нетология

Отказоустойчивость: **Pacemaker.**

Александр
Зубарев

Александр Зубарев

Председатель цикловой комиссии “Информационной безопасности инфокоммуникационных систем”

АКТ (ф) СПбГУТ

Александр Зубарев

Вспомним материал прошлой лекции

Вопрос: Назовите примеры слабосвязанных систем.

Вспомним материал прошлой лекции

Вопрос: Назовите примеры слабосвязанных систем.

Ответ: системы с массовым параллелизмом (MPP), кластерные системы.

Вспомним материал прошлой лекции

Вопрос: Какие типы кластеров различают?

Вспомним материал прошлой лекции

Вопрос: Какие типы кластеров различают?

Ответ:

- Отказоустойчивые кластеры,
- Кластеры с балансировкой нагрузки,
- Вычислительные кластеры,
- Системы распределенных вычислений.

Предисловие

Сегодня мы поговорим о технологиях и методах построения отказоустойчивых кластеров Linux HA на основе Peacemaker и Corosync.

Рассмотрим построение кластера на основе Peacemaker и Corosync и их взаимодействие.

План занятия

1. Peacemaker
2. Heartbeat
3. Corosync
4. Distributed Replicated Block Device
5. Развертка кластера
6. Добавление ресурсов
7. Тестирование
8. Итоги
9. Домашнее задание

Peacemaker

Peacemaker

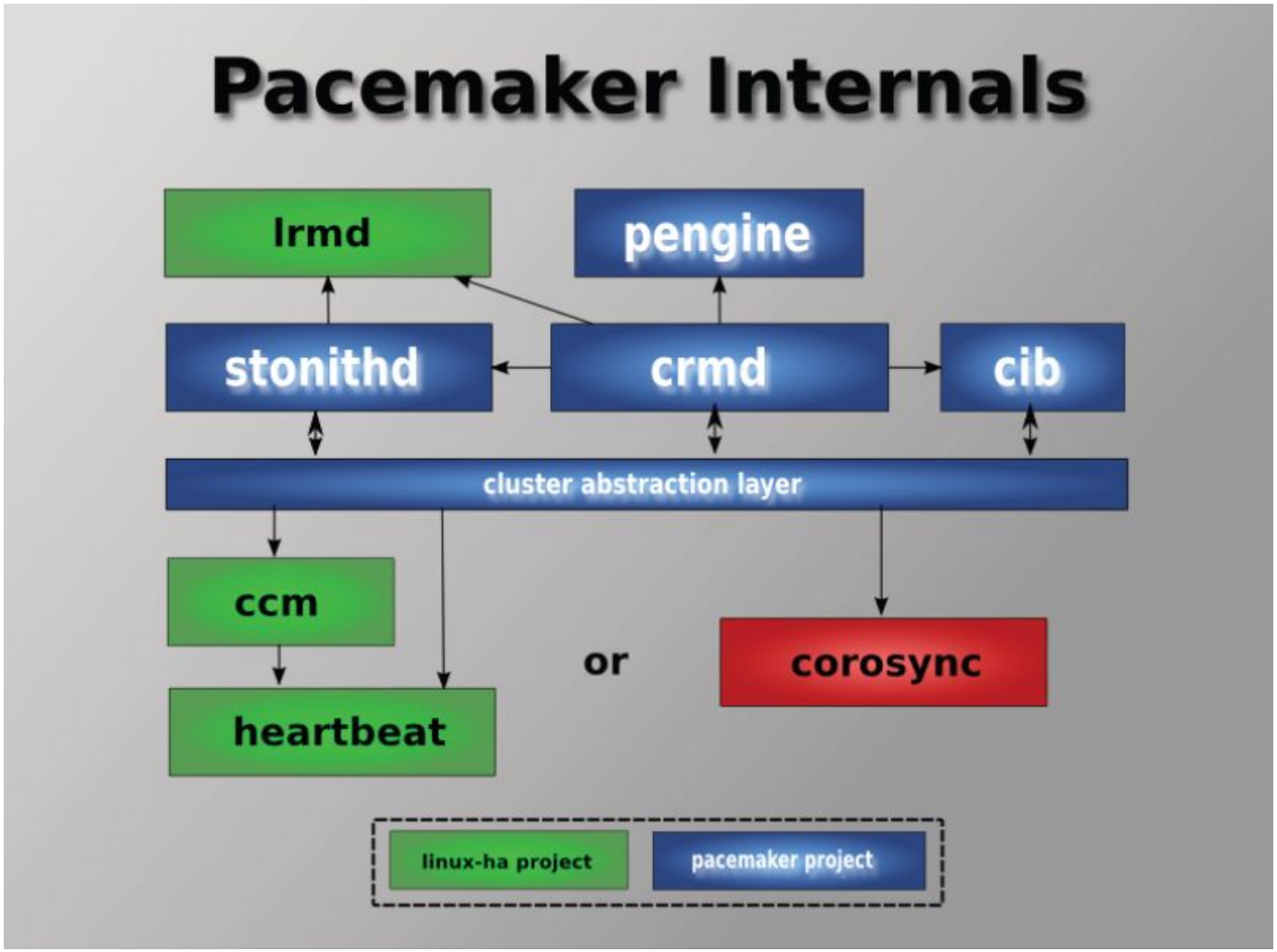
— это менеджер ресурсов кластера со следующими основными функциями:

- Обнаружение и восстановление сбоев на уровне узлов и сервисов;
- Независимость от подсистемы хранения: общий диск не требуется;
- Независимость от типов ресурсов: все, что может быть заскриптовано, может быть кластеризовано;
- Поддержка STONITH (Shoot-The-Other-Node-In-The-Head);
- ....

Peacemaker

- Поддержка кластеров любого размера;
- Поддержка и кворумных и ресурсозависимых кластеров;
- Поддержка практически любой избыточной конфигурации;
- Автоматическая репликация конфига на все узлы кластера;
- Возможность задания порядка запуска ресурсов, а также их совместимости на одном узле;
- Поддержка расширенных типов ресурсов: клонов (запущен на множестве узлов) и с дополнительными состояниями (master/slave и т.п.);
- Единый кластерный шелл (crm), унифицированный, скриптующийся.

Peacemaker


Pacemaker Internals
lrmd
pengine
stonithd
crmd
cib
cluster abstraction layer
ccm
or
corosync
heartbeat
linux-ha project
pacemaker project

Heartbeat

Heartbeat

– технология, которая позволяет проводить упреждающее техническое обслуживание и обеспечивает ощутимые преимущества в надежности и безопасности процессов при эксплуатации оборудования.

Так как устройства сами проводят собственную диагностику, интервал проведения контрольных испытаний может быть увеличен до максимальных значений.

Преимущества Heartbeat

В работе Heartbeat следует отметить следующее:

- **тестирование и документирование** каждой измерительной точки на месте без прерывания процесса;
- **постоянная самодиагностика** позволяет увеличивать интервалы между проведением контрольных испытаний;
- **диагностические сообщения** обеспечивают точные инструкции для технического обслуживания;
- **четкие и документируемые** результаты тестов, простая процедура тестирования;

Преимущества Heartbeat

- **автоматически генерируемый протокол тестирования** обеспечивает соответствие документации нормативным требованиям;
- **данные процесса и устройства** отражают тренды для упреждающего обслуживания;
- **сочетание параметров процесса и устройства** упрощают анализ для оптимизации процесса.

Corosync

Corosync

— программный продукт, позволяющий реализовать кластер серверов. Его основное назначение — знать и передавать состояние всех участников кластера.

В основе работы заложены следующие функции:

- отслеживание состояния приложений;
- оповещение приложений о смене активной ноды кластера;
- отправка одинаковых сообщений процессам на всех узлах кластера;
- предоставление доступа к базе данных с конфигурацией и статистикой, а также отправка уведомлений о ее изменениях.

Distributed Replicated Block Device

DRBD

Distributed Replicated Block Device — «распределённое реплицируемое блочное устройство».

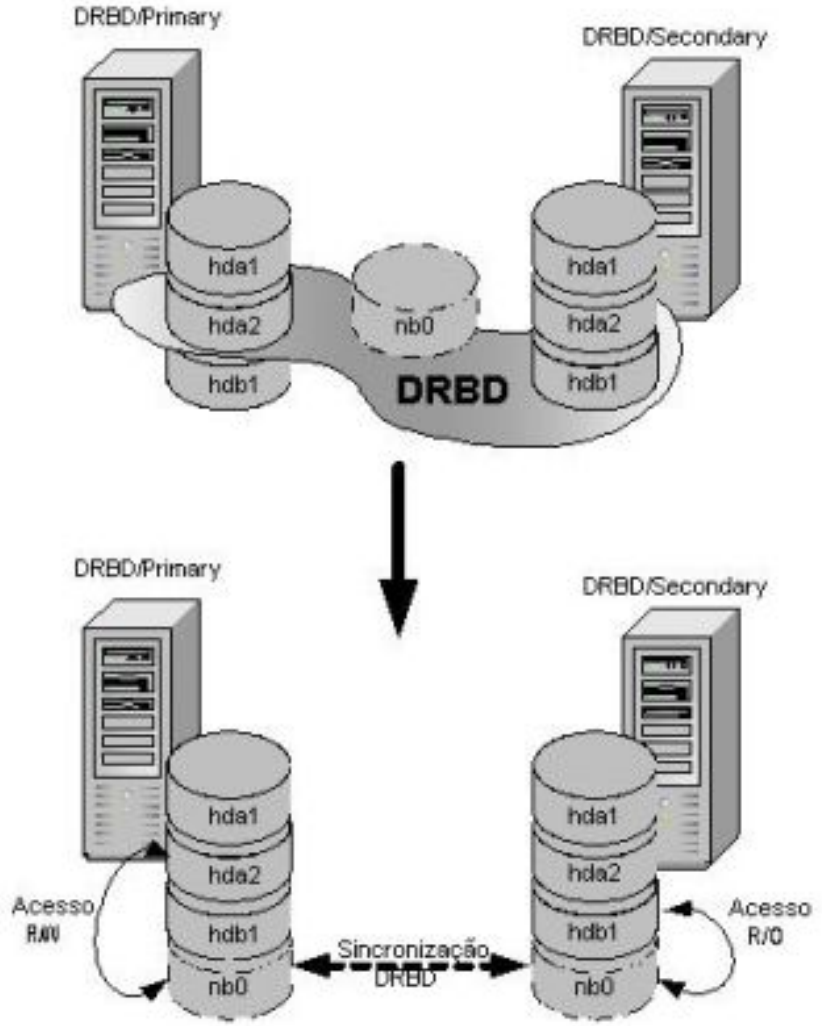
Поддерживает как синхронную, так и асинхронную репликацию.

Используется в HA Кластерах для репликации данных.

Протоколы DRBD

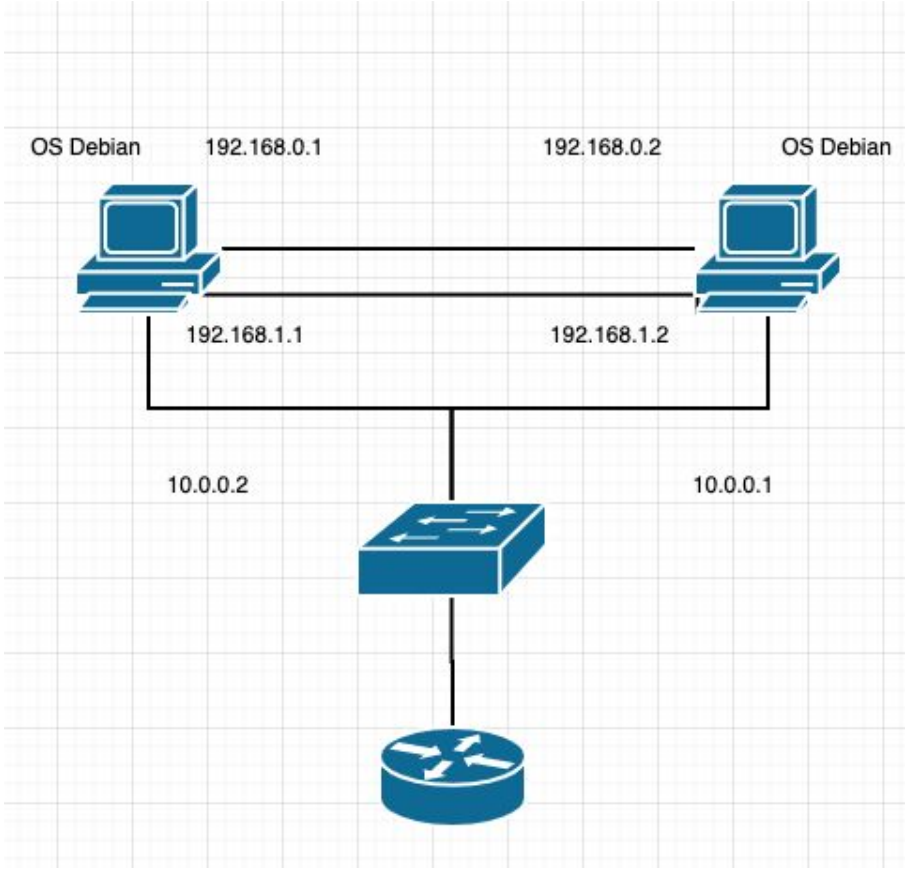
- **Протокол «C»**, операция записи считается завершённой, когда и локальный, и удалённый диски сообщают об успешном завершении записи;
- **Протокол «A»**, запись считается завершённой, когда запись завершилась на локальном устройстве и данные готовы к отправке на удалённый узел;
- **Протокол «B»**, запись считается успешной, если она завершилась на локальном устройстве, и удалённый узел подтвердил получение (но не локальную запись) данных.

Принцип работы DRBD



DRBD

Для выполнения работы требуется построить следующую топологию:



DRBD

Для репликации требуется настроить сетевые интерфейсы согласно схеме:

```
nano /etc/network/interfaces
```

```
auto lo
iface lo inet loopback

auto enp0s3
iface enp0s3 inet dhcp

auto enp0s8
iface enp0s8 inet static
    address 192.168.0.1
    netmask 255.255.255.0
auto enp0s9
iface enp0s9 inet static
    address 192.168.1.1
    netmask 255.255.255.0
```

Интерфейс **enp0s3** служит для доступа к отдельным нодам кластера.

enp0s8 — это heartbeat-интерфейс.

enp0s9 используется для синхронизации DRBD-ресурсов.

DRBD

Установка DRBD:

```
sudo apt install drbd-utils
```

Подключаем DRBD к модулям ядра:

```
sudo modprobe drbd
```

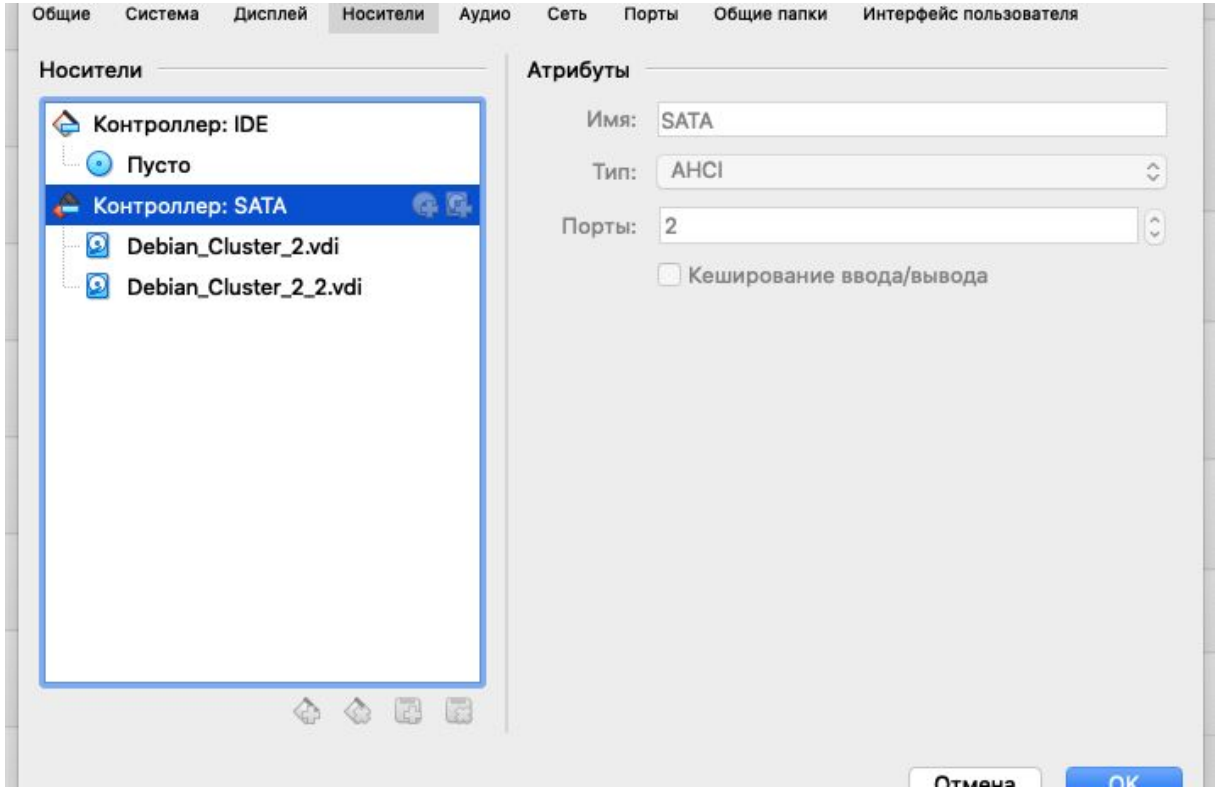
Добавляем в загрузки системы:

```
echo “drbd” >> /etc/modules
```

Установка должна быть на двух машинах.

DRBD

Для выполнения самого накопителя требуется добавить новый диск к виртуальной машине:



DRBD

Следующим шагом требуется проверить созданные накопители:

```
ls /dev |grep sd
```

Вывод операции:

```
sda, sda1, sda2, sda3, sdb
```

Выполнить:

```
fdisk /dev/sdb
```

DRBD

Команды пошагово:

n — создание диска — либо primary, либо extension.

Создаем primary — ключ p.

Остальное по вашему усмотрению — по умолчанию.

Enter — Enter готово.

DRBD

Создаем логические разделы:

```
pvcreate /dev/sdb1
vgcreate vg0 /dev/sdb1
lvcreate -L3G -n www vg0
lvcreate -L3G -n mysql vg0
```

DRBD

Создаем конфигурационные файлы

`/etc/drbd.d/www.res` и `/etc/drbd.d/mysql.res`

следующего содержания:

```
resource www {
    protocol C;

    disk {
        fencing resource-only;
    }

    handlers {
        fence-peer
"/usr/lib/drbd/crm-fence-peer.sh";
        after-resync-target
"/usr/lib/drbd/crm-unfence-peer.sh";
    }
syncer {
        rate 110M;
    }
    on nodeTwo
    {
        device /dev/drbd2;
        disk /dev/vg0/www;
        address 192.168.1.2:7794;
        meta-disk internal;
    }
    on nodeOne
    {
        device /dev/drbd2;
        disk /dev/vg0/www;
        address 192.168.1.1:7794;
        meta-disk internal;
    }
}
```

DRBD

После чего на обоих серверах выполняем:

```
drbdadm create-md www
drbdadm create-md mysql
drbdadm up all
```

Выполнить на первой ноде:

```
drbdadm primary --force www
drbdadm primary --force mysql
```

На второй ноде:

```
drbdadm secondary  www
```

DRBD

Теперь требуется подключить разделы и проверить репликацию:

```
mkdir /mnt/www
mkdir /mnt/mysql
mount /dev/drbd0 /mnt/www
mount /dev/drbd2 /mnt/mysql
```

на обоих нодах.

Развертка кластера

Развертка кластера

Установка pacemaker:

```
sudo apt install pacemaker corosync pcs
```

На обоих серверах ставим:

```
systemctl enable pcsd
```

Развертка кластера

Обязательно прописать в файле ***/etc/hosts*** dns суффиксы нодов:

```
127.0.0.1    localhost
192.168.0.1    nodeone
192.168.0.2    nodetwo
192.168.1.1  nodeone
192.168.1.2  nodetwo
# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Развертка кластера

Для управления кластером рекомендуется пользоваться утилитой **pcs**. При установке pacemaker автоматически будет создан пользователь hacluster. Для использования pcs, а также для доступа в веб-интерфейс, нужно задать пароль пользователю hacluster:

```
passwd hacluster
```

Развертка кластера

Запуск сервиса:

```
service pcsd start
```

Настраиваем аутентификацию (на одном узле):

```
pcs host auth <Сервер_1> <Сервер_2>
Username: hacluster
Password:
<Сервер_1>: Authorized
<Сервер_2>: Authorized
```

Развертка кластера

Следующим шагом запускаем ноды:

```
pcs cluster setup newCluster nodeone nodetwo
```

Важно: предварительно проверьте включена ли служба pcsd на обоих кластерах:

```
service --status-all
```

Развертка кластера

При выполнение скрипта мы должны увидеть следующее:

```
Destroying cluster on hosts: 'nodeone', 'nodetwo'...
nodeone: Successfully destroyed cluster
nodetwo: Successfully destroyed cluster
Requesting remove 'pcsd settings' from 'nodeone', 'nodetwo'
nodetwo: successful removal of the file 'pcsd settings'
nodeone: successful removal of the file 'pcsd settings'
Sending 'corosync authkey', 'pacemaker authkey' to 'nodeone', 'nodetwo'
nodetwo: successful distribution of the file 'corosync authkey'
nodetwo: successful distribution of the file 'pacemaker authkey'
nodeone: successful distribution of the file 'corosync authkey'
nodeone: successful distribution of the file 'pacemaker authkey'
Synchronizing pcsd SSL certificates on nodes 'nodeone', 'nodetwo'...
nodeone: Success
nodetwo: Success
Sending 'corosync.conf' to 'nodeone', 'nodetwo'
nodeone: successful distribution of the file 'corosync.conf'
nodetwo: successful distribution of the file 'corosync.conf'
```

Развертка кластера

Следующим шагом устанавливаем активными все ноды:

```
pcs cluster enable -all
```

Получаем:

```
root@nodeOne:/etc/corosync# pcs cluster enable --all
nodeOne: Cluster Enabled
nodeTwo: Cluster Enabled
```

Развертка кластера

Добавление запуска служб при старте системы на главном сервере (кластере), если вдруг не включены:

```
systemctl enable pcsd
systemctl enable corosync
systemctl enable pacemaker
```

Развертка кластера

Проверяем командой:

```
pcs status
```

Результат:

```
Current DC: nodeone (version 2.0.1-9e909a5bdd) - partition with quorum
Last updated: Mon Jul 12 16:20:55 2021
Last change: Mon Jul 12 16:20:54 2021 by hacluster via crmd on nodeone

2 nodes configured
0 resources configured

Online: [ nodeone nodetwo ]

No resources


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

Добавление ресурсов

Добавление ресурсов

Первым делом установим сервисы (для демонстрации установим веб-сервис):

```
apt install apache2
```

Добавим для репликации кластера в файл следующие строки:

```
nano /etc/apache2/apache2.conf
```

После выполнить:

```
service apache2 restart
```

```
<Location /server-status>
    SetHandler server-status
    Require local
</Location>
```

Добавление ресурсов

Создадим на будущем ресурсе www следующие директории:

```
cd /mnt/www

mkdir html

mkdir cgi-bin

mkdir error
```

Создать файл и внести следующее содержание:

```
nano index.html

<html> <body>Hello, Welcome to main Debian High Availability
Cluster</body></html>
```

Добавление ресурсов

Создадим виртуальный ip для heartbeat:

```
pcs resource create virtual_ip ocf:heartbeat:IPaddr2 ip=192.168.0.10 cidr_netmask=24 op monitor interval=60s
```

Добавим ресурсы drbd:

```
pcs -f drbd_cfg resource create WebDate ocf:linbit:drbd drbd_resource=/dev/drbd2 op monitor interval=60s
pcs -f drbd_cfg resource create MysqlDate ocf:linbit:drbd drbd_resource=/dev/drbd0 op monitor interval=60s
```

Добавление ресурсов

Вводим кластер второй и репликацию:

```
pcs -f drbd_cfg resource promotable WebDate promoted-max=1
promoted-node-max=1 clone-max=2 clone-node-max=1 notify=true

pcs -f drbd_cfg resource promotable MysqlDate promoted-max=1
promoted-node-max=1 clone-max=2 clone-node-max=1 notify=true
```

Добавление ресурсов

В итоге должно быть следующее:

```
root@nodeOne:~# pcs -f drbd_cfg resource
 virtual_ip     (ocf::heartbeat:IPaddr2):       Stopped
 Clone Set: WebDate-clone [WebDate] (promotable)
     Stopped: [ nodeone nodetwo ]
 Clone Set: MysqlDate-clone [MysqlDate] (promotable)
     Stopped: [ nodeone nodetwo ]
```

После чего вводим:

```
psc cluster cib-push drbd_cfg
```

и загружаем конфигурацию в систему.

Тестирование

Тестирование drbd

Для проверки понадобится http или sql сервер.

Размонтируем на Primary ноде накопитель и подключим его на второй ноде:

```
umount /dev/drbd0
umount /dev/drbd2
mount /dev/drbd0 /mnt/mysql
mount /dev/drbd2 /mnt/www
```

Подключаем режим primary и монтируем их, проверяем данные должны быть в полном объеме:

```
drbdadm primary mysql
drbdadm primary www
mount /dev/drbd0 /mnt/mysql & mount /dev/drbd2 /mnt/mysql
```

Тестирование Pacemaker

Возвращаем все к начальному состоянию и выключаем ноду (можно сервисами) — все должно произойти автоматом:

```
pcs cluster status
```

Результат покажет, что репликации нет, как и нода.

Итоги

Итоги

Сегодня мы рассмотрели как создавать репликацию хранилищ, реализацию pacemaker + corosync. Создали HA кластер.

Рассмотрели особенности конфигурирования высоконагруженных серверов.

Домашнее задание

Домашнее задание

Давайте посмотрим ваше домашнее задание.

- Вопросы по домашней работе задавайте **в чате** мессенджера Slack.
- Задачи можно сдавать **по частям**.
- Зачёт по домашней работе проставляется после того, как **приняты все задачи**.

нетология

Задавайте вопросы и пишите отзыв о лекции!

Александр Зубарев

Александр Зубарев