2022-01.lan.cadilinea:target/tpg1/luns create /backstores/block/gfs2
Created LUN 0.
Created LUN 0->0 mapping in node ACL iqn.2022-01.lan.cadilinea:gfs2
/> ls
o- / ..... [...]
  o- backstores ..... [...]
  | o- block ..... [Storage Objects: 1]
  | | o- gfs2 ..... [/dev/vdb (20.0GiB) write-thru activated]
  | |   o- alua ..... [ALUA Groups: 1]
  | |     o- default_tg_pt_gp ..... [ALUA state: Active/optimized]
  | o- fileio ..... [Storage Objects: 0]
  | o- pscsi ..... [Storage Objects: 0]
  | o- ramdisk ..... [Storage Objects: 0]
  o- iscsi ..... [Targets: 1]
  | o- iqn.2022-01.lan.cadilinea:target ..... [TPGs: 1]
  |   o- tpg1 ..... [gen-acls, no-auth]
  |     o- acls ..... [ACLs: 1]
  |     | o- iqn.2022-01.lan.cadilinea:gfs2 ..... [Mapped LUNs: 1]
  |     |   o- mapped_lun0 ..... [lun0 block/gfs2 (rw)]
  |     o- luns ..... [LUNs: 1]
  |     | o- lun0 ..... [block/gfs2 (/dev/vdb) (default_tg_pt_gp)]
  |     o- portals ..... [Portals: 1]
  |       o- 0.0.0.0:3260 ..... [OK]
  o- loopback ..... [Targets: 0]
/> exit
Global pref auto_save_on_exit=true
Last 10 configs saved in /etc/target/backup/.
Configuration saved to /etc/target/saveconfig.json

gfs2-target ~ # systemctl enable --now target.service
gfs2-target ~ # firewall-cmd --permanent --add-port=3260/tcp
gfs2-target ~ # firewall-cmd --reload
```

→ Desde los clientes => gfs2-01 / gfs2-02

```
# yum install iscsi-initiator-utils
# iscsiadm  -m discovery -t st -p 192.168.10.170
192.168.10.170:3260,1 iqn.2022-01.lan.cadilinea:target
# iscsiadm -m node -T iqn.2022-01.lan.cadilinea:target -p 192.168.10.170 -l
Logging in to [iface: default, target: iqn.2022-01.lan.cadilinea:target, portal: 192.168.10.170,3260]
```

(multiple)
Login to [iface: default, target: iqn.2022-01.lan.cadilinea:target, portal: 192.168.10.170,3260] successful.

 # vim /etc/iscsi/initiatorname.iscsi
#InitiatorName=iqn.1994-05.com.redhat:fd7d2c73484d
#InitiatorName=iqn.2022-01.lan.cadilinea:target
InitiatorName=iqn.2022-01.lan.cadilinea:gfs2

 # lsblk /dev/sd[a-z]
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda 8:0 0 20G 0 disk

 # man vgcreate
…
vgcreate VG_new PV ...
[-A|--autobackup y|n]
[-c|--clustered y|n]
[-l|--maxlogicalvolumes Number]
[-p|--maxphysicalvolumes Number]
[-M|--metadatatype lvm2|lvm1]
[-s|--physicalextentsize Size[m|UNIT]]
…

→ Solo desde un nodo → gfs2-01

gfs2-01 ~ # vgcreate --autobackup=y --clustered=y vg_gfs2 /dev/sda
Physical volume "/dev/sda" successfully created.
Clustered volume group "vg_gfs2" successfully created

→ También y de forma abreviada :
 # vgcreate -Ay -cy vg_gfs2 /dev/sda

gfs2-01 ~ # lvcreate -L5G -n lv_gfs2 vg_gfs2
Logical volume "lv_gfs2" created.

gfs2-01 ~ # mkfs.gfs2 -p lock_dlm -t HA_gfs2:gfs2_compartido -j 2 -J 32 /dev/vg_gfs2/lv_gfs2
It appears to contain an existing filesystem (gfs2)
/dev/vg_gfs2/lv_gfs2 is a symbolic link to /dev/dm-2
This will destroy any data on /dev/dm-2
Are you sure you want to proceed? [y/n] y
Discarding device contents (may take a while on large devices): Done
Adding journals: Done
Building resource groups: Done
Creating quota file: Done
Writing superblock and syncing: Done
Device: /dev/vg_gfs2/lv_gfs2

```
Block size:            4096
Device size:           5,00 GB (1310720 blocks)
Filesystem size:       5,00 GB (1310718 blocks)
Journals:              2
Journal size:          32MB
Resource groups:       22
Locking protocol:      "lock_dlm"
Lock table:            "HA_gfs2:gfs2_compartido"
UUID:                  b81d420d-19ff-4367-9055-b8c08da792f3
```

```
gfs2-01 ~ # lsblk /dev/sda -f
NAME            FSTYPE      LABEL              UUID                                   MOUNTPOINT
sda             LVM2_member                    uUNKP6-K8W8-VjIr-zfA8-1L5J-t9Ag-ltEN0L
└─vg_gfs2-lv_gfs2 gfs2        HA_gfs2:gfs2_compartido b81d420d-19ff-4367-9055-b8c08da792f3
gfs2-01 ~ # ssh gfs2-02  lsblk /dev/sda -f
NAME            FSTYPE      LABEL              UUID                                   MOUNTPOINT
sda             LVM2_member                    uUNKP6-K8W8-VjIr-zfA8-1L5J-t9Ag-ltEN0L
└─vg_gfs2-lv_gfs2 gfs2        HA_gfs2:gfs2_compartido b81d420d-19ff-4367-9055-b8c08da792f3
gfs2-target ~ # lsblk /dev/vdb -f
NAME FSTYPE      LABEL UUID                                   MOUNTPOINT
vdb  LVM2_member       uUNKP6-K8W8-VjIr-zfA8-1L5J-t9Ag-ltEN0L
```

```
gfs2-01 ~ # dlm_tool ls
dlm lockspaces
name          clvmd
id            0x4104eefa
flags         0x00000000
change        member 2 joined 1 remove 0 failed 0 seq 1,1
members       1 2
```

```
gfs2-01 ~ # dlm_tool status
cluster nodeid 1 quorate 1 ring seq 153 153
daemon now 828 fence_pid 0
node 1 M add 32 rem 0 fail 0 fence 0 at 0 0
node 2 M add 32 rem 0 fail 0 fence 0 at 0 0
```

```
gfs2-01 ~ # pcs resource create FS_gfs2 Filesystem device="/dev/mapper/vg_gfs2-lv_gfs2"
directory="/mnt/gfs2" fstype="gfs2" options="noatime" op monitor interval=10s on-
fail=fence clone interleave=true
```

```
gfs2-01 ~ # pcs constraint order start clvmd-clone then FS_gfs2-clone
gfs2-01 ~ # pcs constraint colocation add FS_gfs2-clone with clvmd-clone
gfs2-01 ~ # dlm_tool ls
dlm lockspaces
name          gfs2_compartido
id            0x68b028f4
flags         0x00000000
```

```
change       member 2 joined 1 remove 0 failed 0 seq 1,1
members      1 2

name         clvmd
id           0x4104eefa
flags        0x00000000
change       member 2 joined 1 remove 0 failed 0 seq 1,1
members      1 2
```

gfs2-01 ~ # pcs status --full

```
Cluster name: HA_gfs2
Stack: corosync
Current DC: gfs2-01 (1) (version 1.1.23-1.el7_9.1-9acf116022) - partition with quorum
Last updated: Mon Jan  3 19:12:39 2022
Last change: Mon Jan  3 18:59:05 2022 by root via cibadmin on gfs2-01

2 nodes configured
8 resource instances configured

Online: [ gfs2-01 (1) gfs2-02 (2) ]

Full list of resources:

 fence_all      (stonith:fence_xvm):  Started gfs2-01
 Clone Set: dlm-clone [dlm]
     dlm (ocf::pacemaker:controld):    Started gfs2-02
     dlm (ocf::pacemaker:controld):    Started gfs2-01
     Started: [ gfs2-01 gfs2-02 ]
 Clone Set: clvmd-clone [clvmd]
     clvmd      (ocf::heartbeat:clvm): Started gfs2-02
     clvmd      (ocf::heartbeat:clvm): Started gfs2-01
     Started: [ gfs2-01 gfs2-02 ]
 ClusterIP      (ocf::heartbeat:IPaddr2):       Started gfs2-01
 Clone Set: FS_gfs2-clone [FS_gfs2]
     FS_gfs2    (ocf::heartbeat:Filesystem):   Started gfs2-02
     FS_gfs2    (ocf::heartbeat:Filesystem):   Started gfs2-01
     Started: [ gfs2-01 gfs2-02 ]

Node Attributes:
* Node gfs2-01 (1):
* Node gfs2-02 (2):

Migration Summary:
* Node gfs2-02 (2):
* Node gfs2-01 (1):

Fencing History:
```

PCSD Status:
  gfs2-01: Online
  gfs2-02: Online

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

⇒ Testing:

gfs2-01 ~ # df -hT | grep gfs2

/dev/mapper/vg_gfs2-lv_gfs2 gfs2 5,0G 67M 5,0G 2% /mnt/gfs2

gfs2-01 ~ # ssh gfs2-02 df -hT | grep gfs2

/dev/mapper/vg_gfs2-lv_gfs2 gfs2 5,0G 67M 5,0G 2% /mnt/gfs2

gfs2-01 ~ # touch /mnt/gfs2/Desde-01.txt

gfs2-01 ~ # ssh gfs2-02 touch /mnt/gfs2/Desde-02.txt

gfs2-01 ~ # ls /mnt/gfs2/

Desde-01.txt Desde-02.txt

gfs2-01 ~ # ssh gfs2-02 ls /mnt/gfs2/

Desde-01.txt

Desde-02.txt

→ TOPOLOGIA PARA GFS2 (RedHat 8.x):

Anfitrión KVM
192.168.1.250 hp.cadilinea.lan **hp** **# .1.250**

Clientes de Réplica para GFS2 (initiators). Pacemaker/Corosync
192.168.10.161 rhel8-gfs2-01.cadilinea.lan **rhel8-gfs2-01** **# .161**
192.168.10.162 rhel8-gfs2-02.cadilinea.lan **rhel8-gfs2-02** **# .162**

Server iSCSI (target).
192.168.10.170 rhel8-gfs2-target.cadilinea.lan **rhel8-gfs2-target** **# .170**

→ Creamos un cluster básico: pacemaker/corosync (versión 2.x) ⇒ HA_gfs2

 # dnf install pcs pacemaker fence-agents-all -y
 # firewall-cmd --permanent –add-service=high-availability # 2224/tcp
 # firewall-cmd --reload

 # passwd hacluster
 # systemctl enable --now pcsd.service

rhel8-gfs2-01 ~ # pcs host auth rhel8-gfs2-01 rhel8-gfs2-02 -u hacluster
Password:
rhel8-gfs2-02: Authorized
rhel8-gfs2-01: Authorized

rhel8-gfs2-01 ~ # pcs cluster setup HA_gfs2 --start rhel8-gfs2-01 rhel8-gfs2-02
rhel8-gfs2-01 ~ # pcs cluster enable --all
rhel8-gfs2-01 ~ # pcs status
Cluster name: **HA_gfs2**

WARNINGS:
No stonith devices and stonith-enabled is not false

Cluster Summary:
  * Stack: corosync
  * Current DC: rhel8-gfs2-01 (version 2.1.0-8.el8-7c3f660707) - partition with quorum
  * Last updated: Mon Dec 20 17:51:24 2021
  * Last change: Mon Dec 20 17:49:44 2021 by hacluster via crmd on rhel8-gfs2-01
  * 2 nodes configured
  * 0 resource instances configured

Node List:
  * Online: [rhel8-gfs2-01 rhel8-gfs2-02]

Full List of Resources:
  * No resources

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

⇒ **Configuramos para el clúster ⇒ HA_gfs2 ⇒ dlm.service lvmlockd.service**

rhel8-gfs2-01 ~ # pcs resource create dlm --group locking ocf:pacemaker:controld op monitor interval=30s on-fail=fence
rhel8-gfs2-01 ~ # pcs resource clone locking interleave=true
rhel8-gfs2-01 ~ # pcs resource create lvmlockd --group locking ocf:heartbeat:lvmlockd op monitor interval=30s on-fail=fence

rhel8-gfs2-01 ~ # pcs property set no-quorum-policy=freeze
(Si no hay quorum congelamos).

 # dnf install fence-*

⇒ DESDE EL ANFITRIÓN KVM .1.250 => Configuración fencing → fence_xvm

Solo desde el anfitrión KVM ⇒ hp

hp ~ # yum install fence-virtd fence-virtd-libvirt fence-virtd-multicast

Creamos la key:

hp ~ # mkdir /etc/cluster

hp ~ # dd if=/dev/urandom of=/etc/cluster/fence_xvm.key bs=4k count=11

1+0 registros leídos

1+0 registros escritos

4096 bytes (4,1 kB, 4,0 KiB) copied, 0,00112964 s, 3,6 MB/s

Comprobamos los interfaces para libvirt:

hp ~ # firewall-cmd --get-active-zones

FedoraWorkstation

interfaces: virbr0 eno1

interfaces: br10

Generamos la configuración de 'fencing' y para todas las máquinas virtuales KVM:

hp ~ # fence_virtd -c

Module search path [/usr/lib64/fence-virt/]:

Available backends:
  libvirt 0.3
Available listeners:
  vsock 0.1
  multicast 1.2

Listener modules are responsible for accepting requests
from fencing clients.

Listener module [multicast]:

The multicast listener module is designed for use environments
where the guests and hosts may communicate over a network using
multicast.

The multicast address is the address that a client will use to
send fencing requests to fence_virtd.

Multicast IP Address [225.0.0.12]:

Using ipv4 as family.

Multicast IP Port [1229]:

Setting a preferred interface causes fence_virtd to listen only
on that interface. Normally, it listens on all interfaces.
In environments where the virtual machines are using the host
machine as a gateway, this *must* be set (typically to virbr0).
Set to 'none' for no interface.

Interface [virbr0]: br10

The key file is the shared key information which is used to
authenticate fencing requests. The contents of this file must
be distributed to each physical host and virtual machine within
a cluster.

Key File [/etc/cluster/fence_xvm.key]:

Backend modules are responsible for routing requests to
the appropriate hypervisor or management layer.

Backend module [libvirt]:

The libvirt backend module is designed for single desktops or
servers. Do not use in environments where virtual machines
may be migrated between hosts.

Libvirt URI [qemu:///system]:

Configuration complete.

```
=== Begin Configuration ===
backends {
	libvirt {
		uri = "qemu:///system";
	}

}

listeners {
	multicast {
		port = "1229";
		family = "ipv4";
		interface = "br10";
		address = "225.0.0.12";
		key_file = "/etc/cluster/fence_xvm.key";
	}
```

```
}

fence_virtd {
        module_path = "/usr/lib64/fence-virt/";
        backend = "libvirt";
        listener = "multicast";
}

=== End Configuration ===
Replace /etc/fence_virt.conf with the above [y/N]? y
```

hp ~ # cat /etc/fence_virt.conf

```
backends {
        libvirt {
                uri = "qemu:///system";
        }

}

listeners {
        multicast {
                port = "1229";
                family = "ipv4";
                interface = "br10";
                address = "225.0.0.12";
                key_file = "/etc/cluster/fence_xvm.key";
        }

}

fence_virtd {
        module_path = "/usr/lib64/fence-virt/";
        backend = "libvirt";
        listener = "multicast";
}
```

Habilitamos servicios y puertos:

hp ~ # systemctl enable --now fence_virtd.service

hp ~ # firewall-cmd --permanent --add-port=1229/udp

hp ~ # firewall-cmd --reload

Copiamos la key a los nodos:

hp ~ # ssh 192.168.10.161 mkdir /etc/cluster
hp ~ # ssh 192.168.10.162 mkdir /etc/cluster
hp ~ # scp /etc/cluster/fence_xvm.key root@192.168.10.161:/etc/cluster/
hp ~ # scp /etc/cluster/fence_xvm.key root@192.168.10.162:/etc/cluster/

hp ~ # firewall-cmd --get-active-zones
FedoraWorkstation
interfaces: virbr0 eno1
libvirt

hp ~ # firewall-cmd --add-port=1229/udp --permanent --zone=libvirt
hp ~ # firewall-cmd --add-port=1229/tcp --permanent --zone=libvirtt
hp ~ # firewall-cmd --reload
hp ~ # firewall-cmd --list-all --zone=libvirt
libvirt (active)
target: ACCEPT
icmp-block-inversion: no
interfaces: br10
sources:
services: dhcp dhcpv6 dns ssh tftp

protocols: icmp ipv6-icmp
forward: no
masquerade: no
forward-ports:
source-ports:
icmp-blocks:
rich rules:
rule priority=“32767” reject

⇒ **PODEMOS INSTALAR COMO ALTERNATIVA → fence_scsi**

⇒ **DESDE LOS CLIENTES AHORA:**

 # firewall-cmd --add-port=1229/{tcp,udp} --permanent
 # firewall-cmd --reload

 # fence_xvm -o list | grep gfs
rhel8-gfs2-01 a8b5cd67-b511-4fe1-b2be-29d961933c4c on
rhel8-gfs2-02 ca41312b-0d6b-4f50-8f23-b1d79d83306e on

rhel8-gfs2-01 ~ # pcs property

Cluster Properties:
 cluster-infrastructure: corosync
 cluster-name: HA_gfs2
 dc-version: 2.1.0-8.el8-7c3f660707
 have-watchdog: false
 no-quorum-policy: freeze

⇒ **Configuramos el fencing → fence_all**

```
rhel8-gfs2-01 ~ # pcs stonith create fence_all fence_xvm key_file="/etc/cluster/fence_xvm.key" pcmk_host_map="rhel8-gfs2-01.cadilinea.lan:rhel8-gfs2-01,rhel8-gfs2-02.cadilinea.lan:rhel8-gfs2-02" pcmk_host_list="rhel8-gfs2-01,rhel8-gfs2-02" pcmk_host_check=static-list

rhel8-gfs2-01 ~ # pcs stonith
  * fence_all    (stonith:fence_xvm): Started rhel8-gfs2-01
```

⇒ **Repositorios para RedHat / Instalación:**

```
# subscription-manager repos --enable  rhel-8-for-x86_64-resilientstorage-rpms
# subscription-manager repos --enable rhel-8-for-x86_64-appstream-rpms

# dnf install lvm2-lockd gfs2-utils dlm
```

⇒ **Creamos los recursos: dlm / lvmlockd:**

```
rhel8-gfs2-01 ~ # pcs resource create dlm --group locking ocf:pacemaker:controld op monitor interval=30s on-fail=fence
rhel8-gfs2-01 ~ # pcs resource clone locking interleave=true                # Clonamos el Grupo.
rhel8-gfs2-01 ~ # pcs resource create lvmlockd --group locking ocf:heartbeat:lvmlockd op monitor interval=30s on-fail=fence

rhel8-gfs2-01 ~ # pcs status --full
Cluster name: HA_gfs2
Cluster Summary:
  * Stack: corosync
  * Current DC: rhel8-gfs2-01 (1) (version 2.1.0-8.el8-7c3f660707) - partition with quorum
  * Last updated: Mon Dec 20 19:33:21 2021
  * Last change:  Mon Dec 20 19:29:27 2021 by root via cibadmin on rhel8-gfs2-01
  * 2 nodes configured
  * 5 resource instances configured

Node List:
  * Online: [ rhel8-gfs2-01 (1) rhel8-gfs2-02 (2) ]

Full List of Resources:
  * fence_all    (stonith:fence_xvm):   Started rhel8-gfs2-01
  * Clone Set: locking-clone [locking]:
```

```
  * Resource Group: locking:0:
    * dlm       (ocf::pacemaker:controld):     Started rhel8-gfs2-01
    * lvmlockd          (ocf::heartbeat:lvmlockd):      Started rhel8-gfs2-01
  * Resource Group: locking:1:
    * dlm       (ocf::pacemaker:controld):     Started rhel8-gfs2-02
    * lvmlockd          (ocf::heartbeat:lvmlockd):      Started rhel8-gfs2-02

Migration Summary:

Tickets:

PCSD Status:
  rhel8-gfs2-01: Online
  rhel8-gfs2-02: Online

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

→ Ordenamos los recursos → constraint → loc

Utilizando el entrelazado (interleave=true) no es necesario de momento.
Unicamente hacemos mas pegajosos los recursos:

```
rhel8-gfs2-01 ~ # pcs resource defaults resource-stickiness=200
```

```
# ps -ef | grep dlm
root        7007       1  0 17:47 ?        00:00:00 dlm_controld -s 0
root        7009    7007  0 17:47 ?        00:00:00 dlm_controld -s 0
root        7096       1  0 17:47 ?        00:00:00 lvmlockd -p /run/lvmlockd.pid -A 1 -g dlm
```

```
# dlm_tool ls
dlm lockspaces
name          lvm_global
id            0x12aabd2d
flags         0x00000000
change        member 2 joined 1 remove 0 failed 0 seq 1,1
members       1 2
```

→ Utilizamos iSCSI como alternativa → block

```
## Clientes de Réplica para GFS2 (initiators).
192.168.10.161 rhel8-gfs2-01.cadilinea.lan      rhel8-gfs2-01         # .161
192.168.10.162 rhel8-gfs2-02.cadilinea.lan      rhel8-gfs2-02         # .162

## Server iSCSI (target).
192.168.10.170 rhel8-gfs2-target.cadilinea.lan  rhel8-gfs2-target     # .170
```

```
rhel8-gfs2-target ~ # dnf install targetcli
rhel8-gfs2-target ~ # lsblk /dev/vd[b-z]
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
vdb  252:16   0  20G  0 disk
vdc  252:32   0  20G  0 disk
vdd  252:48   0  20G  0 disk

rhel8-gfs2-target ~ # targetcli
targetcli shell version 2.1.53
Copyright 2011-2013 by Datera, Inc and others.
For help on commands, type 'help'.
/> backstores/block create name=gfs2 dev=/dev/vdb
Created block storage object gfs2 using /dev/vdb.
/> iscsi/ create iqn.2022-01.lan.cadilinea:target-gfs2
Created target iqn.2022-01.lan.cadilinea:target-gfs2.
Created TPG 1.
Global pref auto_add_default_portal=true
Created default portal listening on all IPs (0.0.0.0), port 3260.
/> iscsi/iqn.2022-01.lan.cadilinea:target-gfs2/tpg1/acls create iqn.2022-01.lan.cadilinea:target-
gfs2
Created Node ACL for iqn.2022-01.lan.cadilinea:target-gfs2
/> iscsi/iqn.2022-01.lan.cadilinea:target-gfs2/tpg1/ set attribute authentication=0
Parameter authentication is now '0'.
/> iscsi/iqn.2022-01.lan.cadilinea:target-gfs2/tpg1/ set attribute generate_node_acls=1
Parameter generate_node_acls is now '1'.

/> iscsi/iqn.2022-01.lan.cadilinea:target-gfs2/tpg1/luns create /backstores/block/gfs2
Created LUN 0.
Created LUN 0->0 mapping in node ACL iqn.2022-01.lan.cadilinea:target-gfs2
/> ls
o- / ..... [...]
  o- backstores ..... [...]
  | o- block ..... [Storage Objects: 1]
  | | o- gfs2 ..... [/dev/vdb (20.0GiB) write-thru activated]
  | |   o- alua ..... [ALUA Groups: 1]
  | |     o- default_tg_pt_gp ..... [ALUA state: Active/optimized]
  | o- fileio ..... [Storage Objects: 0]
  | o- pscsi ..... [Storage Objects: 0]
  | o- ramdisk ..... [Storage Objects: 0]
  o- iscsi ..... [Targets: 1]
  | o- iqn.2022-01.lan.cadilinea:target-gfs2 ..... [TPGs: 1]
  |   o- tpg1 ..... [gen-acls, no-auth]
  |     o- acls ..... [ACLs: 1]
  |     | o- iqn.2022-01.lan.cadilinea:target-gfs2 ..... [Mapped LUNs: 1]
  |     |   o- mapped_lun0 ..... [lun0 block/gfs2 (rw)]
  |     o- luns ..... [LUNs: 1]
  |     | o- lun0 ..... [block/gfs2 (/dev/vdb) (default_tg_pt_gp)]
  |     o- portals ..... [Portals: 1]
  |       o- 0.0.0.0:3260 ..... [OK]
  o- loopback ..... [Targets: 0]
```

```
/> exit
Global pref auto_save_on_exit=true
Configuration saved to /etc/target/saveconfig.json

rhel8-gfs2-target ~ # systemctl enable --now target.service
rhel8-gfs2-target ~ # firewall-cmd --permanent --add-port=3260/tcp
rhel8-gfs2-target ~ # firewall-cmd --reload
```

→ Desde los clientes ahora:

```
# iscsiadm -m discovery -t st -p 192.168.10.170
192.168.10.170:3260,1 iqn.2022-01.lan.cadilinea:target-gfs2
# iscsiadm -m node -T iqn.2022-01.lan.cadilinea:target-gfs2 -p 192.168.10.170 -l
Logging in to [iface: default, target: iqn.2022-01.lan.cadilinea:target-gfs2, portal:
192.168.10.170,3260]
Login to [iface: default, target: iqn.2022-01.lan.cadilinea:target-gfs2, portal: 192.168.10.170,3260]
successful
# lsblk /dev/sda
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda    8:0    0  20G  0 disk

# vim /etc/iscsi/initiatorname.iscsi
#InitiatorName=iqn.1994-05.com.redhat:195fe743df75
InitiatorName=iqn.2022-01.lan.cadilinea:target-gfs2
```

→ Procedemos con la creación del block compartido descubierto →/dev/sda

```
rhel8-gfs2-01 ~ # vgcreate --shared --locktype dlm shared_gfs2 /dev/sda
  Physical volume "/dev/sda" successfully created.
  Volume group "shared_gfs2" successfully created
  VG shared_gfs2 starting dlm lockspace
  Starting locking.  Waiting until locks are ready...

# vgs
  Reading VG shared_vg1 without a lock.
  VG         #PV #LV #SN Attr   VSize   VFree
  rhel         1   3   0 wz--n- <24,00g   4,80g
  shared_gfs2  1   0   0 wz--ns <20,00g <20,00g
```

→ En un nodo alternativo (NO PRINCIPAL) => rhel8-gfs2-02

```
rhel8-gfs2-02 ~ # vgchange --lock-start shared_gfs2
  VG shared_gfs2 starting dlm lockspace
  Starting locking.  Waiting until locks are ready...
```

→ En un nodo indistinto:

```
rhel8-gfs2-01 ~ # lvcreate --activate sy -L5G -n lv_shared_gfs2 shared_gfs2
```

Logical volume "lv_shared_gfs2" created.

rhel8-gfs2-01 ~ # mkfs.gfs2 -j2 -p lock_dlm -t HA_gfs2:gfs2-compartido /dev/mapper/shared_gfs2-lv_shared_gfs2
/dev/mapper/shared_gfs2-lv_shared_gfs2 is a symbolic link to /dev/dm-6
This will destroy any data on /dev/dm-6
Are you sure you want to proceed? [y/n] y
Discarding device contents (may take a while on large devices): Done
Adding journals: Done
Building resource groups: Done
Creating quota file: Done
Writing superblock and syncing: Done
Device: /dev/mapper/shared_gfs2-lv_shared_gfs2
Block size: 4096
Device size: 5,00 GB (1310720 blocks)
Filesystem size: 5,00 GB (1310718 blocks)
Journals: 2
Journal size: 32MB
Resource groups: 22
Locking protocol: "lock_dlm"
Lock table: "HA_gfs2:gfs2-compartido"
UUID: 9e3f48cc-80ca-459d-b10d-5fbf3cb46f74

→ Creamos un nuevo recurso → shared_lv-gfs2 para que pacemaker lo active automaticamente (clone) , y en todos los nodos:

rhel8-gfs2-01 ~ # pcs resource create shared_lv-gfs2 --group shared_gfs2 ocf:heartbeat:LVM-activate lvname=lv_shared_gfs2 vgname=shared_gfs2 activation_mode=shared vg_access_mode=lvmlockd
rhel8-gfs2-01 ~ # pcs resource clone shared_gfs2 interleave=true # Clonamos el Grupo.

→ Configuramos las restricciones → loc → para que los recursos se inicien en orden !! :

=> location

No procede en principio una configuración para **location** en una configuración entrelazada. (interleave=true).

=> order

rhel8-gfs2-01 ~ # pcs constraint order start locking-clone then shared_gfs2-clone
Adding locking-clone shared_gfs2-clone (kind: Mandatory) (Options: first-action=start then-action=start)

=> colocation

rhel8-gfs2-01 ~ # pcs constraint colocation add shared_gfs2-clone with locking-clone

=> Resúmen:

rhel8-gfs2-01 ~ # pcs constraint
Location Constraints:
Ordering Constraints:
  start locking-clone then start shared_gfs2-clone (kind:Mandatory)
Colocation Constraints:
  shared_gfs2-clone with locking-clone (score:INFINITY)
Ticket Constraints:

→ Preparamos el montaje automático del FS→GFS2 de ejemplo:

rhel8-gfs2-01 ~ # pcs resource create FS-GFS2_mount --group shared_gfs2 ocf:heartbeat:Filesystem device="/dev/shared_gfs2/lv_shared_gfs2" directory="/mnt/gfs2" fstype="gfs2" options=noatime op monitor interval=10s on-fail=fence

rhel8-gfs2-01 ~ # pcs status --full
Cluster name: **HA_gfs2**
Cluster Summary:
  * Stack: corosync
  * Current DC: rhel8-gfs2-01 (1) (version 2.1.0-8.el8-7c3f660707) - partition with quorum
  * Last updated: Fri Dec 31 18:48:55 2021
  * Last change: Fri Dec 31 18:44:57 2021 by root via cibadmin on rhel8-gfs2-01
  * 2 nodes configured
  * 9 resource instances configured

Node List:
  * Online: [rhel8-gfs2-01 (1) rhel8-gfs2-02 (2)]

Full List of Resources:
  * Clone Set: locking-clone [locking]:
    * Resource Group: locking:0:
      * dlm (ocf::pacemaker:controld): Started rhel8-gfs2-02
      * lvmlockd (ocf::heartbeat:lvmlockd): Started rhel8-gfs2-02
    * Resource Group: locking:1:
      * dlm (ocf::pacemaker:controld): Started rhel8-gfs2-01
      * lvmlockd (ocf::heartbeat:lvmlockd): Started rhel8-gfs2-01
  * fence_all (stonith:fence_xvm): Started rhel8-gfs2-01
  * Clone Set: shared_gfs2-clone [shared_gfs2]:
    * Resource Group: shared_gfs2:0:
      * shared_lv-gfs2 (ocf::heartbeat:LVM-activate): Started rhel8-gfs2-02
      * FS-GFS2_mount (ocf::heartbeat:Filesystem): Started rhel8-gfs2-02
    * Resource Group: shared_gfs2:1:
      * shared_lv-gfs2 (ocf::heartbeat:LVM-activate): Started rhel8-gfs2-01
      * FS-GFS2_mount (ocf::heartbeat:Filesystem): Started rhel8-gfs2-01

Migration Summary:

Tickets:

PCSD Status:
  rhel8-gfs2-01: Online
  rhel8-gfs2-02: Online

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

 # dlm_tool ls
dlm lockspaces
name gfs2-compartido
id 0xa0c1379f
flags 0x00000000
change member 2 joined 1 remove 0 failed 0 seq 1,1
members 1 2

name lvm_global
id 0x12aabd2d
flags 0x00000000
change member 2 joined 1 remove 0 failed 0 seq 1,1
members 1 2

→ Es todo OK !!!

 # dlm_tool status
luster nodeid 2 quorate 1 ring seq 213 213
daemon now 489 fence_pid 0
node 1 M add 21 rem 0 fail 0 fence 0 at 0 0
node 2 M add 21 rem 0 fail 0 fence 0 at 0 0

→ Suspender el FS → FS-GFS2_mount

rhel8-gfs2-01 ~ # pcs resource config FS-GFS2_mount
 Resource: FS-GFS2_mount (class=ocf provider=heartbeat type=Filesystem)
  Attributes: device=/dev/shared_gfs2/lv_shared_gfs2 directory=/mnt/gfs2 fstype=gfs2 options=noatime
  Operations: monitor interval=10s on-fail=fence (FS-GFS2_mount-monitor-interval-10s)
              start interval=0s timeout=60s (FS-GFS2_mount-start-interval-0s)
              stop interval=0s timeout=60s (FS-GFS2_mount-stop-interval-0s)

rhel8-gfs2-01 ~ # pcs resource status FS-GFS2_mount
  * Clone Set: shared_gfs2-clone [shared_gfs2]:
    * Started: [rhel8-gfs2-01 rhel8-gfs2-02]

```
rhel8-gfs2-01 ~ # pcs resource status shared_gfs2:0
  * Clone Set: shared_gfs2-clone [shared_gfs2]:
    * Started: [ rhel8-gfs2-02 ]
rhel8-gfs2-01 ~ # pcs resource status shared_gfs2:1
  * Clone Set: shared_gfs2-clone [shared_gfs2]:
* Started: [ rhel8-gfs2-01 ]

rhel8-gfs2-01 ~ # pcs resource disable FS-GFS2_mount
rhel8-gfs2-01 ~ # pcs resource status FS-GFS2_mount
  * Clone Set: shared_gfs2-clone [shared_gfs2]:
    * Resource Group: shared_gfs2:0:
      * shared_lv-gfs2   (ocf::heartbeat:LVM-activate):        Started rhel8-gfs2-02
      * FS-GFS2_mount        (ocf::heartbeat:Filesystem):    Stopped (disabled)
    * Resource Group: shared_gfs2:1:
      * shared_lv-gfs2   (ocf::heartbeat:LVM-activate):        Started rhel8-gfs2-01
      * FS-GFS2_mount        (ocf::heartbeat:Filesystem):    Stopped (disabled)
```

!!! Para chequear el Sistema debe estar previamente DESMONTADO !!!

```
rhel8-gfs2-01 ~ # fsck.gfs2 -y /dev/mapper/shared_gfs2-lv_shared_gfs2 > /tmp/fsck-gfs2.out
rhel8-gfs2-01 ~ # cat /tmp/fsck-gfs2.out
Initializing fsck
Validating resource group index.
Level 1 resource group check: Checking if all rgrp and rindex values are good.
(level 1 passed)
Starting pass1
Reconciling bitmaps.
reconcile_bitmaps completed in 0.016s
pass1 completed in 0.039s
Starting pass1b
pass1b completed in 0.000s
Starting pass2
pass2 completed in 0.001s
Starting pass3
pass3 completed in 0.000s
Starting pass4
pass4 completed in 0.003s
Starting check_statfs
check_statfs completed in 0.000s
fsck.gfs2 complete

rhel8-gfs2-01 ~ # pcs resource enable --wait=100 FS-GFS2_mount
Waiting for the cluster to apply configuration changes (timeout: 100 seconds)...
resource 'FS-GFS2_mount' is running on nodes 'rhel8-gfs2-01', 'rhel8-gfs2-02'
```

→ Crecimiento del FS → +3G

```
rhel8-gfs2-01 ~ # lsblk /dev/sda
NAME                      MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda                         8:0    0  20G  0 disk
└─shared_gfs2-lv_shared_gfs2 253:6    0   5G  0 lvm  /mnt/gfs2

rhel8-gfs2-01 ~ # lvextend --lockopt skiplv -L +3G /dev/shared_gfs2/lv_shared_gfs2
  WARNING: skipping LV lock in lvmlockd.
  WARNING: shared LV may require refresh on other hosts where it is active.
  Size of logical volume shared_gfs2/lv_shared_gfs2 changed from 5,00 GiB (1280 extents) to 8,00 GiB (2048 extents).
  Logical volume shared_gfs2/lv_shared_gfs2 successfully resized.

rhel8-gfs2-01 ~ # lsblk /dev/sda
NAME                      MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda                         8:0    0  20G  0 disk
└─shared_gfs2-lv_shared_gfs2 253:6    0   8G  0 lvm  /mnt/gfs2
rhel8-gfs2-01 ~ # ssh rhel8-gfs2-02 lsblk /dev/sda
NAME                      MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda                         8:0    0  20G  0 disk
└─shared_gfs2-lv_shared_gfs2 253:6    0   5G  0 lvm  /mnt/gfs2
```

→ Refrescamos en los otros nodos:

```
rhel8-gfs2-01 ~ # ssh rhel8-gfs2-02 lvchange --refresh  /dev/shared_gfs2/lv_shared_gfs2
rhel8-gfs2-01 ~ # lsblk /dev/sda
NAME                      MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda                         8:0    0  20G  0 disk
└─shared_gfs2-lv_shared_gfs2 253:6    0   8G  0 lvm  /mnt/gfs2
rhel8-gfs2-01 ~ # ssh rhel8-gfs2-02  lsblk /dev/sda
NAME                      MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda                         8:0    0  20G  0 disk
└─shared_gfs2-lv_shared_gfs2 253:6    0   8G  0 lvm  /mnt/gfs2
```

→ Añadir journals -j2:

```
rhel8-gfs2-01 ~ # gfs2_edit  /dev/shared_gfs2/lv_shared_gfs2
gfs2_edit - Global File System Editor (use with extreme caution)
Block #16    (0x10) of 2097152 (0x200000) (superblock)
(p.1 of 16--Resrv)
00010000 01161970 00000001 00000000 00000000 [...p.....] sb_header.mh_magic
00010010 00000064 00000000 00000709 0000076c [...d.....l]
00010020 00000000 00001000 0000000c 00000000 [.....]
00010030 00000000 00000003 00000000 00002024 [..... $]
00010040 00000000 00000000 00000000 00000000 [.....]
00010050 00000000 00000010 00000000 00004248 [.....BH]
00010060 6c6f636b 5f646c6d 00000000 00000000 [lock_dlm.....]
```

```
00010070 00000000 00000000 00000000 00000000 [................]
00010080 00000000 00000000 00000000 00000000 [................]
00010090 00000000 00000000 00000000 00000000 [................]
000100a0 48415f67 6673323a 67667332 2d636f6d [HA_gfs2:gfs2-com]
000100b0 70617274 69646f00 00000000 00000000 [partido.........]
000100c0 00000000 00000000 00000000 00000000 [................]
000100d0 00000000 00000000 00000000 00000000 [................]
000100e0 00000000 00000000 00000000 00000000 [................]
000100f0 00000000 00000000 00000000 00000000 [................]

Copyright (C) Red Hat, Inc.  2004-2010  All rights reserved. - Mode: Hex view p
```

rhel8-gfs2-01 ~ # gfs2_edit -p jindex /dev/shared_gfs2/lv_shared_gfs2 | grep journal
```
  3/3 [fc7745eb] 1/18 (0x1/0x12) +0: File    journal0
  4/4 [8b70757d] 2/8231 (0x2/0x2027) +0: File    journal1
```

rhel8-gfs2-01 ~ # gfs2_jadd -j 2 /dev/shared_gfs2/lv_shared_gfs2
```
Filesystem: /dev/shared_gfs2/lv_shared_gfs2
Old journals: 2
New journals: 4
```

rhel8-gfs2-01 ~ # gfs2_edit -p jindex /dev/shared_gfs2/lv_shared_gfs2 | grep journal
```
  3/3 [fc7745eb] 1/18 (0x1/0x12) +0: File    journal0
  4/4 [8b70757d] 2/8231 (0x2/0x2027) +0: File    journal1
  5/5 [127924c7] 10/17237 (0xa/0x4355) +1: File    journal2
  6/6 [657e1451] 14/50334 (0xe/0xc49e) +1: File    journal3
```

→ Si no estuviesemos en un cluster pacemaker Suspensión/Reanudación:
rhel8-gfs2-01 ~ # dmsetup suspend /dev/shared_gfs2/lv_shared_gfs2
rhel8-gfs2-01 ~ # dmsetup resume /dev/shared_gfs2/lv_shared_gfs2

→ Resúmen:

rhel8-gfs2-01 ~ # pcs status --full
```
Cluster name: HA_gfs2
Cluster Summary:
  * Stack: corosync
  * Current DC: rhel8-gfs2-02 (2) (version 2.1.0-8.el8-7c3f660707) - partition with quorum
  * Last updated: Sun Jan  2 19:02:21 2022
  * Last change:  Sun Jan  2 18:51:48 2022 by hacluster via crmd on rhel8-gfs2-02
  * 2 nodes configured
  * 9 resource instances configured

Node List:
  * Online: [ rhel8-gfs2-01 (1) rhel8-gfs2-02 (2) ]

Full List of Resources:
```

```
  * Clone Set: locking-clone [locking]:
    * Resource Group: locking:0:
      * dlm       (ocf::pacemaker:controld):     Started rhel8-gfs2-02
      * lvmlockd          (ocf::heartbeat:lvmlockd):     Started rhel8-gfs2-02
    * Resource Group: locking:1:
      * dlm       (ocf::pacemaker:controld):     Started rhel8-gfs2-01
      * lvmlockd          (ocf::heartbeat:lvmlockd):     Started rhel8-gfs2-01
  * fence_all   (stonith:fence_xvm):   Started rhel8-gfs2-01
  * Clone Set: shared_gfs2-clone [shared_gfs2]:
    * Resource Group: shared_gfs2:0:
      * shared_lv-gfs2  (ocf::heartbeat:LVM-activate):         Started rhel8-gfs2-02
      * FS-GFS2_mount         (ocf::heartbeat:Filesystem):   Started rhel8-gfs2-02
    * Resource Group: shared_gfs2:1:
      * shared_lv-gfs2  (ocf::heartbeat:LVM-activate):         Started rhel8-gfs2-01
      * FS-GFS2_mount         (ocf::heartbeat:Filesystem):   Started rhel8-gfs2-01

Migration Summary:

Tickets:

PCSD Status:
  rhel8-gfs2-01: Online
  rhel8-gfs2-02: Online

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

→ Testing:

rhel8-gfs2-01 ~ # touch /mnt/gfs2/Desde-01.txt

rhel8-gfs2-01 ~ # ssh rhel8-gfs2-02 touch /mnt/gfs2/Desde-02.txt

rhel8-gfs2-01 ~ # ls /mnt/gfs2/

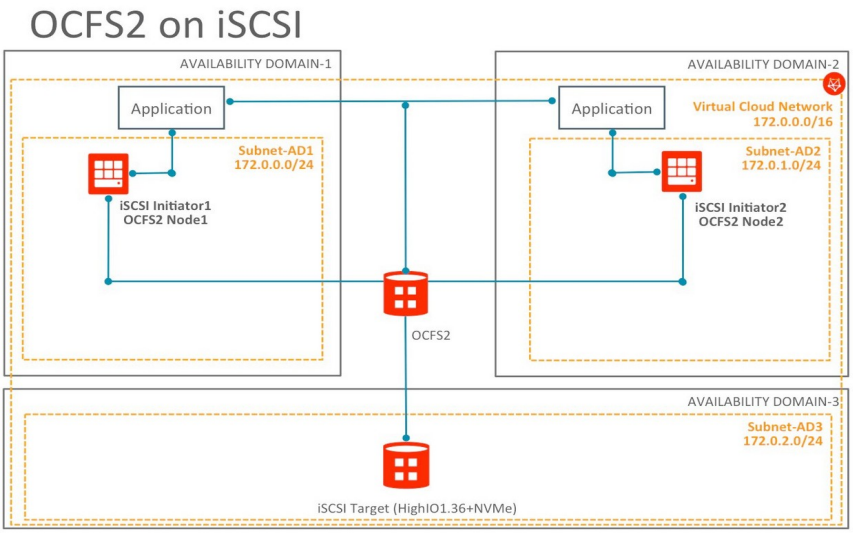
```
Desde-01.txt  Desde-02.txt
```

rhel8-gfs2-01 ~ # ssh rhel8-gfs2-02 ls /mnt/gfs2/

```
Desde-01.txt
Desde-02.txt
```

→ OCFS2

→ Modelo ejemplo de arquitectura a desplegar.



→ Conceptos teóricos previos.

Oracle Cluster File System versión 2 (OCFS2) es un sistema de archivos de disco compartido de uso general, alto rendimiento y alta disponibilidad diseñado para su uso en clústeres.

También es posible montar un volumen OCFS2 en un sistema independiente no agrupado.

Aunque pueda parecer que no hay ningún beneficio en montar ocfs2 localmente en comparación con sistemas de archivos alternativos como ext4 o btrfs, puede usar el comando reflink con OCFS2 para crear clones de copia en escritura de archivos individuales de una manera similar a usar el comando cp --reflink con el sistema de archivos btrfs.

Por lo general, estos clones le permiten ahorrar espacio en el disco al almacenar varias copias de archivos muy similares, como imágenes de máquinas virtuales o contenedores de Linux.

Además, montar un sistema de archivos OCFS2 local le permite migrarlo posteriormente a un sistema de archivos de clúster sin necesidad de conversión.

Tenga en cuenta que al usar el comando reflink, el sistema de archivos resultante se comporta como un clon del sistema de archivos original. Esto significa que sus UUID son idénticos.

Cuando use reflink para crear un clon, debe cambiar el UUID usando el comando tunefs.ocfs2. Consulte la Sección 7.3.10, “Consulta y cambio de parámetros de volumen” para obtener más información.

Casi todas las aplicaciones pueden utilizar OCFS2, ya que proporciona semántica del sistema de archivos local. Las aplicaciones que son compatibles con el clúster pueden utilizar E / S paralelas coherentes con la memoria caché de varios nodos del clúster para equilibrar la actividad en todo el clúster, o pueden utilizar la funcionalidad del sistema de archivos disponible para realizar una

conmutación por error y ejecutarse en otro nodo en caso de que se produzca un error, el nodo falla. Los siguientes ejemplos tipifican algunos casos de uso de OCFS2:

Oracle VM para alojar acceso compartido a imágenes de máquinas virtuales.
Oracle VM y VirtualBox para permitir que las máquinas invitadas de Linux compartan un sistema de archivos.
Oracle Real Application Cluster (RAC) en clústeres de bases de datos.
Oracle E-Business Suite en clústeres de middleware.
OCFS2 tiene una gran cantidad de características que lo hacen adecuado para su implementación en un entorno informático de nivel empresarial: Soporte para el registro de datos ordenado y de escritura diferida que proporciona consistencia al sistema de archivos en caso de falla de energía o falla del sistema. Tamaños de bloque que van desde 512 bytes a 4 KB, y tamaños de clúster del sistema de archivos que van desde 4 KB a 1 MB (ambos en incrementos de potencias de 2). El tamaño de volumen máximo admitido es de 16 TB, que corresponde a un tamaño de clúster de 4 KB. Un tamaño de volumen tan grande como 4 PB es teóricamente posible para un tamaño de clúster de 1 MB, aunque este límite no se ha probado. Asignaciones basadas en la extensión para un almacenamiento eficiente de archivos muy grandes. Soporte de asignación optimizado para archivos dispersos, datos en línea, extensiones no escritas, perforaciones, enlaces de referencia y reserva de asignación para un alto rendimiento y almacenamiento eficiente. Indexación de directorios para permitir un acceso eficiente a un directorio incluso si contiene millones de objetos. Sumas de comprobación de metadatos para la detección de inodos y directorios corruptos. Atributos extendidos para permitir que un número ilimitado de pares nombre: valor se adjunten a objetos del sistema de archivos, como archivos regulares, directorios y enlaces simbólicos. Soporte de seguridad avanzada para POSIX ACL y SELinux además del modelo tradicional de permisos de acceso a archivos. Soporte para cuotas de usuarios y grupos. Soporte para clústeres heterogéneos de nodos con una combinación de arquitecturas little-endian (x86, x86_64, ia64) y big-endian (ppc64) de 32 y 64 bits. Una pila de clúster en el núcleo (O2CB) fácil de configurar con un administrador de bloqueo distribuido (DLM), que administra el acceso simultáneo desde los nodos del clúster. Soporte para E / S en búfer, directas, asíncronas, empalmadas y mapeadas en memoria. Un conjunto de herramientas que utiliza parámetros similares al sistema de archivos ext3.
Para obtener el mejor rendimiento, cada nodo del clúster debe tener al menos dos interfaces de red. Una interfaz está conectada a una red pública para permitir el acceso general a los sistemas. La otra interfaz se utiliza para la comunicación privada entre los nodos; el latido del clúster que determina cómo los nodos del clúster coordinan su acceso a los recursos compartidos y cómo monitorean el estado de los demás. Esta interfaz debe conectarse a través de un conmutador de red. Asegúrese de que todas las interfaces de red estén configuradas y funcionando antes de continuar con la configuración del clúster. Tiene la opción de dos configuraciones de latido del clúster: Hilo de latido local para cada dispositivo compartido. En este modo, un nodo inicia un subproceso de latido cuando monta un volumen OCFS2 y detiene el subproceso cuando desmonta el volumen. Este es el modo de latido predeterminado. Existe una gran sobrecarga de CPU en los nodos que montan una gran cantidad de volúmenes OCFS2, ya que cada montaje requiere un subproceso de latido independiente. Una gran cantidad de montajes también aumenta el riesgo de que un nodo se separe del clúster debido a un tiempo de espera de E / S de latido en un solo montaje.

Latido global en dispositivos compartidos específicos. Puede configurar cualquier volumen OCFS2 como un dispositivo de latido global siempre que ocupe un dispositivo de disco completo y no una partición. En este modo, el latido del dispositivo comienza cuando el clúster se conecta y se detiene cuando el clúster se desconecta. Este modo se recomienda para clústeres que montan una gran cantidad de volúmenes OCFS2. Un nodo se separa del clúster si se agota el tiempo de espera de E / S de latido en más de la mitad de los dispositivos de latido globales. Para proporcionar redundancia contra la falla de uno de los dispositivos, debe configurar al menos tres dispositivos de latido global.

→ **Sincronización horaria** → **chronyd.service**

→ **Topología** → **Oracle8**

Oracle8

[iSCSI Target Server]		**[iSCSI Initiator's]**
ocfs2-target.cadilinea.lan	<====>	**ocfs2-01.cadilinea.lan → 192.168.10.161**
192.168.10.160		**ocfs2-02.cadilinea.lan → 192.168.10.162**

 # vim /etc/hosts

```
## OCFS2.
192.168.10.160  ocfs2-target.cadilinea.lan    ocfs2-target
192.168.10.161  ocfs2-01.cadilinea.lan        ocfs2-01
192.168.10.162  ocfs2-02.cadilinea.lan        ocfs2-02
```

→ **Desde** → **ocfs2-target**

```
ocfs2-target ~ # vim /etc/hosts
192.168.10.160  ocfs2-target.cadilinea.lan    ocfs2-target
192.168.10.161  ocfs2-01.cadilinea.lan        ocfs2-01
192.168.10.162  ocfs2-02.cadilinea.lan        ocfs2-02
```

→ **Instalación de iSCSI** → **target** → **ocfs2-target**

```
ocfs2-target ~ # firewall-cmd --permanent --add-port=3260/tcp
ocfs2-target ~ # firewall-cmd --reload

ocfs2-target ~ # lsblk /dev/vd[b-z]
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
vdb  251:16   0  20G  0 disk
vdc  251:32   0  20G  0 disk
vdd  251:48   0  20G  0 disk

ocfs2-target ~ # pvcreate /dev/vdb
ocfs2-target ~ # vgcreate vg_ocfs2 /dev/vdb
ocfs2-target ~ # lvcreate -n lv_ocfs2 -L 5G vg_ocfs2

ocfs2-target ~ # yum install targetcli
```

```
ocfs2-target ~ # targetcli
Warning: Could not load preferences file /root/.targetcli/prefs.bin.
targetcli shell version 2.1.53
Copyright 2011-2013 by Datera, Inc and others.
For help on commands, type 'help'.

/> ls
o- / ..... [...]
  o- backstores ..... [...]
  | o- block ..... [Storage Objects: 0]
  | o- fileio ..... [Storage Objects: 0]
  | o- pscsi ..... [Storage Objects: 0]
  | o- ramdisk ..... [Storage Objects: 0]
  o- iscsi ..... [Targets: 0]
  o- loopback ..... [Targets: 0]
  o- vhost ..... [Targets: 0]
/> cd backstores/block
/backstores/block> create name=ocfs2 dev=/dev/vg_ocfs2/lv_ocfs2
Created block storage object ocfs2 using /dev/vg_ocfs2/lv_ocfs2.
/backstores/block> cd /iscsi
/iscsi> create iqn.2021-05.lan.cadilinea:target
Created target iqn.2021-05.lan.cadilinea:target.
Created TPG 1.
Global pref auto_add_default_portal=true
Created default portal listening on all IPs (0.0.0.0), port 3260.
/iscsi> cd /iscsi/iqn.2021-05.lan.cadilinea:target/tpg1/acls
/iscsi/iqn.20...get/tpg1/acls> create iqn.2021-05.lan.cadilinea:target
Created Node ACL for iqn.2021-05.lan.cadilinea:target
/iscsi/iqn.20...get/tpg1/acls> cd /iscsi/iqn.2021-05.lan.cadilinea:target/tpg1/
/iscsi/iqn.20...a:target/tpg1> set attribute authentication=0
Parameter authentication is now '0'.
/iscsi/iqn.20...a:target/tpg1> set attribute generate_node_acls=1
Parameter generate_node_acls is now '1'.
/iscsi/iqn.20...a:target/tpg1> cd /iscsi/iqn.2021-05.lan.cadilinea:target/tpg1/luns
/iscsi/iqn.20...get/tpg1/luns> pwd
/iscsi/iqn.2021-05.lan.cadilinea:target/tpg1/luns
/iscsi/iqn.20...get/tpg1/luns> create /backstores/block/ocfs2
Created LUN 0.
Created LUN 0->0 mapping in node ACL iqn.2021-05.lan.cadilinea:ocfs2
/iscsi/iqn.20...get/tpg1/luns> cd /
/> ls
o- / ..... [...]
  o- backstores ..... [...]
  | o- block ..... [Storage Objects: 1]
  | | o- ocfs2 ..... [/dev/vg_ocfs2/lv_ocfs2 (5.0GiB) write-thru
activated]
  | |   o- alua ..... [ALUA Groups: 1]
```

```
  | |    o- default_tg_pt_gp ..... [ALUA state:
Active/optimized]
  | o- fileio ..... [Storage Objects: 0]
  | o- pscsi ..... [Storage Objects: 0]
  | o- ramdisk ..... [Storage Objects: 0]
  o- iscsi ..... [Targets: 1]
  | o- iqn.2021-05.lan.cadilinea:target ..... [TPGs: 1]
  |   o- tpg1 ..... [gen-acls, no-auth]
  |     o- acls ..... [ACLs: 1]
  |     | o- iqn.2021-05.lan.cadilinea:target ..... [Mapped
LUNs: 1]
  |     |   o- mapped_lun0 ..... [lun0 block/ocfs2 (rw)]
  |     o- luns ..... [LUNs: 1]
  |     | o- lun0 ..... [block/ocfs2 (/dev/vg_ocfs2/lv_ocfs2)
(default_tg_pt_gp)]
  |     o- portals ..... [Portals: 1]
  |       o- 0.0.0.0:3260 ..... [OK]
  o- loopback ..... [Targets: 0]
  o- vhost ..... [Targets: 0]
/> saveconfig
Configuration saved to /etc/target/saveconfig.json
/> exit
Global pref auto_save_on_exit=true
Last 10 configs saved in /etc/target/backup/.
Configuration saved to /etc/target/saveconfig.json
```

→ **Desde** → **ocfs2-01 / ocfs2-02**

```
ocfs2-0X ~ # dnf install iscsi-initiator-utils
ocfs2-0X ~ # iscsiadm -m discovery -t st -p 192.168.10.160
192.168.10.160:3260,1 iqn.2021-05.lan.cadilinea:target

ocfs2-0X ~ # vim /etc/iscsi/initiatorname.iscsi
#InitiatorName=iqn.1988-12.com.oracle:26a09b814d27
InitiatorName=iqn.2021-05.lan.cadilinea:target

ocfs2-0X ~ # systemctl enable --now iscsid.service

ocfs2-0X ~ # iscsiadm -m node -T iqn.2021-05.lan.cadilinea:target -p 192.168.10.160 -l
Logging in to [iface: default, target: iqn.2021-05.lan.cadilinea:target, portal: 192.168.10.160,3260]
Login to [iface: default, target: iqn.2021-05.lan.cadilinea:target, portal: 192.168.10.160,3260]
successful.

ocfs2-0X ~ # fdisk -l |grep sd[a-z]
Disco /dev/sda: 5 GiB, 5368709120 bytes, 10485760 sectores
```

→ **OCFS2** →**initiators** → **ocfs2-01 / ocfs2-02**

```
ocfs2-0X ~ # dnf install ocfs2-tools
ocfs2-0X ~ # firewall-cmd --permanent --add-port=7777/{tcp,udp}
ocfs2-0X ~ # firewall-cmd --reload

ocfs2-0X ~ # o2cb add-cluster HAocfs2
ocfs2-0X ~ # o2cb add-node HAocfs2 ocfs2-01 --ip 192.168.10.161
ocfs2-0X ~ # o2cb add-node HAocfs2 ocfs2-02 --ip 192.168.10.162

ocfs2-0X ~ # cat /etc/ocfs2/cluster.conf
cluster:
        heartbeat_mode = local
        node_count = 2
        name = HAocfs2

node:
        number = 0
        cluster = HAocfs2
        ip_port = 7777
        ip_address = 192.168.10.161
        name = ocfs2-01

node:
        number = 1
        cluster = HAocfs2
        ip_port = 7777
        ip_address = 192.168.10.162
        name = ocfs2-02

ocfs2-0X ~ # /sbin/o2cb.init configure
Configuring the O2CB driver.

This will configure the on-boot properties of the O2CB driver.
The following questions will determine whether the driver is loaded on
boot.  The current values will be shown in brackets ('[]').  Hitting
<ENTER> without typing an answer will keep that current value.  Ctrl-C
will abort.

Load O2CB driver on boot (y/n) [n]: y
Cluster stack backing O2CB [o2cb]:
Cluster to start on boot (Enter "none" to clear) [ocfs2]: HAocfs2
Specify heartbeat dead threshold (>=7) [31]:
Specify network idle timeout in ms (>=5000) [30000]:
Specify network keepalive delay in ms (>=1000) [2000]:
Specify network reconnect delay in ms (>=2000) [2000]:
Writing O2CB configuration: OK
checking debugfs...
```

```
Loading stack plugin "o2cb": OK
Loading filesystem "ocfs2_dlmfs": OK
Creating directory '/dlm': OK
Mounting ocfs2_dlmfs filesystem at /dlm: OK
Setting cluster stack "o2cb": OK
Registering O2CB cluster "HAocfs2": OK
Setting O2CB cluster timeouts : OK
```

```
ocfs2-0X ~ # /sbin/o2cb.init status
Driver for "configfs": Loaded
Filesystem "configfs": Mounted
Stack glue driver: Loaded
Stack plugin "o2cb": Loaded
Driver for "ocfs2_dlmfs": Loaded
Filesystem "ocfs2_dlmfs": Mounted
Checking O2CB cluster "HAocfs2": Online
  Heartbeat dead threshold: 31
  Network idle timeout: 30000
  Network keepalive delay: 2000
  Network reconnect delay: 2000
  Heartbeat mode: Local
Checking O2CB heartbeat: Not active
Debug file system at /sys/kernel/debug: mounted
```

```
ocfs2-0X ~ # systemctl enable --now ocfs2.service o2cb.service
```

```
ocfs2-0X ~ # vim /etc/sysctl.conf
kernel.panic=30
kernel.panic_on_oops=1
```

```
ocfs2-0X ~ # sysctl -p
kernel.panic = 30
kernel.panic_on_oops = 1
```

==> Starting and Stopping the Cluster Stack <==

COMMAND	DESCRIPTION
/sbin/o2cb.init status	Check the status of the cluster stack.
/sbin/o2cb.init online	Start the cluster stack.
/sbin/o2cb.init offline	Stop the cluster stack.
/sbin/o2cb.init unload	Unload the cluster stack.

```
#  mkfs.ocfs2
usage: mkfs.ocfs2 [-b block-size] [-C cluster-size] [-J journal-options]
                  [-L volume-label] [-M mount-type] [-N number-of-node-slots]
                  [-T filesystem-type] [-U uuid][-HFnqvV] [--dry-run]
```

[--fs-feature-level=[default|max-compat|max-features]]
[--fs-features=[[no]sparse,...]] [--global-heartbeat]
[--cluster-stack=stackname] [--cluster-name=clustername]
[--no-backup-super] device [blocks-count]

Ejemplo:

 # mkfs.ocfs2 -b 4K -C 4K -J size=4M -N 4 -L HAocfs2 --cluster-name=HAocfs2 --cluster-stack=o2cb --global-heartbeat /dev/sda

ocfs2-01 ~ # mkfs.ocfs2 -L "HAocfs2" /dev/sda
mkfs.ocfs2 1.8.6
Cluster stack: classic o2cb
Label: **HAocfs2**
Features: sparse extended-slotmap backup-super unwritten inline-data strict-journal-super xattr indexed-dirs refcount discontig-bg
Block size: 4096 (12 bits)
Cluster size: 4096 (12 bits)
Volume size: 5368709120 (1310720 clusters) (1310720 blocks)
Cluster groups: 41 (tail covers 20480 clusters, rest cover 32256 clusters)
Extent allocator size: 4194304 (1 groups)
Journal size: 67108864
Node slots: 4
Creating bitmaps: done
Initializing superblock: done
Writing system files: done
Writing superblock: done
Writing backup superblock: 2 block(s)
Formatting Journals: done
Growing extent allocator: done
Formatting slot map: done
Formatting quota files: done
Writing lost+found: done
mkfs.ocfs2 successful

ocfs2-0X ~ # mkdir /mnt/ocfs2
ocfs2-0X ~ # mount.ocfs2 /dev/sda /mnt/ocfs2/
ocfs2-0X ~ # df -hT /dev/sda
S.ficheros Tipo Tamaño Usados Disp Uso% Montado en
/dev/sda ocfs2 5,0G 279M 4,8G 6% /mnt/ocfs2

ocfs2-01 ~ # mounted.ocfs2 -f
Device Stack Cluster F Nodes
/dev/sda o2cb ocfs2-02, ocfs2-01

ocfs2-01 ~ # mounted.ocfs2 -d
Device Stack Cluster F UUID Label
/dev/sda o2cb 5FBB174992F049FF83EB0CEB5E6B1DED HAocfs2

→ **Montaje Definitivo.**

ocfs2-0X ~ # vim /etc/fstab
…
LABEL=**HAocfs2** /mnt/ocfs2 ocfs2 _netdev,defaults 0 0

ocfs2-01 ~ # o2image /dev/sda /tmp/sda.img
ocfs2-01 ~ # tunefs.ocfs2 -Q "Label = %V\nUUID = %U\nNumSlots =%N\n" /dev/sda
Label = HAocfs2
UUID = 5FBB174992F049FF83EB0CEB5E6B1DED
NumSlots =4

ocfs2-01 ~ # o2info /dev/sda --volinfo
Label: **HAocfs2**
UUID: 5FBB174992F049FF83EB0CEB5E6B1DED
Block Size: 4096
Cluster Size: 4096
Node Slots: 4
Features: backup-super strict-journal-super sparse extended-slotmap
Features: inline-data xattr indexed-dirs refcount discontig-bg unwritten

ocfs2-01 ~ # o2info /dev/sda --mkfs
-N 4 -J size=67108864 -b 4096 -C 4096 --fs-features backup-super,strict-journal-super,sparse,extended-slotmap,inline-data,xattr,indexed-dirs,refcount,discontig-bg,unwritten -L HAocfs2

→ **Ampliación del volúmen lógico → ocfs-target.**

ocfs2-target ~ # lvs /dev/vg_ocfs2/lv_ocfs2
LV VG Attr LSize Pool Origin Data% Meta% Move Log Cpy%Sync Convert
lv_ocfs2 vg_ocfs2 -wi-ao---- 5,00g

ocfs2-01 ~ # lsblk /dev/sda
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda 8:0 0 5G 0 disk /mnt/ocfs2
ocfs2-02 ~ # lsblk /dev/sda
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda 8:0 0 5G 0 disk /mnt/ocfs2

ocfs2-01 ~ # umount /mnt/ocfs2
ocfs2-02 ~ # umount /mnt/ocfs2

→ **Ampliamos +5G desde el target para su lvs.**

ocfs2-target ~ # lvextend -L +5G /dev/vg_ocfs2/lv_ocfs2
Size of logical volume vg_ocfs2/lv_ocfs2 changed from 5,00 GiB (1280 extents) to 10,00 GiB

(2560 extents).
  Logical volume vg_ocfs2/lv_ocfs2 successfully resized.

ocfs2-target ~ # lvs /dev/vg_ocfs2/lv_ocfs2
  LV   VG   Attr   LSize Pool Origin Data% Meta% Move Log Cpy%Sync Convert
  lv_ocfs2 vg_ocfs2 -wi-ao---- 10,00g

→ Reacondicionamos el FS, y → solo desde un nodo.
(Posiblemente NO HAGA FALTA hacerlo !.).

ocfs2-01 ~ # tunefs.ocfs2 -S /dev/sda ## -S|--volume-size
ocfs2-01 ~ # lsblk /dev/sda
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda   8:0   0 10G 0 disk

ocfs2-01 ~ # fsck.ocfs2 -fn /dev/sda
fsck.ocfs2 1.8.6
Checking OCFS2 filesystem in /dev/sda:
  Label: HAocfs2
  UUID: 5FBB174992F049FF83EB0CEB5E6B1DED
  Number of blocks: 2621440
  Block size: 4096
  Number of clusters: 2621440
  Cluster size: 4096
  Number of slots: 4

** Skipping slot recovery because -n was given. **
/dev/sda was run with -f, check forced.
Pass 0a: Checking cluster allocation chains
Pass 0b: Checking inode allocation chains
Pass 0c: Checking extent block allocation chains
Pass 1: Checking inodes and blocks
Pass 2: Checking directory entries
Pass 3: Checking directory connectivity
Pass 4a: Checking for orphaned inodes
Pass 4b: Checking inodes link counts
All passes succeeded.

ocfs2-01 ~ # mount -a
ocfs2-01 ~ # lsblk /dev/sda
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda   8:0   0 10G 0 disk /mnt/ocfs2

ocfs2-01 ~ # mounted.ocfs2 -f
Device   Stack Cluster F Nodes
/dev/sda o2cb   ocfs2-01, ocfs2-02

→ AFS (Andrew File System).

El **Andrew File System** (*Sistema de archivos Andrew*), o **AFS**1 es un sistema de archivos distribuido a través de la red que fue desarrollado como parte del proyecto Andrew por la Universidad Carnegie Mellon.2 Su nombre proviene de Andrew Carnegie y Andrew Mellon. Es utilizado fundamentalmente en entornos de computación distribuida.

Índice

Características

AFS3 tiene varios beneficios sobre los sistemas de archivos en red tradicionales, en particular en áreas de seguridad y escalabilidad. Es frecuente que los despliegues de AFS en empresas lleguen a tener más de 25.000 clientes.4 AFS usa Kerberos como mecanismo de autenticación, e implementa listas de control de acceso en directorios para usuarios y grupos. Cada cliente mantiene una caché en el sistema de archivos local para aumentar la velocidad de acceso a los archivos. Esto también permite el acceso limitado al sistema de archivos en el caso de caída del servidor o un fallo de red.

AFS utiliza el modelo de baja consistencia.5 En AFS los archivos son cacheados bajo demanda en las estaciones locales, las cuales son lo suficientemente grandes como para almacenar un gran número de archivos.

La estrategia en la que se basa la utilización de estas cachés se fundamenta en un uso normal de archivos. Donde por norma general un usuario suele tratar con archivos pequeños, de acceso secuencial y en modo lectura.

De igual modo hay que resaltar que las bases de datos son el único tipo de archivo que debido a su concurrencia y continua actualización por parte de los usuarios se excluyen del diseño de AFS.

Funcionamiento básico

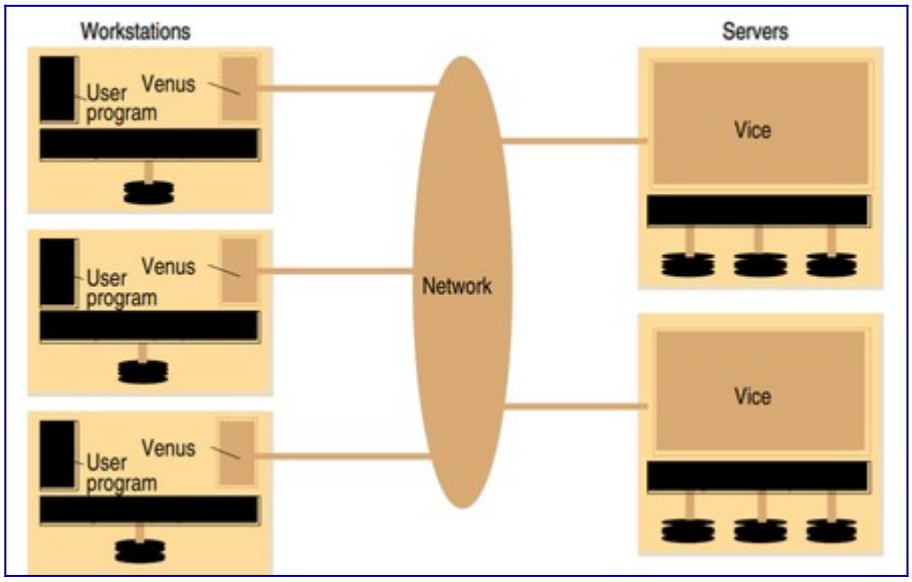
1. Cuando un usuario en un equipo cliente realiza una llamada open para la obtención de un archivo compartido y no hay una copia actual de dicho archivo en la caché local, el servidor que contiene el archivo lo busca y le envía una copia del mismo.

2. La copia se almacena en el sistema local de la máquina cliente para posteriormente abrirse y devolver un *descriptor* a la aplicación del cliente.
3. Se realizan las operaciones de lectura, escritura y/u otras operaciones posteriores en el archivo, las cuales se aplicarán a la copia local.
4. Cuando el proceso en el cliente emite una llamada *close*, si la copia local ha sido actualizada sus contenidos se envían de vuelta al servidor. El servidor por su parte actualiza el archivo y las marcas de tiempo y mantiene la copia en el disco local del cliente por si se realiza una nueva solicitud del archivo.

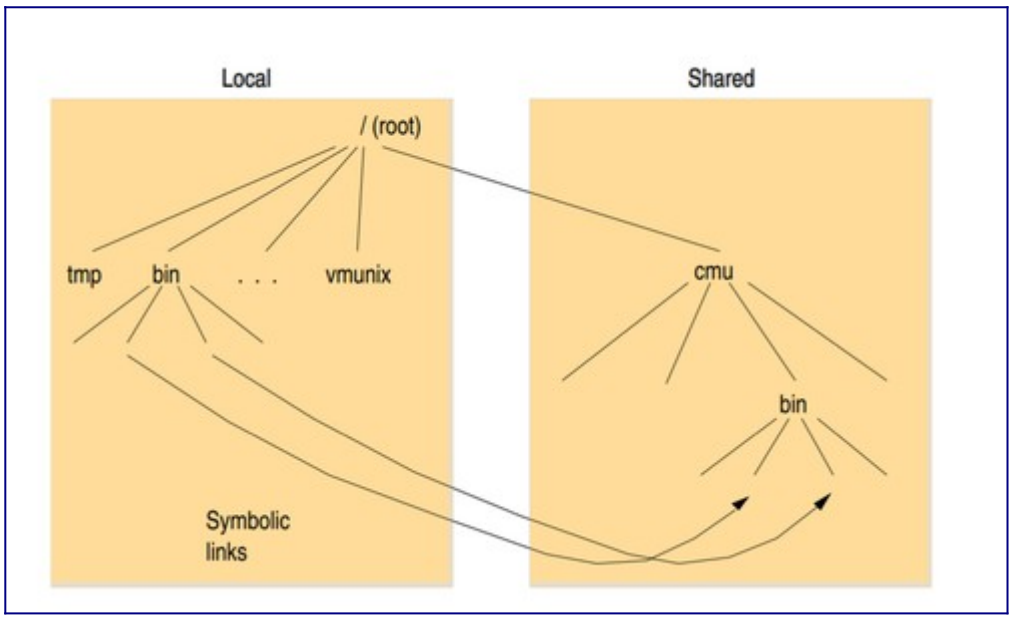
Arquitectura

La arquitectura de AFS está implementada como dos componentes software, **Vice** y **Venus**, y la red que las comunica.

- Vice: servicio software que se ejecuta como un proceso Unix a nivel de usuario en la parte del servidor.
- Venus: proceso de usuario que actúa en la parte del cliente.



Volumen de archivos



Una característica importante de AFS es el **volumen**, un árbol de archivos y subdirectorios. Los volúmenes los crean los administradores y los enlazan con una ruta específica en una celda AFS. Una vez ha sido creado, los usuarios del sistema de archivos pueden crear directorios y archivos de manera normal sin tener en cuenta donde se encuentra físicamente el volumen. Un volumen puede tener una cuota asignada para limitar la cantidad de espacio consumido. Según las necesidades, los administradores de AFS pueden mover ese volumen a otro servidor y otra localización en disco sin la necesidad de notificar a los usuarios de dicho cambio; de hecho esta operación puede realizarse mientras se están usando los archivos dentro del volumen.

Los volúmenes pueden ser replicados para copias de respaldo de sólo lectura. Cuando se accede a archivos en un volumen de sólo lectura, un sistema cliente obtendrá datos de una copia de sólo lectura particular. Si en algún punto esa copia deja de estar disponible, el cliente buscará otra de las copias restantes. De nuevo, los usuarios de esos datos se despreocupan sobre la localización física de esta copia; los administradores pueden crear y recolocar tales copias según las necesidades. La suite de comandos AFS garantiza que todos los volúmenes de sólo lectura contienen copias iguales del volumen original de lectura-escritura en el momento que se creó la copia de sólo lectura.

El espacio de nombres de archivos en una estación de trabajo Andrew es particionada en dos espacios: el espacio de nombre compartido y el local. El espacio de nombres compartido es idéntico en todas las estaciones y se encuentra bajo el directorio "*/cmu*". Por otra parte, el espacio local es único para cada estación. Contiene archivos temporales necesarios para la inicialización de la estación y enlaces simbólicos a los archivos que se encuentran en el espacio de nombres compartido.

Sistema de llamadas

El sistema de llamadas se fundamenta en el uso de llamadas del tipo "callback" y "callback promise" entre Venus y Vice para la transferencia de archivos entre el servidor y los clientes

Proceso de Usuario	Kernel de Unix	Venus	Net	Vice
open(nombreFichero,modo)	Si nombreFichero hace referencia a un archivo que se encuentra en un espacio compartido realiza una solicitud a Venus. Abre el archivo local y devuelve el descriptor de la solicitud.	Busca en la lista de archivos de caché local si se encuentra el archivo deseado. Si no se encuentra o el valor de *callback promise* no es válido se realiza una solicitud del archivo al servidor Vice que lo contenga. Establece la copia del archivo en los archivos locales del sistema, adjunta su nombre en la caché local y devuelve el nombre local a UNIX.		Envía una copia del archivo y una callback promise al equipo que la solicitó. Registra la callback promise.
read (nombreDescriptor, Buffer, longitud)	Realiza una operación de lectura en UNIX de la copia local del archivo.			
write (nombreDescriptor, Buffer, longitud)	Realiza una operación de escritura en UNIX de la copia local del archivo.			
close(nombreDescriptor)	Cierra la copia local y notifica a Venus de que el archivo ha sido cerrado.	Si la copia ha cambiado se envía una copia al servidor Vice donde se encuentra el archivo.		Reemplaza el archivo y envía callback a los demás clientes que tienen callback promises de ese archivo.

Cuando Vice proporciona una copia de un archivo a Venus también le proporciona una *callback promise*, la cual es un token emitido por el servidor Vice que almacena el archivo y garantiza que notificará al proceso Venus cuando cualquier otro cliente modifique el archivo.

Las variables *callback promise* son almacenadas junto con los archivos en la caché del equipo cliente y tienen dos estados: válido o cancelado. Cuando un servidor realiza una solicitud para actualizar un archivo, notifica a todos los procesos Venus que actualmente tienen *callback promises* mediante el envío de una llamada *callback* para que establezcan su token (*callback promise*) a estado cancelado. Por otra parte si Vice sirve el archivo a Venus el estado de *callback promise* sería valido.

Cada vez que Venus realiza un *open* de un archivo en el cliente busca si dicho archivo se encuentra en cache y en caso afirmativo verificar su token. Si el valor del token es cancelado se debe obtener una nueva copia del archivo. Pero si el token es válido, entonces la copia en caché se puede abrir y usar sin hacer referencia a Vice.

Si un cliente falla, se reinicia o sufre un apagado Venus puede retener tantos archivos de caché como admita el disco local, pero no puede saber si los valores de las *callback promise* siguen siendo válidos. De modo, que antes de acceder a cada uno de los archivos que contiene en caché debe validar con cada servidor que contiene dicho archivo la marca de tiempo de modificación de cada uno de ellos. De esta forma se puede saber que si la marca de tiempo es la actual debe restablecer el valor del token a válido. Si por el contrario la marca de tiempo es inferior, implica que el archivo está desactualizado y el servidor responderá con el valor del token a cancelado.

Implementaciones

Hay tres grandes implementaciones de AFS: Transarc (IBM), OpenAFS y Arla, aunque Transarc ya no tiene soporte y está desatendido. AFS (versión dos) también es el predecesor del sistema de archivos Coda.

Existe una cuarta implementación en el **código fuente** de **Linux** desde la versión 2.6.10.**6** Enviada por **Red Hat**, es una implementación muy sencilla que todavía se encuentra en las primeras fases de desarrollo y por tanto incompleta a fecha de enero de 2013.**7**

→ LUSTRE FS.

Lustre* is an open-source, global single-namespace, POSIX-compliant, distributed parallel file system designed for scalability, high-performance, and high-availability. Lustre runs on Linux-based operating systems and employs a client-server network architecture. Storage is provided by a set of servers that can scale to populations measuring up to several hundred hosts. Lustre servers for a single file system instance can, in aggregate, present up to tens of petabytes of storage to thousands of compute clients, with more than a terabyte-per-second of combined throughput.

Lustre is a file system that scales to meet the requirements of applications running on a range of systems from small-scale HPC environments up to the very largest supercomputers and has been created using object-based storage building blocks to maximize scalability.

Redundant servers support storage fail-over, while metadata and data are stored on separate servers,

allowing each file system to be optimized for different workloads. Lustre can deliver fast IO to applications across high-speed network fabrics, such as Intel® Omni-Path Architecture (OPA), InfiniBand* and Ethernet.

BIBLIOGRAFIA:

https://access.redhat.com/documentation/es-es/red_hat_enterprise_linux/8/html/configuring_gfs2_file_systems/index

https://www.server-world.info/en/note?os=CentOS_7&p=pacemaker&f=3

https://www.lisenet.com/2016/o2cb-cluster-with-dual-primary-drbd-and-ocfs2-on-oracle-linux-7/

https://crmsh.github.io/download/

https://wiki.clusterlabs.org/wiki/Dual_Primary_DRBD_%2B_OCFS2

https://github.com/LINBIT/linbit-documentation/blob/master/UG8.4/en/ocfs2.adoc

https://docs.oracle.com/en/operating-systems/oracle-linux/7/fsadmin/ol7-ocfs2.html#ol7-create-ocfs2
https://blogs.oracle.com/cloud-infrastructure/a-simple-guide-to-oracle-cluster-file-system-ocfs2-using-iscsi-on-oracle-cloud-infrastructure
https://ccia.esei.uvigo.es/docencia/CDA/1819/practicas/ejercicio-iscsi/

https://es.wikipedia.org/wiki/Andrew_File_System

https://www.openafs.org/

https://www.golinuxcloud.com/what-is-fencing-configure-kvm-cluster-fencing/

https://wiki.lustre.org/Category:Lustre_Systems_Administration

https://wiki.lustre.org/Introduction_to_Lustre

https://gist.github.com/dleske/743f9dafc212b7bb0edce370e961b99e

https://kojiwell.github.io/blog/2018/02/27/lustre-on-centos72

https://www.unixarena.com/2016/01/rhel7-configuring-gfs2-on-pacemakercorosync-cluster.html/
https://www.server-world.info/en/note?os=CentOS_7&p=pacemaker&f=3

https://www.lisenet.com/2018/libvirt-fencing-on-a-physical-kvm-host/
https://www.headdesk.me/Redhat_Cluster_on_EL7/8
https://www.golinuxcloud.com/what-is-fencing-configure-kvm-cluster-fencing/

https://www.lisenet.com/2016/activeactive-high-availability-pacemaker-cluster-with-gfs2-and-iscsi-shared-storage-on-centos-7/