Documentos

ISSN 1677-9274
Dezembro, 2016
141

Desvendando a infraestrutura de TI nos bastidores da implementação de cenários de virtualização utilizando o Xen

Embrapa

ISSN 1677-9274
Dezembro, 2016

Empresa Brasileira de Pesquisa Agropecuária
Embrapa Informática Agropecuária
Ministério da Agricultura, Pecuária e Abastecimento

Documentos 141

Desvendando a infraestrutura de TI nos bastidores da implementação de cenários de virtualização utilizando o Xen

Dácio Miranda Ferreira

Embrapa Informática Agropecuária
Campinas, SP
2016

Embrapa Informática Agropecuária
Av. André Tosello, 209 - Barão Geraldo
Caixa Postal 6041 - 13083-886 - Campinas, SP
Fone: (19) 3211-5700
www.embrapa.br/informatica-agropecuaria
SAC: www.embrapa.br/fale-conosco/sac/

1ª edição
publicação digitalizada 2016

Dados Internacionais de Catalogação na Publicação (CIP)
Embrapa Informática Agropecuária

Ferreira, Dácio Miranda.

Desvendando a infraestrutura de TI nos bastidores da implementação de cenários de virtualização utilizando o Xen / Dácio Miranda Ferreira.- Campinas : Embrapa Informática Agropecuária, 2016.

53 p. : il. ; cm. - (Documentos / Embrapa Informática Agropecuária, ISSN 1677-9274; 141).

1. Tecnologia da informação. 2. Virtualização. 3. Cluster. 4. Alta disponibilidade. I. Embrapa Informática Agropecuária. II. Título. III. Série.

CDD 006.6 (21.ed.)

Autor

Dácio Miranda Ferreira

Bacharel em Ciência da Computação, mestre em Informática em Saúde
Analista da Embrapa Informática Agropecuária, Campinas, SP

Apresentação

Este documento visa disseminar na Embrapa todo o conhecimento adquirido na Embrapa Informática Agropecuária para implementação de uma solução de virtualização, considerando desde um cenário mais simples, com o uso de apenas um servidor físico de grande porte, até o mais complexo, que envolve a utilização de diversos servidores físicos em c*luster, switches* e *storages* de discos.

Serão discutidas diversas tecnologias de TI que juntas serão capazes de atender as necessidades para implementação de cada cenário apresentado e que podem ser utilizadas até mesmo em outras implementações diferentes da virtualização, como por exemplo, para expansão da capacidade de armazenamento de um servidor.

São muitos os benefícios já conhecidos em utilizar uma solução de virtualização, como por exemplo, economia de energia, otimização do espaço físico, etc. Além disso, este tipo de solução oferece uma enorme flexibilidade ao profissional de TI, principalmente para o responsável pela infraestrutura de redes e servidores, que será capaz de oferecer o recurso computacional ao usuário na medida certa e de forma rápida e eficiente.

Entender os bastidores da implementação da solução oferecerá aos profissionais de TI a possibilidade de quebrar barreiras e desmistificar questões relacionadas a uma melhor utilização dos recursos computacionais em uma infraestrutura de grande porte para armazenamento e processamento de dados.

Silvia Maria Fonseca Silveira Massruhá
Chefe-geral
Embrapa Informática Agropecuária

Sumário

Desvendando a infraestrutura de TI nos bastidores da implementação de cenários de virtualização utilizando o Xen

Dácio Miranda Ferreira

Introdução

A virtualização é uma tecnologia bastante utilizada e disseminada entre os profissionais de Tecnologia da Informação (TI) devido aos diversos benefícios que ela oferece tais como: melhor aproveitamento dos recursos de hardware, economia de energia elétrica, otimização do espaço físico do data center, otimização dos serviços de administração dos sistemas, etc.

Diversos cenários, desde os mais simples até os mais complexos, podem ser propostos para se obter um ambiente virtualizado e, à medida que a complexidade aumenta, também aumentam as tecnologias que devem ser utilizadas para prover um ambiente confiável e funcional.

Dentre os cenários mais simples, podemos destacar a utilização de um servidor físico de grande porte capaz de hospedar diversos servidores virtuais, enquanto num ambiente mais complexo, podemos destacar o uso de diversos servidores físicos em *cluster* capazes de hospedar um grande número de servidores virtuais que podem ser migrados de um servidor físico para o outro, mantendo a disponibilidade dos sistemas.

Neste material, apresenta-se na prática 3 cenários de virtualização e as tecnologias envolvidas na construção de cada um deles, desta forma pretende-se desvendar os bastidores de uma infraestrutura deste tipo. Os cenários apresentados serão:

1. Um servidor físico de grande porte capaz de hospedar diversos servidores virtuais. Os dados dos servidores virtuais serão armazenados no disco local do servidor físico.

2. Um servidor físico de grande porte conectado a um *storage*. Os dados dos servidores virtuais serão armazenados no *storage*.

3. Diversos servidores físicos conectados em cluster. Os dados dos servidores virtuais serão armazenados em um *storage* compartilhado entre os servidores físicos.

Diversas tecnologias serão apresentadas em cada cenário, dentre elas destacam-se: Xen Hypervisor, LVM, iSCSI, Multipath, RAID, Corosync, Pacemaker, etc.

Os cenários

Nesta seção apresenta-se uma discussão de cada um dos três cenários propostos.

Servidor físico *standalone*

Neste cenário será utilizado apenas um servidor físico de grande porte, com grande capacidade de armazenamento, processamento e memória. Os servidores virtuais que serão criados ficarão armazenados no disco local do servidor físico e não haverá possibilidade de migração dos servidores virtuais. As tecnologias envolvidas neste cenário serão o Xen Hypervisor e LVM.

Servidor físico conectado a um *storage*

Neste cenário também será utilizado apenas um servidor físico de grande porte, com grande capacidade principalmente de processamento e memória. Em relação ao armazenamento, não é necessário que o servidor físico possua discos locais de grande capacidade (o ideal é que ele possua algum esquema de redundância dos discos locais, em RAID1, por exemplo), pois ele será conectado a um *storage*, e este sim vai fornecer a área de armazenamento onde serão alocados os dados dos servidores virtuais que serão criados.

A Figura 1 mostra como será a configuração de rede entre o servidor e o *storage* e as conexões físicas existentes entre eles. Neste cenário serão utilizadas as seguintes tecnologias: Xen, LVM, iSCSI e Multipath.

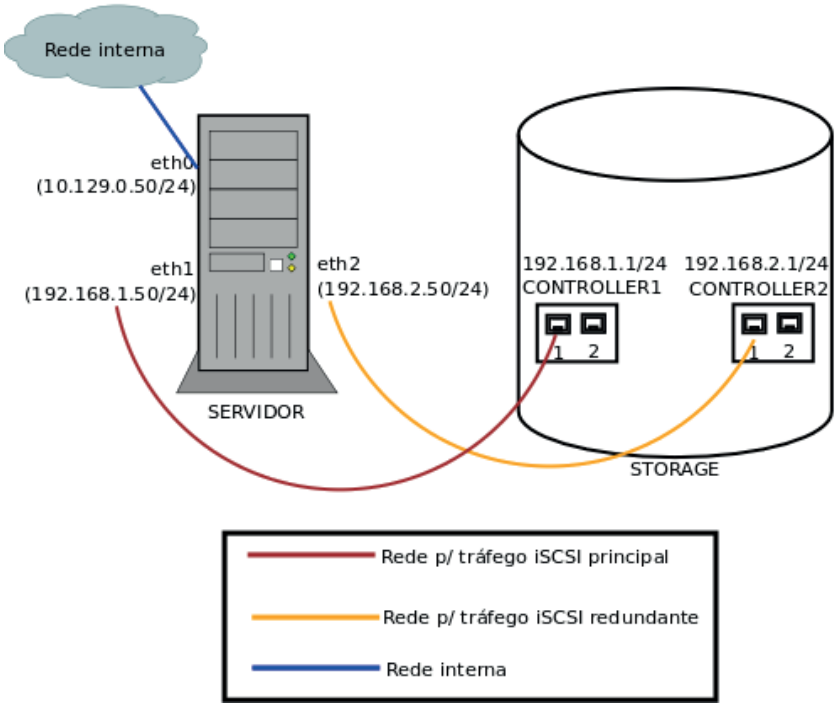

Figura 1. Cenário com um servidor físico conectado ao *storage*.

Cluster de servidores físicos conectados a um *storage*

Neste cenário serão utilizados 4 (ou mais) servidores físicos em *cluster* com as mesmas características do cenário anterior. É importante que os processadores dos servidores sejam pelo menos da mesma família, pois, como haverá a possibilidade de migração a quente dos servidores virtuais, o uso de processadores de famílias diferentes pode gerar algum prejuízo para o servidor virtual, uma vez que algumas instruções podem existir em uma família de processadores, mas pode não existir em outras, o que causaria o travamento do servidor virtual em uma eventual migração.

Assim como no cenário anterior, os dados dos servidores virtuais ficarão armazenados no *storage*, enquanto os servidores físicos terão que se comunicar entre si para poderem estabelecer o cluster e gerenciar os recursos que serão executados neles.

A Figura 2 mostra como será a configuração de rede dos servidores e do

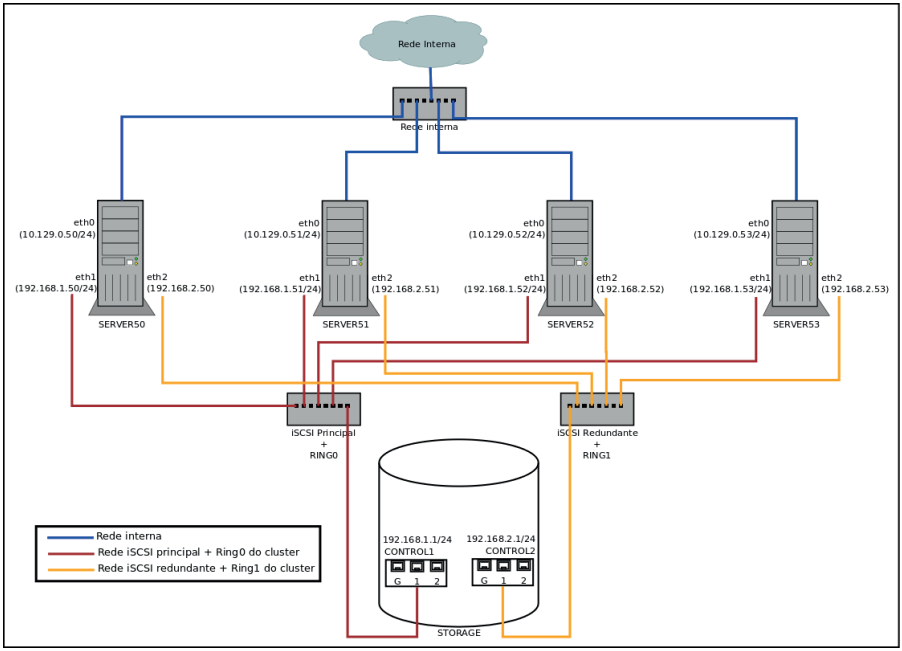

Figura 2. Cenário com um *cluster* de servidores físicos conectados ao *storage*.

storage e as conexões físicas existentes entre eles. Neste cenário serão utilizados dois *switchs* (um principal e outro redundante) uma vez que o *storage* não dispõe de portas suficientes para conexão direta entre ele e os 4 servidores físicos pertencentes ao cluster.

Neste cenário serão utilizadas as seguintes tecnologias: Xen, LVM, iSCSI, Multipath, Corosync e Pacemaker.

Tecnologias e procedimentos envolvidos

Todas as tecnologias envolvidas na implementação dos cenários propostos são livres e/ou de código aberto e estão disponíveis em sua maioria sob licença GPLv2. O Ubuntu 16.04 LTS será o sistema operacional utilizado nos servidores físicos. Não será descrito aqui o procedimento de instalação do sistema operacional, portanto, considerar que ele já está instalado e que o servidor possui conectividade com a internet para instalação dos pacotes.

Xen Hypervisor

Um *hypervisor* nada mais é do que um software capaz de criar e executar servidores virtuais em um computador ou, no nosso caso, em um servidor físico de grande porte. Neste documento será utilizado o Xen Hypervisor[1].

O Xen usa o conceito de dom0 e domU, onde o primeiro se refere ao sistema operacional do servidor físico que vai hospedar os servidores virtuais, enquanto o segundo se refere aos servidores virtuais hospedados no servidor físico. O dom0 pode hospedar várias domUs.

Por se tratar de um servidor físico de grande porte, o dom0 possui uma grande quantidade de memória RAM (128GB, por exemplo) e mais de um processador com diversos núcleos e com *Hyper-Threading* habilitado, vale ressaltar que uma *thread* de processamento equivale a uma CPU no servidor virtual. O servidor físico pode ter 24, 32, 64 *threads* ou até mais dependendo do tipo de processador existente nele. Em relação a disco nem sempre é necessário grande quantidade, uma vez que pode ser utilizado

[1] Disponível em: <http://www.xenproject.org/>.

um *storage* com vários TB de espaço disponível.

O comando abaixo instala o Xen Hypervisor:

```
# apt-get install xen-hypervisor-4.6-amd64 xen-tools
```

Concluída a instalação, abrir o arquivo /etc/default/grub.d/xen.cfg e alterar as diretivas:

```
XEN_OVERRIDE_GRUB_DEFAULT=1
GRUB_CMDLINE_XEN=”dom0_max_vcpus=2 dom0_vcpus_pin dom0_mem=6291456”
```

A primeira, XEN_OVERRIDE_GRUB_DEFAULT, define que o *kernel* xenificado será utilizado como padrão durante o boot do sistema. A segunda, GRUB_CMDLINE_XEN, define que a quantidade total de CPUs virtuais (VCPUs) e de memória RAM alocada para o servidor físico (dom0) será de 2 VCPUs e 6GB de RAM respectivamente, o restante estará disponível para alocação aos servidores virtuais.

Atualizar o *grub* para que as configurações passem a valer no próximo boot do servidor físico.

```
# update-grub2
```

O Xen pode utilizar diversas *toolstacks* que são as ferramentas utilizadas para gerenciar os servidores virtuais. Dentre elas existem: xm, xl, xe, virsh, etc. A partir da versão 4.1 do Xen, a *toolstack* xm, que era utilizada como padrão, foi descontinuada e substituída pela xl. Na versão 4.5 do Xen, a *toolstack* xm foi removida, não podendo mais ser utilizada, e, consequentemente, o uso do arquivo de configuração /etc/xen/xend-config.sxp, utilizado para definir algumas das configurações do Xen foi descontinuado.

As interfaces de rede dos servidores virtuais que serão criados farão parte de uma *bridge* que deverá ser criada no servidor físico, e posteriormente, no arquivo de configuração dos servidores virtuais, será necessário referenciá-la. Para criar a *bridge*, inserir os comandos abaixo no arquivo /etc/network/interfaces do servidor físico.

```
auto eth0
iface eth0 inet manual


auto xenbr0
iface xenbr0 inet dhcp
  bridge_ports eth0
  bridge_stp off # disable Spanning Tree Protocol
  bridge_waitport 0 # no delay before a port becomes available
  bridge_fd 0    # no forwarding delay

NOTA: eth0 é a interface que será utilizada para comunicação dos
servidores virtuais com os computadores da rede local. Seu nome deve
ser ajustado conforme nome da interface do servidor, uma vez que pode
variar conforme o modelo da placa de rede do servidor físico.
```

Como pode ser visto na configuração acima, a interface de rede do servidor físico pertencente a *bridge* é a eth0. É por esta interface que os servidores virtuais vão se comunicar com os demais equipamentos da rede local e com a internet. Da forma como está é possível criar servidores virtuais que pertençam apenas à mesma rede ou ao segmento de rede do servidor físico.

ATENÇÃO! A configuração a seguir é válida apenas para um ambiente que utiliza VLANs. Se este não for o caso, não efetuar estas modificações.

Numa infraestrutura com diversos segmentos de rede (VLANs), caso seja necessário executar servidores virtuais pertencentes a VLANs diferentes da do servidor físico, é preciso criar subinterfaces de rede e *bridge*s para todas as VLANs que eventualmente possam ter servidores virtuais em execução, inclusive para a VLAN do servidor físico. Neste caso, o arquivo /etc/network/interfaces deve ser alterado para que fique da seguinte forma:

```
auto eth0
iface eth0 inet manual

auto xenbr0
iface xenbr0 inet manual
  bridge_ports eth0
  bridge_stp off # disable Spanning Tree Protocol
  bridge_waitport 0 # no delay before a port becomes available
  bridge_fd 0    # no forwarding delay

# ID 2097 se refere a VLAN do servidor físico (ajustar se necessário)
auto eth0.2097
iface eth0.2097 inet manual
  vlan-raw-device eth0

auto xbr2097
iface xbr2097 inet dhcp
  bridge_ports eth0.2097
  bridge_stp off          # disable Spanning Tree Protocol
  bridge_waitport 0       # no delay before a port becomes available
  bridge_fd 0

# As demais VLANs podem ser adicionadas no diretório abaixo ao invés
de ficarem todas dentro do arquivo /etc/network/interfaces.
source-directory interfaces.d
```

Como mencionado acima, em vez de criar todas as subinterfaces e *bridge*s das demais VLANs no arquivo /etc/network/interfaces, podemos criar um arquivo para cada VLAN e colocá-lo dentro do diretório interfaces.d, desde que a diretiva source-directory esteja configurada. Por exemplo, criar o arquivo VLAN2032NTI em /etc/network/interfaces.d com o seguinte conteúdo:

```
auto eth0.2032
iface  eth0.2032 inet manual
       vlan-raw-device eth0
auto xbr2032
iface xbr2032 inet manual
bridge_ports eth0.2032
bridge_stp off
bridge_waitport 0
bridge_fd 0
```

Por último deve-se colocar a porta do *switch* no qual a interface eth0 do servidor físico está conectada em modo trunk, uma vez que ela deverá aceitar pacotes de diferentes segmentos de rede ao mesmo tempo.

Para finalizar a configuração do Xen, reiniciar o servidor físico para que ele passe a executar com o *kernel* xenificado e com as novas configurações definidas. Voltaremos ao Xen durante o procedimento de criação de um servidor virtual. O Xen Hypervisor será utilizado em todos os cenários propostos neste documento.

Comandos úteis do Xen

Alguns comandos úteis usados para gerenciamento dos servidores virtuais:

```
# xl list → lista os servidores virtuais em execução no servidor
físico
# xl create server.cfg → liga o servidor com base em seu arquivo de
configuração
# xl shutdown <server> → desliga o servidor
# xl destroy <server> → desliga o servidor abruptamente
# xl console <server> → conecta na console do servidor virtual
# xl info → mostra informações sobre os recursos do Xen
```

```
# xl migrate --live <server> <physicalserver> → migra o servidor
virtual para o servidor físico desejado


NOTA: A migração só funciona quando o servidor físico de origem (que
é o que está executando o servidor virtual) compartilha o disco
que armazena os dados do servidor virtual com o servidor físico de
destino. A opção --live efetua a migração sem desligar o servidor
virtual desde que as configurações de migração sejam efetuadas no Xen.
```

Logical Volume Manager (LVM)

O Logical Volume Manager (LVM) é um gerenciador de volumes lógicos e pode ser definido como uma área para armazenamento de dados. Ele funciona como uma partição em um disco, porém, oferece maior flexibilidade para gerenciamento dos volumes, sendo uma das principais a possibilidade de aumentar a área de armazenamento. O LVM é composto por 3 elementos básicos que são: 1) o volume físico; 2) o agrupamento de volumes; 3) o volume lógico.

O volume físico é um dispositivo de blocos (ou parte dele) identificado para ser utilizado pelo LVM (uma área de disco, espaço armazenável). Um grupo de volumes é um conjunto com um ou mais volumes físicos, ele permite que vários volumes físicos sejam agrupados formando uma área de armazenamento única para o sistema, e o volume lógico que é uma parte do grupo de volumes e será alocado para armazenamento dos dados do servidor virtual, ele vai funcionar com um disco rígido ou partição do servidor virtual. A Figura 3 abaixo exemplifica a estrutura do LVM.

Para instalar o LVM, o comando é:

```
# apt-get install lvm2
```

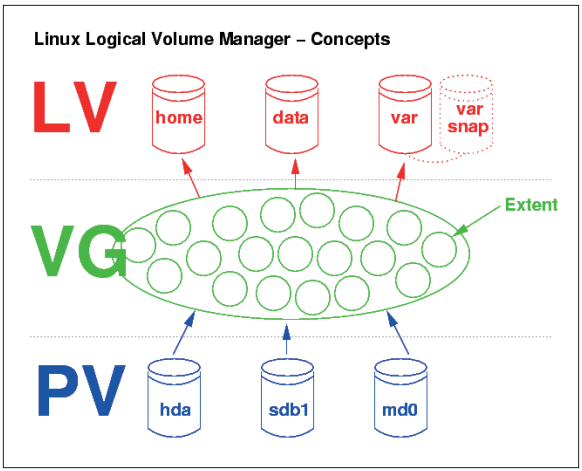

Figura 3. Exemplo da estrutura básica que compõem o LVM.
Fonte: [LVM Concepts] (2016).

O arquivo de configuração do LVM é o /etc/lvm/lvm.conf. Neste arquivo a diretiva *filter*, deve ser alterada conforme abaixo:

```
filter = [“r|block|”, “r|disk|”, “r|loop|”, “r|ram|”, “a|.*|” ]
global_filter = [“r|block|”, “r|disk|”, “r|loop|”, “r|ram|”, “a|.*|” ]
syslog = 0
file = “/var/log/lvm2.log”

Nota: no ambiente de cluster deverá ser acrescentado “r|sd.*|” na
diretiva filter, pois o disco local do servidor físico não deverá ser
utilizado para criação de volumes do LVM.
```

A diretiva filter permite (a) ou não permite (r) que um dispositivo de blocos possa ser utilizado como volume físico do LVM. Lembre-se que os dispositivos de blocos podem ser listados no /dev do servidor físico e possuem a letra b como primeiro caractere das permissões do arquivo. No exemplo, não será permitido usar os *device*s /dev/block, /dev/disk, /dev/loop e /dev/ram no LVM, os demais podem ser utilizados. Para negar o uso do dispositivo /dev/sda4, por exemplo, deve-se adicionar na diretiva “r|sda4|”.

A diretiva global_filter funciona da mesma forma que a anterior, mas, se for utilizado apenas a diretiva filter, alguns filtros podem ser sobrescritos utilizando a linha de comando. A diretiva *syslog* com o valor 0 (zero) indica que o log não deverá ser gerado no *syslog* e sim em um arquivo específico para o LVM que será o /var/log/lvm2.log.

Se ao instalar o SO em um servidor, a partição sda4 de 1TB foi criada como sendo do tipo LVM, para criar o volume físico, o agrupamento de volumes e um volume lógico, os comandos são, respectivamente:

```
# pvcreate /dev/sda4
# vgcreate nome_vg /dev/sda4
# lvcreate -L50G -n nome_lv nome_vg
```

Os comandos abaixo mostram as informações sobre os 3 elementos do LVM (a segunda forma entre parênteses exibe maiores detalhes):

```
# pvs (pvdisplay)
# vgs (vgdisplay)
# lvs (lvdisplay)
```

Quando um volume lógico é criado, o SO gera um novo *device* em /dev/nome_vg/nome_lv. Este *device* pode ser formatado com qualquer tipo de filesystem. Em seguida, pode-se montá-lo em alguma pasta do SO e armazenar dados nele. O comando abaixo formata o *device* criado com sendo do tipo ext4 e o monta em /mnt.

```
# mkfs.ext4 /dev/nome_vg/nome_lv
# mount /dev/nome_vg/nome_lv /mnt
```

Ao copiar dados para /mnt, na verdade eles estão sendo armazenados no dispositivo do LVM. Lembrar de desmontar o volume para que ele possa ser utilizado pelo servidor virtual.

O LVM será utilizado em todos os cenários do documento.

Comandos úteis do LVM

Os comandos a seguir também podem ser utilizados para gerenciar volumes lógicos do LVM:

```
# lvrename → renomeia um volume lógico
Ex.: lvrename nome_vg nome_lv novo_nome_lv
# lvremove → remove um volume lógico
Ex.: lvremove nome_vg/nome_lv
# lvextend → aumenta o tamanho de um volume lógico
Ex.: lvextend -L200G /dev/nome_vg/nome_lv → aumenta o volume lógico
para 200G
```

Criação do servidor virtual

Com o Xen instalado e o LVM configurado, o próximo passo é criar o primeiro servidor virtual. Vale ressaltar que não é necessário criar explicitamente um volume lógico do LVM para receber o servidor virtual, pois o comando de criação do servidor virtual vai se encarregar de fazer isso.

Antes de criar o primeiro servidor virtual, será necessário definir um servidor virtual modelo que será utilizado como base para criação dos novos servidores virtuais. Pode-se criar diversos servidores virtuais modelos, cada um com uma configuração específica que atenda a uma necessidade específica, facilitando e agilizando o processo de criação dos servidores virtuais.

Preparação do servidor virtual modelo

Criar um diretório qualquer onde ficarão armazenadas todos os servidores virtuais modelos e dentro dele criar um diretório que identifique o tipo de servidor virtual que será utilizado. Por exemplo:

```
# cd /home
# mkdir vms_modelo
# cd vms_modelo
```

```
# mkdir ubuntu_1604LTS_NTI
```

Instalar o pacote debootstrap, utilizado para baixar uma versão inicial do servidor virtual, como se fosse um CD de instalação.

```
# apt-get install debootstrap
```

Em seguida execute o comando para baixar a versão inicial do Ubuntu desejada.

```
# debootstrap --arch=amd64 xenial /home/vms_modelo/ubuntu_1604LTS_NTI
http://br.archive.ubuntu.com/ubuntu/
```

Assim que for concluído o download, entrar no diretório /home/vms_modelo/ubuntu_1604LTS_NTI e se enjaular dentro dele para que se possa customizar as configurações do servidor conforme necessidade específica. Quando é efetuado o enjaulamento, deve-se montar o /proc, /sys e /dev dentro da jaula e, assim, existirá a mesma estrutura de diretórios e arquivos de um servidor qualquer. Pode-se utilizar esta estrutura para trabalhar neste servidor como se ele fosse um servidor comum, como qualquer outro.

```
# chroot /home/vms_modelo/ubuntu_1604LTS_NTI
# mount /proc
# mount /sys
# mount /dev
```

Neste ponto devem ser configuradas as características específicas que este servidor modelo deverá possuir, como por exemplo, criar um usuário local padrão, definir variáveis de ambiente, definir um espelho interno para o *apt*, instalar e atualizar pacotes, etc.

Instalar um *kernel* para o servidor modelo e o grub e em seguida atualizar o grub. Vale ressaltar que o Ubuntu disponibiliza o *kernel* linux-image-virtual específico para servidores virtuais.

```
# apt-get install linux-image-virtual grub
# update-grub

NOTA: se o grub exibir uma tela perguntando em qual dispositivo ele
deve ser instalado, deixar todos desmarcados e prosseguir.
```

Todo servidor, seja físico ou virtual, precisa de uma área de *swap*. Será abordado o uso de *swap* em arquivo, apesar de existirem outras formas para criação de *swap* para servidores virtuais. Executar o conjunto de comandos abaixo para criar o arquivo que será utilizado como área de *swap* pelo servidor virtual.

```
# dd if=/dev/zero of=/swapfile bs=1024 count=2048k
# mkswap /swapfile
# echo vm.swappiness = 10 | sudo tee -a /etc/sysctl.conf
# chown root:root /swapfile
# chmod 0600 /swapfile
```

Este procedimento faz com que os servidores virtuais criados nesta infraestrutura utilizem como área de *swap* um arquivo dentro do próprio sistema de arquivos. Anteriormente, era bastante comum a utilização de partições como áreas de *swap*, no entanto, com a evolução dos sistemas de arquivos, o overhead de se utilizar uma área para *swap* dentro dele mesmo não representa mais uma perda significativa. Além disso, o gerenciamento dos volumes lógicos do LVM fica facilitado, visto que não é necessário criar um volume lógico para o disco e um volume lógico para o *swap* de cada servidor virtual, além de ser possível alterar o tamanho do *swap* de forma bem simples, apenas alterando o tamanho do arquivo.

Criar o arquivo /etc/fstab com o conteúdo abaixo.

```
# /etc/fstab: static file system information.
#
# <file system><mount point> <type> <options>  <dump> <pass>
proc           /proc         proc   defaults     0      0
```

```
devpts          /dev/pts      devpts  rw,noexec,nosuid,gid=5,mode=620  0
0
/dev/xvda1 /    ext4          errors=remount-ro   0       1
/swapfile       none          swap      sw        0       0
```

Desmontar o /proc, /sys e /dev e desenjaular.

```
# umount /proc
# umount /sys
# umount /dev
# exit
```

Sempre que for necessário efetuar ajustes no servidor modelo, repetir o procedimento efetuado acima, ou seja, se enjaular dentro da pasta do servidor modelo, montar o /proc, /sys e /dev, efetuar as alterações necessárias, desmontar /proc, /sys e /dev e sair da jaula.

ATENCÃO! Tomar muito cuidado com o procedimento de enjaulamento, pois um comando executado no arquivo /etc/fstab, por exemplo, sem estar enjaulado, refletirá no arquivo do servidor físico e não no arquivo do servidor virtual e sempre sair da jaula quando concluir as customizações e ajustes do servidor virtual modelo.

Com o servidor modelo pronto, pode-se iniciar o procedimento de criação de servidor virtual.

O primeiro servidor virtual

Antes de executar o comando de criação do primeiro servidor virtual, deve-se definir: um endereço MAC para ele (--mac), um nome (--hostname), o tamanho do disco (--size), a quantidade de memória (--memory), a quantidade de cpus virtuais (--vcpus), a arquitetura (--arch), a versão da distribuição (--dist), o método utilizado para sua criação (--install-method), o caminho onde se encontra o servidor modelo (--install-source), o tipo

do filesystem (--fs), se deve ser gerada uma senha para o usuário root (--genpass), o nome do agrupamento de volumes do LVM onde deverá ser criado o volume lógico referente ao servidor virtual (--lvm) e que o servidor virtual usará o seu próprio *kernel* ao invés de usar o *kernel* do servidor físico (--pygrub).

Se o novo servidor virtual possuir IP fixo, ainda pode-se usar os parâmetros que identificam seu IP (--ip), máscara (--netmask) e gateway (--gateway). Se o IP for obtido através de servidor DHCP, usar o parâmetro --dhcp.

Outros comandos úteis são: --force, que força a criação do volume lógico do LVM mesmo que ele já exista, neste caso, ele será removido e criado novamente e o --noswap que não vai gerar um volume lógico do LVM para o swap, afinal está sendo usado swap em arquivo e não é preciso se preocupar com ele, uma vez que o servidor modelo já está preparado para isso.

Seguem 2 exemplos de comandos para criação de um servidor virtual:

```
# xen-create-image --force --lvm=nome_vg --hostname=myserver
--size=50Gb --noswap --memory=2Gb --ip=10.129.1.11
--netmask=255.255.255.0 --gateway=10.129.1.100 --arch=amd64
--dist=xenial --pygrub --genpass=0 --fs=ext4 --vcpus=2 --install-
method=copy –install-source=/home/vms_modelo/ubuntu_1604LTS_NTI


# xen-create-image --lvm=nome_vg --hostname=myserver --size=50Gb
--noswap --memory=2Gb --dhcp --arch=amd64 --dist=xenial --pygrub
--genpass=0 --fs=ext4 --vcpus=2 --install-method=copy –install-
source=/home/vms_modelo/ubuntu_1604LTS_NTI
```

No diretório /var/log/xen-tools é possível visualizar os logs, caso ocorra algum problema durante a criação do servidor virtual.

Gerenciador do *Storage*

Os *storage*s são equipamentos capazes de oferecer áreas de armazenamento a servidores que precisam ter sua capacidade de armazenamento de dados expandida. Eles vêm acompanhados de um software de gerenciamento fornecido pelo seu fabricante que possuem características peculiares. O acesso à interface de gerenciamento e aos procedimentos de configuração são exclusivos de cada equipamento, portanto, não é possível descrever aqui um procedimento que valha para todos, porém, a lógica de funcionamento de um *storage* é semelhante e é esta lógica que será descrita aqui. Lembre-se de ter em mãos o manual do equipamento que contém as informações específicas de cada um.

O software de gerenciamento pode ser fornecido em CD-ROM, devendo ser instalado em um notebook ou PC, ou pode ser uma interface web. Os *storage*s possuem diversos tipos de interfaces diferentes (iSCSI, SAS, fibre channel), e, além dessas, existem também as interfaces de gerenciamento que possibilitam o acesso aos recursos oferecidos por eles. Estas interfaces devem ser configuradas com um IP para que possam ser acessadas na rede interna.

Nesta apostila serão utilizadas as interfaces iSCSI para comunicação entre os servidores físicos e o *storage*, assim, se faz necessário configurar estas interfaces nos equipamentos, tanto do lado do servidor quanto do *storage*. A configuração das interfaces do lado do servidor será descrita na próxima seção, para configurar as interfaces iSCSI do *storage* e consultar o manual do equipamento que contém as instruções. Lembre-se de usar o endereçamento de rede definido nas Figuras 1 e 2.

Com acesso à interface de gerenciamento do *storage*, deve-se localizar os discos pertencentes a ele e, em seguida, criar um grupo de discos. Este grupo define um tipo de RAID para um conjunto de discos com características iguais, pode-se criar um único grupo com todos os discos ou dividir em quantos grupos forem necessários. Em um *storage* com 12 discos, por exemplo, pode-se criar 2 grupos sendo um com 8 discos em RAID10 e outro com 4 em RAID 5. Os *storage*s também oferecem a possibilidade de utilização de disco sobressalente (hotspare) com substituição automática (hotswap). Cada grupo de discos deve possuir um disco sobressalente de forma que em uma eventual falha em um dos discos do grupo, o disco

sobressalente assume seu lugar.

Em seguida devem ser criadas as LUNs ou volumes que representam discos virtuais. O disco virtual é o HD que o servidor físico vai usar como se fosse um HD local. Os discos virtuais são criados dentro de um grupo de discos, podendo ser do tamanho total do grupo ou apenas parte dele. Definidas as LUNs, devem ser cadastrados os hosts (servidores físicos) que poderão ter acesso ao *storage* e as LUNs criadas.

Por último, os hosts devem ser mapeados a suas respectivas LUNs, somente após este procedimento é que o servidor físico passará a enxergar a LUN criada no *storage* como um disco local. Em alguns modelos de *storage*, o mapeamento é possível apenas após ter sido executado o procedimento de discovery do *storage* a partir do servidor, que estabelecerá a primeira comunicação entre eles. Os detalhes de estabelecimento da comunicação entre os equipamentos estão descritos na próxima seção.

iSCSI

iSCSI é um protocolo que possibilita o acesso a uma área de armazenamento de dados através de uma rede TCP/IP. O que ele faz basicamente é transportar comando SCSI de leitura e escrita em disco via rede TCP/IP.

Sua lógica de funcionamento é bastante simples e se parece com uma arquitetura do tipo cliente/servidor, onde o cliente é quem necessita de uma área de armazenamento e o servidor é quem fornece esta área – o *storage*. Na terminologia do iSCSI, o cliente é o initiator e o servidor é o target.

O protocolo iSCSI será utilizado nos cenários 2 e 3 deste material, onde existe a figura de um *storage* para armazenamento dos dados. Como o servidor físico é quem vai precisar visualizar uma área para armazenar os dados (ele enxergará esta área como um disco local), é ele que fará o papel do cliente, portanto, o cliente iSCSI deve ser instalado no servidor físico.

```
# apt-get install open-iscsi
```

O arquivo de configuração do open-iscsi é o /etc/iscsi/iscsid.conf. Neste arquivo deve ser alterada a seguinte diretiva:

```
node.startup = automatic
```

Esta diretiva indica que não será necessário efetuar login no *storage* para que o servidor físico enxergue o disco iSCSI como um disco local. Ao reiniciar o daemon iSCSI ou durante o boot do servidor físico, esta tarefa será executada automaticamente.

Outro arquivo importante é o /etc/iscsi/initiatorname.iscsi. Cada cliente iSCSI possui um identificador único, chamado initiator e este arquivo contém justamente o valor deste identificador e não deve ser alterado em hipótese alguma, pois este valor é utilizado pelo *storage* para autorizar um cliente (host). A mudança deste valor acarretará a perda do acesso do servidor ao *storage*. Vale ressaltar que se o pacote do open-iscsi for removido e instalado novamente, será gerado um novo identificador diferente do anterior.

Conforme mencionado, o iSCSI funciona em conjunto com o protocolo TCP/IP e a próxima etapa da configuração se refere ao endereçamento de rede entre o servidor físico e o *storage*. Todo o tráfego iSCSI entre o servidor e o *storage* deve estar separado do tráfego da rede local, além disso devemos ter caminhos redundantes entre estes equipamentos para aumentar a disponibilidade e evitar que falhas de hardware possam colapsar a infraestrutura.

A grande maioria dos *storage*s possui 2 controladoras, justamente para garantir que na ocorrência de falha em uma delas a outra possa ser utilizada sem prejuízo para a infraestrutura. Cada controladora possui um conjunto de portas iSCSI e será necessário configurar no mínimo uma porta iSCSI em cada controladora conforme endereçamento de rede definido na Figura 1. Na necessidade de conexão de diversos servidores com o *storage*, pode-se utilizar um *switch* entre os equipamentos. Esta configuração é a utilizada no cenário 3.

Considerando que as interfaces iSCSI do *storage* estão configuradas e as conexões físicas entre os equipamentos efetuadas, deve-se configurar as interfaces de rede do servidor físico. No arquivo /etc/network/interfaces, adicionar o conteúdo:

```
# Trafego iSCSI principal
auto eth1
iface eth1 inet static
address 192.168.1.50
netmask 255.255.255.0

# Trafego iSCSI redundante
auto eth2
iface eth2 inet static
address 192.168.2.50
netmask 255.255.255.0

Nota: no cenário de cluster efetuar esta configuração para os demais
servidores físicos lembrando de alterar o IP
```

Efetuar um ping a partir do servidor para os IPs do *storage*, a fim de garantir que a comunicação entre eles está ocorrendo normalmente. Se não estiver, revisar os procedimentos de configuração dos IPs e as conexões físicas.

Considerando que a LUN (disco virtual) foi criada no *storage*, efetuar o discovery no *storage* a partir do servidor físico para que ele conheça todos os *targets* existentes no *storage*. Como existem 2 caminhos (um para cada controladora) serão executados 2 comandos de *discovery*. Em seguida reiniciar o daemon do open-iscsi.

```
# iscsiadm -m discovery -t st -p 192.168.1.1
192.168.1.1:3260,11 iqn.1986-03.com.ibm:2145.storage.node1
# iscsiadm -m discovery -t st -p 192.168.2.1
192.168.2.1:3260,11 iqn.1986-03.com.ibm:2145.storage.node2
# service open-iscsi restart
```

No software de gerenciamento do *storage*, deve-se criar um host, caso ainda não tenha sido criado, e associá-lo ao seu IQN (valor da diretiva

InitiatorName do arquivo /etc/iscsi/initiatorname.iscsi do servidor físico). Em seguida deve-se criar uma LUN (disco virtual) no *storage*, caso ainda não tenha sido criada, e mapeá-la ao host.

Considerando que o mapeamento foi efetuado, deve-se executar o comando de login no servidor físico para que ele possa enxergar os discos virtuais criados no *storage* como discos locais, somente assim o servidor físico poderá armazenar dados no disco virtual criado no *storage*.

```
# iscsiadm -m node --targetname "iqn.1986-03.com.ibm:2145.storage.
node1" --portal "192.168.1.1:3260" --login
Logging in to [iface: default, target: iqn.1986-03.com.ibm:2145.
storage.node1, portal: 192.168.1.1:3260]
Login to [iface: default, target: iqn.1986-03.com.ibm:2145.storage.
node1, portal: 192.168.1.1,3260]: successful

# iscsiadm -m node --targetname "iqn.1986-03.com.ibm:2145.storage.
node2" --portal "192.168.2.1:3260" --login
Logging in to [iface: default, target: iqn.1986-03.com.ibm:2145.
storage.node2, portal: 192.168.2.1:3260]
Login to [iface: default, target: iqn.1986-03.com.ibm:2145.storage.
node2, portal: 192.168.2.1,3260]: successful
```

Uma forma mais simples é reiniciar o *daemon* do iSCSI, pois como a diretiva *node.startup f*oi alterada para automatic, a reinicializacão do *daemon* executa o procedimento de login e logout de forma automática.

```
# service open-iscsi restart
```

Ao efetuar o procedimento acima, pode-se verificar que os discos virtuais foram criados no servidor físico como discos locais executando o comando:

```
# ls -l /dev/sd*
brw-rw---- 1 root disk 8,  0 Abr  7 11:14 /dev/sda
brw-rw---- 1 root disk 8,  1 Abr  7 11:14 /dev/sda1
```

```
brw-rw---- 1 root disk 8,  2 Abr  7 11:14 /dev/sda2
brw-rw---- 1 root disk 8,  3 Abr  7 11:14 /dev/sda3
brw-rw---- 1 root disk 8,  5 Abr  7 11:14 /dev/sda4
brw-rw---- 1 root disk 8, 16 Abr  7 11:15 /dev/sdb
brw-rw---- 1 root disk 8, 32 Abr  7 11:15 /dev/sdc
```

No exemplo, os dispositivos sdb e sdc nada mais são do que o disco virtual criado no *storage* visto como discos locais no servidor físico. Mas neste momento pode-se perguntar: Por que existem 2 discos se foi criada e mapeada apenas uma LUN ao servidor físico? Isto ocorre pois temos 2 caminhos entre o servidor e o *storage*. O resultado do comando fdisk -l pode confirmar isso:

```
# fdisk -l /dev/sdb
Disk /dev/sdb: 5 TiB, 5497558138880 bytes, 10737418240 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
# fdisk -l /dev/sdc
Disk /dev/sdc: 5 TiB, 5497558138880 bytes, 10737418240 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
```

Note que o tamanho dos dispositivos é idêntico. Outro comando que pode ser utilizado para confirmar que ambos são o mesmo dispositivo é o scsi_ id, que mostra um identificador único do dispositivo desejado.

```
# /lib/udev/scsi_id --whitelisted --device=/dev/sdb
360022a11000aeda0001cf7bf00000000
# /lib/udev/scsi_id --whitelisted --device=/dev/sdc
360022a11000aeda0001cf7bf00000000
```

Note que o valor retornado é o mesmo, indicando que eles são o mesmo disco.

Multipath

O Multipath é a ferramenta utilizada para efetivamente tirar proveito dos caminhos redundantes criados entre o servidor físico e o *storage*. É ele que gerencia qual caminho deve ser utilizado e efetua o chaveamento automático para o caminho redundante em caso de falha no principal.

Ele utiliza os UUIDs dos dispositivos para identificar aqueles que são idênticos e desta forma usar o que for mais conveniente. O outro apenas será utilizado se houver algum tipo de falha de comunicação com o primeiro, seja física ou lógica. Para instalar o *Multipath*, o comando é:

```
# apt-get install multipath-tools
```

Se o arquivo /etc/multipath.conf não existir, o próprio Multipath vai tentar identificar o modelo do *storage* que está sendo usado e efetuar a configuração. Ele usará como referência os dados do arquivo /usr/share/doc/multipath-tools/examples/multipath.conf.annotated. O arquivo multipath.conf pode ser criado manualmente com as configurações conforme o ambiente. Segue um exemplo de configuração:

```
defaults {
  user_friendly_names yes # exibe um nome amigável para o device
criado
}

# lista os devices que não podem ser utilizados no multipath
blacklist {
  devnode "^(ram|raw|loop|fd|md|dm-|sr|scd|st)[0-9]*"
  devnode "^sda"
}

# Define as informações sobre como se comunicar com o storage. Este
exemplo de configuração é para o IBM V3700, para outros modelos a
configuração pode ser outra
devices {
  device {
```

```
    vendor "IBM"
    product "2145"
    path_grouping_policy failover
    getuid_callout "/lib/udev/scsi_id --whitelisted --device=/dev/%n"
    features "1 queue_if_no_path"
    prio alua
    path_checker tur
    failback immediate
  }
}

# Especifica o WWID do dispositivo e define o nome que será associado a
ele
multipaths {
  multipath {
    wwid "360022a11000aed9e1651afd600000001"
    alias mpath0
  }
}
```

Quando uma LUN é criada no *storage*, ela recebe um identificador gerado pelo próprio *storage* que é diferente do UUID exibido pelo comando blkid, este identificador é chamado de World Wide Name (WWN) e wwid para o Multipath . Para verificar se o Multipath está funcionando corretamente, deve-se executar o comando:

```
# multipath -ll
mpath0 (360022a11000aed9e1651afd600000001) dm-1 IBM,2145
size=5T features='1 queue_if_no_path' hwhandler='0' wp=rw
|-+- policy='round-robin 0' prio=50 status=active
| `- 7:0:0:0  sdb 8:16 active ready running
```

```
`-+- policy='round-robin 0' prio=10 status=enabled
  `- 8:0:0:0 sde 8:32 active ready running
```

Pode-se notar que o device mpath0 equivale a um disco de 5TB, que na verdade se refere aos discos sdb e sdc, que representam a LUN (disco virtual) que foi criado no *storage* e é acessada por 2 caminhos diferentes.

Sempre que precisar fazer referência ao disco de 5TB, deve-se utilizar o *device* /dev/mapper/mpath0, pois em caso de falha em um dos caminhos este *device* se encarrega de utilizar o outro. Se for utilizado o *device* / dev/sdb, por exemplo, e ele fizer referência ao caminho da controladora 1, caso haja falha em algum componente deste caminho, o acesso ficará indisponível, o que não ocorre se for utilizado o *device* do Multipath.

Criação do servidor virtual no *Storage*

Da mesma forma que ocorre quando se cria um volume no LVM, quando o *Multipath* está ativo um *device* é criado em /dev/mapper, no exemplo o *device* é /dev/mapper/mpath0. Ele pode ser utilizado como um volume físico do LVM onde poderão ser criados os volumes lógicos que armazenarão efetivamente os dados dos servidores virtuais. Os comandos abaixo adicionam o *device* do Multipath num volume físico do LVM e também cria um grupo de volumes onde será possível criar os volumes lógicos dos servidores virtuais.

```
# pvcreate /dev/mapper/mpath0
# vgcreate storage_vg /dev/mapper/mpath0
```

Com as configurações efetuadas até este ponto, já se pode criar um servidor virtual, que terá todos os seus dados armazenados no volume que foi criado no *storage*. Basta criar o servidor virtual passando no parâmetro --lvm do comando xen-create-image o nome do grupo do LVM criado anteriormente que representa a área de armazenamento do *storage*.

```
# xen-create-image --lvm=storage_vg --hostname=myserver2
--size=50Gb --noswap --memory=2Gb --dhcp --arch=amd64
--dist=xenial --pygrub --genpass=0 --fs=ext4 --vcpus=2
```

```
--install-method=copy --install-source=/home/vms_modelo/
ubuntu_1604LTS_NTI
```

Corosync e Pacemaker

O Corosync e o Pacemaker são duas ferramentas que trabalham em conjunto para formar uma infraestrutura de *cluster*. O primeiro é responsável pela comunicação entre os servidores físicos (nós) para que o *cluster* possa ser estabelecido e funcionar, enquanto o segundo gerencia os recursos que serão utilizados nele. Entenda-se como recursos os diversos tipos de serviços que podem ser utilizados, dentre eles, servidores virtuais executando sobre o Xen.

Assim como acontece na configuração do iSCSI, também será necessário utilizar 2 interfaces de rede distintas nos servidores físicos. Desta forma, o tráfego de rede referente ao *cluster* que for gerado entre os nós não se mistura com o tráfego da rede local. Além disso, também haverá redundância entre os anéis (infraestrutura de rede que conecta os nós do *cluster*) de forma a aumentar a disponibilidade do *cluster*. Caso o servidor físico não possua mais interfaces de rede disponíveis, pode-se utilizar as mesmas interfaces já utilizadas no iSCSI. Esta será a abordagem apresentada neste material.

Configurar as interfaces de rede dos demais nós que farão parte do *cluster* conforme já explicado na configuração do iSCSI. Usar a tabela abaixo:

Tabela 1. Endereços de rede das interfaces principal e redundante

Nó	Iface principal	Iface redundante
0	192.168.1.50/24	192.168.2.50/24
1	192.168.1.51/24	192.168.2.51/24
2	192.168.1.52/24	192.168.2.52/24
3	192.168.1.53/24	192.168.2.53/24

Para instalar o Corosync e o Pacemaker, o comando é:

```
# apt-get install corosync pacemaker
```

Lembre-se que eles devem ser instalados em todos os nós que farão parte do *cluster*. No arquivo /etc/corosync/corosync.conf, deverão ser efetuadas as configurações necessárias e todos os nós deverão possuir a mesma configuração. As seguintes modificações deverão ser efetuadas no corosync.conf.

```
totem {
...
        rrp_mode: passive
        interface {
                ringnumber: 0
                bindnetaddr: 192.168.1.0
                mcastaddr: 239.255.1.1
                mcastport: 5415
        }
        interface {
                ringnumber: 1
                bindnetaddr: 192.168.2.0
                mcastaddr: 239.255.2.1
                mcastport: 5416
        }
}
logging {
        to_logfile: yes
        logfile: /var/log/corosync/corosync.log
        to_syslog: no
}
quorum {
        provider: corosync_votequorum
        expected_votes: 4
}
nodelist {
```

```
        node {
                nodeid: 50
                name: server50
                ring0_addr: 192.168.1.50
                ring1_addr: 192.168.2.50
        }
        node {
                nodeid: 51
                name: server51
                ring0_addr: 192.168.1.51
                ring1_addr: 192.168.2.51
        }
        node {
                nodeid: 52
                name: server52
                ring0_addr: 192.168.1.52
                ring1_addr: 192.168.2.52
        }
        node {
                nodeid: 53
                name: server53
                ring0_addr: 192.168.1.53
                ring1_addr: 192.168.2.53
        }
}
```

A diretiva rrp_mode determina o protocolo de anel redundante (Redundant Ring Protocol – RRP) que deverá ser utilizado. Como foram definidos 2 anéis de forma a prover redundância, esta diretiva precisa conter o valor active ou passive. A primeira utiliza as duas interfaces ao mesmo tempo, enquanto a segunda utiliza ao anéis de forma alternada. Existe uma terceira opção, none, que é utilizada quando há apenas um anel.

Na seção interface, são definidas os anéis do *cluster* (podem ser configurados no máximo 2 anéis), o endereço de rede de cada um deles e como será a descoberta dos nós. Caso exista outro *cluster* na infraestrutura que esteja utilizando a mesma rede definida em bindnetaddr, utilizar portas diferentes em mcastport para a configuração de cada *cluster*, caso contrário, os nós do novo *cluster* entrarão como novos nós do já existente.

A seção logging, determina as configurações de log da ferramenta.

Na seção quorum, define-se quantos nós farão parte do *cluster* e quantos votos serão necessários para definir o quorum. Quorum é uma propriedade que garante que o *cluster* seja considerado consistente somente se tiver mais da metade dos seus nós ativos. No exemplo, são esperados 4 votos, caso ocorra apenas metade dos votos o *cluster* é considerado inconsistente.

A seção nodelist define informações sobre os nós que farão parte do *cluster*, dentre elas, o endereço IP específico de cada nó no anel em questão.

Editar o arquivo /etc/*host*s de todos os servidores físicos para que fiquem conforme abaixo. Lembre-se de ajustar o nome e IP de acordo com os nomes e IPs de cada um dos servidores.

```
127.0.0.1           localhost.cnptia.embrapa.br     localhost
ip.da.rede.local    server50.cnptia.embrapa.br      server50
```

Iniciar o Corosync e o Pacemaker em todos os nós.

```
# service corosync start
# service pacemaker start
```

Vale ressaltar que quando o *daemon* do *Corosync* é parado, o *daemon* do *Pacemaker* também para automaticamente, porém, quando se inicia o *daemon* do Corosync, é necessário iniciar em seguida o *daemon* do *Pacemaker* pois ele não é iniciado automaticamente.

Os comandos a seguir são específicos para configurar determinadas situações de comportamento do *cluster* e dependem da quantidade de nós, e

características de funcionamento desejadas. Estes comandos devem ser executados uma vez em apenas um dos nós.

Desativar o STONITH (acrônimo para Shoot The Other Node In The Head). Esta propriedade é utilizada quando se deve garantir que uma segunda cópia de um recurso não rode ao mesmo tempo que a primeira. Se isso ocorrer, ele desliga um dos nós para garantir que o recurso seja executado em apenas um deles. O comando pode ser executado em apenas um nó do *cluster*.

```
# crm configure property stonith-enabled=false
```

Ignorar o QUORUM desta forma, ainda que menos da metade dos nós do *cluster* estejam ativos, o *cluster* será mantido em funcionamento.

```
# crm configure property no-quorum-policy=ignore
```

Existem outras opções que podem ser configuradas no *cluster*, que vão determinar seu modo de funcionamento:

- migration-limit: define o número máximo de recursos que poderão ser migrados entre os nós de uma única vez. É utilizado para limitar o número de servidores virtuais que poderão ser migrados de uma só vez entre os nós do *cluster*.

```
# crm configure property migration-limit=X
```

- maintenance-mode: coloca o *cluster* em modo de manutenção. Neste modo, os recursos deixam de ser monitorados e é possível alterar os arquivos de configuração do corosync e do pacemaker para que as novas configurações entrem em vigor.

```
# crm configure property maintenance-mode=true
```

- resource-stickiness: se seu valor for 0 (zero), o próprio *cluster* vai se encarregar de migrar automaticamente os recursos entre os nós de forma a não sobrecarregar nenhum nó. Se o seu valor for 100 (cem), o gerenciador do *cluster* não migra os recursos de forma automática, ficando sob responsabilidade do operador do *cluster* efetuar manualmente esta tarefa.

```
# crm configure rsc_defaults resource-stickiness=100
```

Para verificar se o *cluster* foi estabelecido, executar o comando abaixo:

```
# crm status
Last updated: Tue Apr 14 20:01:31 2015
Last change: Mon Apr 13 14:46:43 2015 via cibadmin on server53
Stack: corosync
Current DC: server53 (version 1.1.14-70404b0) - partition with quorum
4 Nodes configured
Online: [ server50 server51 server52 server53 ]
```

Note que o *cluster* está com o status Online e são exibidos os nomes dos servidores físicos que fazem parte dele. Com o *cluster* estabelecido, o próximo passo é começar a criar os recursos de servidores virtuais do Xen.

Estas duas ferramentas são utilizadas apenas no cenário 3 deste material.

Recurso do Xen no *cluster*

Como já mencionado, um *cluster* pode gerenciar os mais variados tipos de recursos, como por exemplo, Apache, bancos de dados (MySQL e PostgreSQL), LDAP, Squid, etc. Neste documento usaremos apenas o recurso do Xen para gerenciar o *cluster* de servidores virtuais.

Depois de instalado o Corosync e o Pacemaker, o gerenciador do *cluster* utiliza scripts em shell para efetuar o gerenciamento dos recursos. O recurso do Xen fica localizado em /usr/lib/ocf/resource.d/heartbeat/Xen. Este script possui diversas funções, cada uma responsável por uma tarefa efetuada pelo Xen, como por exemplo, monitorar um servidor virtual, iniciá-lo, finalizá-lo, migrá-lo entre os nós, etc.

Para que o recurso do gerenciamento do Xen possa funcionar corretamente num ambiente em *cluster*, deve-se efetuar inicialmente algumas modificações no arquivo /etc/default/xendomains, para que fiquem com os seguintes valores:

```
XENDOMAINS_SAVE=

XENDOMAINS_RESTORE=false
```

XENDOMAINS_SAVE tem que ficar com o valor vazio, pois caso o servidor físico seja desligado com algum servidor virtual em execução, o Xen vai tentar salvar o seu estado no caminho indicado antes de desligá-lo.

XENDOMAINS_RESTORE deve ser desativado, caso contrário, os servidores virtuais que tiverem seu estado salvo durante o desligamento do servidor físico serão restaurados neste mesmo servidor. Não se pode correr este risco, pois o *cluster* pode iniciar o servidor virtual em outro servidor físico e desta forma corre-se o risco de ter o mesmo servidor virtual sendo executado em servidores físicos diferentes. Como o *cluster* migra os servidores virtuais durante o desligamento, quando o servidor físico voltar a ficar online, ele não pode iniciar o servidor virtual nele mesmo, uma vez que o servidor virtual estará em execução em outro servidor físico. Se o mesmo servidor virtual rodar em 2 servidores físicos ao mesmo tempo, seus dados serão corrompidos.

O disco compartilhado entre os servidores do *cluster*

Num ambiente em *cluster*, deve-se criar um volume no *storage* que será compartilhado entre todos os servidores físicos pertencentes ao *cluster*. O procedimento de criação segue a mesma lógica definida no item 3.4, mas com uma característica peculiar. Ao criar o volume, ele deverá ser mapeado para todos os hosts que fazem parte do *cluster*. Neste cenário o próprio *storage* deverá emitir um alerta informando que ao mapear o mesmo volume para mais de um host, o usuário deverá estar ciente de que será sua responsabilidade garantir que os hosts não utilizem uma mesma área do disco ao mesmo tempo, pois, se isto ocorrer, dados poderão ser corrompidos.

Uma vez que o volume foi criado e mapeado para todos os *hosts*, deve-se reiniciar o *daemon* iSCSI em todos os servidores físicos para que eles passem a enxergar o novo disco. Lembre-se de certificar que os discos não estejam em uso para que não ocorra travamento do servidor físico.

```
# service open-iscsi restart

NOTA: Se o comando acima não funcionar, efetuar o logout e em seguida
o login da sessão iSCSI para que o novo disco seja reconhecido pelo
servidor físico.
```

Reiniciar também o *daemon* do Multipath para identificar o wwid do novo disco e adicioná-lo dentro da seção multipaths do arquivo /etc/multipath.conf dos servidores físicos, conforme exemplo exibido no item 3.6.

```
No arquivo multipath.conf, dentro da seção multipaths adicionar o
trecho:
multipath {
    wwid "360022a11000aeda0001cf7bf00000000"
    alias mpath1
  }

# service multipath-tools restart
```

Por fim, adicionar o device do Multipath como um volume físico do LVM e em seguida criar um grupo do LVM com o mesmo device. Este grupo deverá ser utilizado na criação dos servidores virtuais no *cluster*.

```
# pvcreate /dev/mapper/mpath
# vgcreate cluster_vg /dev/mapper/mpath1

NOTA: estes comandos deverão ser executados em apenas um dos
servidores físicos
```

Habilitando o Live Migration

O live migration é a uma opção que permite que os servidores virtuais possam ser migrados de um servidor físico para o outro. Esta opção é extremamente útil nas situações onde é necessário efetuar alguma manutenção em um servidor físico, por exemplo. Numa situação deste tipo, os servido-

res virtuais são migrados do servidor físico que sofrerá a manutenção para outro servidor físico que esteja íntegro no *cluster*. Esta migração é feita a quente, o que indica que o servidor virtual não ficará indisponível durante o processo de migração. Esta opção fornece grande flexibilidade para o administrador de redes que pode efetuar manutenções corretivas e preventivas nos servidores físicos do *cluster* sem que seja necessário parar os serviços em execução nos diversos servidores virtuais dos usuários que estejam em execução nos servidores do *cluster*.

Para que o live migration funcione, é necessário indicar para o *cluster* qual interface de rede deverá ser utilizada para migração. Isto é feito definindo atributos para os nós, para criar um atributo em um nó do *cluster*, o comando é:

```
crm_attribute --type nodes --node-uname <nomde_do_no> --attr-name
<nome_do_atributo> --attr-value <valor_desejado>
Ex.: crm_attribute --type nodes --node-uname server50 --attr-name
ip_livemigration --attr-value 192.168.1.50
NOTA: criar o atributo para todos os nós do cluster alterando os
valores conforme configuração do IP de cada nó
```

Quando o recurso do Xen é demandado para executar uma operação de migração, ele executa o comando abaixo para obter o valor do atributo que indica a interface.

```
# crm_attribute --type nodes --node-uname server50 --attr-name ip_
livemigration --get-value -q
```

Uma vez que o atributo foi criado, devemos passar no comando de geração do recurso do Xen, o nome do atributo que será utilizado para esta finalidade. Quando ocorrer um live migration, será checado o nome do atributo definido em node_ip_attribute. A partir do nome, será obtido o valor do atributo (que é o IP da interface), e o live migration utilizará esta interface. O comando do live migration executado pelo *cluster* ficaria da seguinte forma:

```
# xm migrate --live <nomde_da_vm_a_ser_migrada> 192.168.200.65
```

Se o atributo não tivesse sido definido, o *cluster* faria o live migration utilizando o nome do host como parâmetro e a transferência do servidor virtual entre os nós seria efetuado pela interface de acesso. O comando acima ficaria da seguinte forma:

```
# xm migrate --live <nomde_da_vm_a_ser_migrada>  server50
```

Para apagar um atributo o comando é:

```
# crm_attribute --type nodes --node-uname server50 --attr-name
<nome_do_atributo> --delete
```

Para atualizar o valor de um atributo:

```
crm_attribute --type nodes --node-uname server50 --attr-name <nome_
do_atributo> --update <novo_valor>
```

Para finalizar a configuração do live migration, deve-se configurar o acesso via SSH com chave e sem senha entre todos os servidores físicos do *cluster*. A partir do Xen 4.5 o live migration funciona por SSH e não é mais efetuado por meio da porta 8002. Para efetuar a configuração do SSH nos servidores, executar o procedimento a seguir:

Entrar como root em cada um dos servidores físicos e no diretório /root, executar o comando:

```
# ssh-keygen -b 2048 -t rsa
```

Será exibida a mensagem "*Enter file in which to save the key* (/root/.ssh/id_rsa):", pressione *Enter*. Este é o local onde serão geradas as chaves, sendo que, id_rsa se refere à chave privada e a id_ras.pub se refere à chave pública.

```
NOTA!: não trocar o nome do arquivo id_rsa para outro qualquer. Para
trocar, é preciso executar o ssh, passando, com o parâmetro da chave
privada, o nome escolhido, e neste caso, não é usado o nome padrão
(id_rsa).
```

Em seguida será exibida a mensagem: "*Enter passphrase (empty for no passphrase)*:", não informe senha nenhuma, pois queremos fazer o login sem senha. Pressione Enter para concluir.

Copiar a chave pública (id_rsa.pub) de um servidor para a pasta /root/.

ssh de todos os outros. Lembrar de alterar o nome do arquivo no destino para não sobrescrever o arquivo id_rsa.pub original do servidor. Entrar no servidor de destino e criar o arquivo authorized_keys e dentro dele colocar o conteúdo do arquivo id_rsa.pub, a permissão deste arquivo deve ser 600.

Por último efetuar um teste de conexão via SSH entre os servidores físicos, mas lembrar que a conexão deve ser efetuada no IP da interface que será utilizada no *live migration*. Executar em todos os servidores físicos, os comandos:

```
# ssh root@192.168.1.50
# ssh root@192.168.1.51
# ssh root@192.168.1.52
# ssh root@192.168.1.53
```

No primeiro acesso, o SSH exibirá a mensagem informando que não pode estabelecer a autenticidade do *host*, digite yes para prosseguir.

```
# ssh root@192.168.1.50
The authenticity of host '192.168.1.50 (192.168.1.50)' can't be
established.
ECDSA key fingerprint is SHA256:7otpE984ztrGRzjFSM9wI9RHTXIVDMqvQTQ9P
UORvbg.
Are you sure you want to continue connecting (yes/no)? yes
```

As novas conexões não exibirão mais esta mensagem e o *live migration* está pronto para funcionar.

Criação do servidor virtual no *cluster*

Com toda a configuração do ambiente de *cluster* concluída, deve-se criar um servidor virtual, como já apresentado nos cenários anteriores, lembrando de alterar a diretiva --lvm do comando xen-create-image para usar o grupo de volumes *cluster*_vg do LVM.

```
# xen-create-image --lvm=cluster_vg --hostname=myvirtserver3
--size=50Gb --noswap --memory=2Gb --dhcp --arch=amd64 --dist=xenial
--pygrub --genpass=0 --fs=ext4 --vcpus=2 --install-method=copy
--install-source=/home/vms_modelo/ubuntu_1604LTS_NTI
```

O comando acima deve ser executado em apenas um dos servidores físicos do *cluster*. Quando ele é executado, um volume lógico do LVM é criado dentro do grupo indicado representado pelo device /dev/*cluster*_vg/myvirtserver3-disk. Este device existe apenas no servidor físico onde foi executado o comando xen-create-image, nos demais servidores do *cluster* ele ainda não existe. Para que os demais servidores passem a enxergar o device, deve-se executar o comando vgchange -ay em todos os outros servidores. Este comando faz com que os servidores releiam as informações do LVM no disco compartilhado e atualizem o seu sistema operacional com os devices do LVM que ele ainda não possui. Caso o device referente ao servidor virtual não exista em algum dos servidores físicos do *cluster* quando o servidor virtual for executado em um destes servidores físicos ocorrerá uma falha e o servidor virtual não será executado.

```
NOTA: A seção a seguir introduz o CLVM, que é o LVM em modo de
cluster. Com o CLVM não é necessário executar o comando vgchange
-ay nos demais nós, pois o próprio LVM, pelo fato de funcionar em
cluster, atualiza automaticamente as informações do novo volume nos
demais nós.
```

Ao finalizar a execução do xen-create-image, será gerado o arquivo de configuração do servidor virtual dentro do diretório /etc/xen do servidor físico no qual o comando foi executado. Devemos criar o diretório *cluster* em /etc/xen e mover o arquivo de configuração do servidor virtual para dentro dele.

```
# cd /etx/xen
# mkdir cluster
# mv vm01.cfg cluster
```

O diretório *cluster* deve ser criado em todos os servidores físicos e, consequentemente, o arquivo de configuração do servidor virtual também deve ser copiado para os demais servidores físicos que ainda não o possui.

Após garantir que o *device* do servidor virtual esteja sendo visto por todos os servidores físicos e que todos os servidores físicos possuem os arquivos de configuração dos servidores virtuais, podemos executar o comando para criação do recurso do Xen referente ao novo servidor virtual.

```
# crm configure primitive vm01 ocf:heartbeat:Xen params xmfile="/
etc/xen/cluster/vm01.cfg" node_ip_attribute="ip_livemigration" op
monitor interval="30s" timeout="30s" op migrate_from timeout="600s"
op migrate_to timeout="600s" op start timeout="90s" op stop
timeout="90s" meta target-role="started" allow-migrate="true"


NOTA: Os timeouts de migração (migrate_from e migrate_to) possuem
relação com a quantidade de memória do servidor virtual, portanto,
quanto mais memória for alocada, maior deve ser o timeout para que
haja tempo suficiente para concluir a migração. Por exemplo, uma VM
com 32GB de RAM demorou aproximadamente 18min para migrar de um nó
para o outro.
```

Este comando vai criar o recurso do Xen dentro do *cluster* que por sua vez vai se encarregar de subir o servidor virtual em um dos servidores físicos existentes. Para visualizar o novo recurso e saber em qual servidor físico ele está rodando, executar o comando abaixo:

```
# crm -1rf
Last updated: Tue Apr 15 09:10:21 2015
Last change: Mon Apr 13 11:22:24 2015 via cibadmin on server53
Stack: corosync
Current DC: server53 (53) - partition with quorum
Version: 1.1.10-42f2063
4 Nodes configured
1 Resources configured


Online: [ server50 server51 server52 server53 ]


Full list of resources:
 vm01        (ocf::heartbeat:Xen):   Started  server50
```

O comando indica que o recurso vm01, que é um servidor virtual, está sendo executado no servidor físico server50.

Comandos úteis para gerenciamento do *cluster*

```
# crm_verify -L -V → verifica se as configurações do cluster estão
corretas
# crm configure show → mostra as configurações do cluster
# crm configure edit → altera as configurações do cluster
# crm resource stop|start <nome_recurso> → para/inicia um recurso do
cluster
# crm resource cleanup <nome_recurso> → limpa os erros referentes ao
recurso
# crm_resource -D -t primitive -r <nome_recurso> → apaga o recurso
# crm node standby → coloca o servidor físico em modo standby (tem
que ser executado no servidor físico que se deseja colocar em
standby e todos os recursos são migrados para outro servidor físico
do cluster)
# crm node online → coloca o servidor físico online novamente no
cluster (tem que ser executado no servidor físico que se deseja
colocar novamente online)
```

Segue abaixo um passo a passo para execução de procedimento de manutenção em um dos servidores físicos do *cluster*, em situações onde seja necessário desligá-lo seja para efetuar uma manutenção física ou lógica.

1. Colocar o servidor físico em questão no modo standby.

```
# crm node standby
```

2. Aguardar até que todos os recursos que estavam em execução no servidor tenham sido migrados para os outros servidores do *cluster* e desligá-lo.

3. Efetuar a manutenção lógica (atualização de sistema operacional, atualização de *kernel*, etc), se houver, e desligar o servidor.

4. Efetuar a manutenção física (substituição de HD, memória, troca de peças, etc), se houver.

5. Ligar o servidor novamente. E assim que ele estiver acessível, executar o comando abaixo para colocá-lo novamente no *cluster*.

```
# crm node online
```

Cluster LVM (CLVM)

Conforme mencionado anteriormente, ao criar um servidor virtual, o volume lógico do LVM que representa o disco do servidor virtual é criado apenas no servidor físico onde o comando foi executado. Para que os demais servidores físicos passem a enxergar o volume lógico é necessário executar o comando vgchange -ay.

Com o CLVM, ao criar o servidor virtual, o volume lógico é replicado automaticamente para os demais servidores físicos do cluster, facilitando o gerenciamento. Em qualquer operação seja de inclusão, modificação ou exclusão de um volume, os demais nós são informados sobre a modificação e atualizam as informações localmente. Para usar o CLVM é necessário que os servidores físicos compartilhem a mesma área de armazenamento no *storage*. Vale ressaltar que os nós tomam conhecimento apenas das mudanças relacionadas aos metadados do LVM e não coordena o acesso aos dados, como um filesystem de cluster, por exemplo.

Para instalar o CLVM, o comando é:

```
# apt-get install clvm
```

No arquivo /etc/lvm/lvm.conf, alterar as seguintes diretivas:

```
locking_type = 3

use_lvmetad = 0
```

A primeira diretiva indica que o LVM será utilizado em modo de *cluster*, enquanto a segunda indica ao LVM que ele deverá fazer cache dos metadados, porém, o uso do lvmetad é incompatível com o modo *cluster* e por isso ele deve ser desativado. Verificar se o daemon do lvmetad está em execução, se estiver ele dever ser parado:

```
# systemctl stop lvm2-lvmetad.service
```

No arquivo /etc/default/clvm, adicionar na diretiva LVM_VGS o nome dos grupos de volume que funcionarão em *cluster*. Caso esta diretiva esteja comentada, o CLVM vai entender que todos os volumes funcionarão em *cluster*.

Não há diferença para criar um volume lógico em um ambiente de *cluster*. O único ponto que deve-se ficar atendo é durante a criação do grupo de volumes, e este deverá ser criado com a flag -cy, mostrando que o grupo será compartilhado com outros nós. Se o grupo já existir, mas sem a flag de *cluster*, pode-se alterar sua configuração. Para checar o estado atual do grupo de volumes, executar:

```
# vgs --config 'global {locking_type = 0}'
  WARNING: Locking disabled. Be careful! This could corrupt your
metadata.
  VG           #PV #LV #SN Attr   VSize  VFree
  VMSCluster   1   1   0 wz--nc 96.00M 76.00M

NOTA: O flag 'c' em Attr indica que ele está em modo cluster.
```

Caso a flag c não exista, executar o comando abaixo em apenas um nó do *cluster*. Este comando vai colocar o grupo de volumes em modo de *cluster*. Certificar que não haja nenhum volume lógico aberto para que não haja comprometimento dos dados.

```
# vgchange -cy VMSCluster
```

A seguinte mensagem de erro será exibida, pois o *daemon* do CLVM ainda não está em execução, mas basta confirmar para prosseguir.

```
connect() failed on local socket: No such file or directory
  Internal cluster locking initialisation failed.
  WARNING: Falling back to local file-based locking.
  Volume Groups with the clustered attribute will be inaccessible.
LVM cluster daemon (clvmd) is not running. Make volume group
"VMSCluster" clustered anyway? [y/n]: yes
```

Além do CLVM será necessário instalar o pacote DLM que gerencia alguns eventos do *cluster* em nível do kernel e que trabalha em conjunto com o CLVM.

```
# apt-get install dlm
```

É necessário que o DLM esteja ativo em todos os nós para que o CLVM funcione. Ambos possuem um *daemon* que pode ser inicializado e parado a qualquer momento. Iniciar o DLM em todos os nós.

```
# service dlm start
```

Quando se inicia o DLM, o sistema operacional cria três novos *device*s: /dev/misc/dlm-control, /dev/misc/dlm-monitor, /dev/misc/dlm-plock.

Antes de iniciar o CLVM é preciso modificar o arquivo /etc/init.d/clvm, pois ele checa se o arquivo /etc/*cluster*/*cluster*.conf existe e neste cenário ele não é necessário, pois o CLVM vai utilizar o arquivo de configuração do Corosync. Comentar as seguintes linhas:

```
#if [ ! -f /etc/cluster/cluster.conf ]; then
# log_failure_msg "clvmd: cluster not configured. Aborting."
# exit 0
#fi
```

Ainda no /etc/init.d/clvm, remover o caractere ! da linha 38, pois ele nega o resultado do comando que mostra informações do Corosync e entra na condição de erro. Deixar a linha conforme abaixo:

```
if /usr/sbin/corosync-quorumtool -s >/dev/null 2>&1; then
```

No Ubuntu 16.04, quando se altera algum arquivo dentro do diretório /etc/init.d, deve-se executar o comando a seguir para que as modificações sejam aplicadas e não apareça mensagem de warning:

```
# systemctl daemon-reload
```

Iniciar o *daemon* do CLVM em todos os nós:

```
# service clvm start
```

Pronto! Após ter iniciado o CLVM, pode-se verificar com o comando vgs se o grupo de volumes está ativo e em modo de *cluster*, procurando pela flag c conforme já explicado.

```
# vgs
  VG          #PV #LV #SN Attr   VSize VFree
  VMS*Cluster*   1   0   0 wz--nc 5.00t 5.00t
```

Basta criar, modificar, excluir um volume em apenas um dos nós que ele será replicado automaticamente para os demais:

```
# lvcreate -L50Gb -n testelv-disk VMSCluster
# lvrename VMSCluster testelv-disk testealt-disk
# lvremove VMSCluster/testealt-disk
```

Ao criar um servidor virtual, o comando xen-create-image cria o volume lógico que representará o disco deste servidor e consecutivamente não há mais necessidade de executar o vgchange -ay nos demais nós do *cluster*.

Referências

BUILDING a Xen guest domain using Xen-Tools. Disponível em: <http://www.virtuatopia.com/index.php/Building_a_Xen_Guest_Domain_using_Xen-Tools>. Acesso em: 15 fev. 2016.

DEBOOTSTRAPCHROOT. Disponível em: <https://wiki.ubuntu.com/DebootstrapChroot>. Acesso em: 15 fev. 2016.

DEVICE Mapper Multipathing. Disponível em: <https://help.ubuntu.com/lts/serverguide/device-mapper-multipathing.html>. Acesso em: 15 fev. 2016.

HOW TO CREATE a high availability setup with Corosync, Pacemaker, and Floating IPs on Ubuntu 14.04. Disponível em: <https://www.digitalocean.com/community/tutorials/how-to-create-a-high-availability-setup-with-corosync-pacemaker-and-floating-ips-on-ubuntu-14-04>. Acesso em: 15 fev. 2016.

iSCSI Initiator. Disponível em: <https://help.ubuntu.com/lts/serverguide/iscsi-initiator.html>. Acesso em: 15 fev. 2016.

LEVINE, S. **Red Hat Enterprise Linux 6 - logical volume manager administration - LVM administrator guide.** Disponível em: <https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/6/html-single/Logical_Volume_Manager_Administration/>. Acesso em: 15 fev. 2016.

LVM Concepts. Disponível em: <http://www.errorists.org/stuff/lvm/lvm-concepts.png>. Acesso em 17 fev. 2016.

MIGRATING a Linux server from the command line - running the sync. Disponível em: <http://articles.slicehost.com/2011/2/4/migrating-a-linux-server-from-the-command-line-running-the-sync>. Acesso em: 15 fev. 2016.

PACEMAKER. *Cluster* Labs - Pacemaker documentation. Disponível em: <http://*cluster*labs.org/doc>. Acesso em: 15 fev. 2016.

SETTING up DM-Multipath Overview. Disponível em: <https://help.ubuntu.com/lts/serverguide/multipath-setting-up-dm-multipath.html>. Disponível em:

THE XEN Project wiki. Disponível em: <https://wiki.xenproject.org>. Acesso em: 15 fev. 2016.

TIMME, F. **A beginner's guide to LVM**. Disponível em: <https://www.howtoforge.com/linux_lvm>. Acesso em: 15 fev. 2016.

XEN LIVE migration of An LVM-Based virtual machine with iSCSI On Debian Lennyhttps. Disponível em: <www.howtoforge.com/xen-live-migration-of-an-lvm-based-virtual-machine-with-iscsi-on-debian-lenny>. Acesso em: 15 fev. 2016.

Embrapa

Informática Agropecuária

MINISTÉRIO DA
AGRICULTURA, PECUÁRIA
E ABASTECIMENTO

BRASIL
GOVERNO FEDERAL

CGPE 13446