Автор:
29.12.12 21:31 - Последнее обновление 29.12.12 21:35

Здравствуйте!

Хочу рассказать о том, как мы используем у себя Proxmox Virtual Environment .

Я не буду описывать установку и первоначальную настройку — Proxmox очень прост и приятен и в установке, и в настройке. Расскажу о том, как мы используем систему в кластерном окружении.

Для полноценной работы кластера необходимо, чтобы управление виртуальной машиной оперативно могли брать на себя разные хосты кластера. Данные виртуалок при этом не должны никуда копироваться. То есть все хосты кластера должны иметь доступ к данным конкретной машины, или, иными словами, все хосты кластера должны работать с единым хранилищем данных, в рамках которого работает конкретный набор виртуальных машин.

Proxmox работает с двумя типами виртуализации: уровня операционной системы, на основе OpenVZ и аппаратной, на основе KVM. В этих двух типах используется разный подход к утилизации дискового пространства. Если в случае с OpenVZ-контейнерами работа с диском виртуальной машины осуществляется на уровне файловой системы хоста, то в случае с KVM-машинами используется образ диска, в котором находится собственная файловая система виртуальной машины. Операционная система хоста не заботится о размещении данных внутри KVM-диска. Этим занимается гипервизор. При организации работы кластера вариант с образами диска реализуется проще, чем работа с файловой системой. Данные KVM-машины с точки зрения операционной системы хоста могут просто находиться "где-то" в хранилище. Эта концепция замечательно ложится на схему работы LVM, когда образ KVM-диска находится внутри логического тома.

В случае же с OpenVZ мы имеем дело с файловой системой, а не просто с областями данных на Shared Storage. Нам нужна полноценная кластерная файловая система.

О кластерной файловой системе речь пойдет не в этой части статьи. О работе с KVM — тоже. Сейчас поговорим о подготовке кластера к работе с общим хранилищем.

Хочу сразу сказать, что коммерческую нагрузку мы на кластер не даем, и не планируем. Система используется для внутренних нужд, которых у нас предостаточно. Как я писал в предыдущей статье , коммерческую нагрузку мы постепенно переносим в vCloud, а на освобождающихся мощностях разворачиваем Proxmox.

В нашем случае проблема организации общего хранилища сводится к двум аспектам:

1. У нас есть блочное устройство, отдаваемое по сети, к которому будут иметь доступ несколько хостов одновременно. Для того, чтобы эти хосты не подрались за место на устройстве, нам нужен CLVM —Clustered Logical Volume Manager. Это то же самое, что LVM, только Clustered. Благодаря CLVM каждый хост имеет актуальную информацию (и может ее безопасно изменять, без риска нарушить целостность) о состоянии LVM-томов на Shared Storage. Логические тома в CLVM живут точно так же, как в обычномLVM. В логических томах находятся либо KVM-образы, либо кластерная FS.
2. В случае с OpenVZ у нас есть логический том, на котором расположена файловая система. Одновременная работа нескольких машин с некластерной файловой системой ведет к неминуемым ошибкам в работе всего — это лебедь, рак и щука, только хуже. Файловая система обязательно должна знать о том, что она живет на общем ресурсе, и уметь работать в таком режиме.

Создание кластера и подключение к нему нод я не буду описывать. Об этом можно почитать, например, на сайте разработчиков . Кстати, их база знаний довольно обширна и информативна.

Мы работаем с Proxmox версии 2.2. Будем считать, что кластер у нас настроен, и работает.

**Мы же займемся настройкой fenced-демона**

Специфика кластерной среды обязывает ноды кластера следовать некоторым правилам поведения. Нельзя просто так взять и начать писать на устройство. Сначала надо спросить разрешения. Контролем этого занимаются несколько узлов кластера — CMAN (Cluster manager), DLM (Distributed lock manager) и Fenced.Fenced-демон выполняет роль вышибалы. Если с точки зрения кластера какая-то нода начинает вести себя неадекватно — общение с хранилищем в кластере замирает, а fenced пытается отключить сбойную ноду от кластера.

Fencing — это процесс исключения ноды из работы с кластерным хранилищем. Так как неадекватная машина может и не отреагировать на просьбы уйти по-хорошему, отлучение ноды производится с использованием внешних по отношению к кластеру сил. Для общения с этими силами используются жрецы специально обученные fence-агенты. Как правило, fencing сводится к обесточиванию ноды, после чего кластер переводит

дух, и возобновляет работу.

В роли внешних сил может выступать любое оборудование, способное по сигналу залепить дуло обесточить, или изолировать машину от сети. Чаще всего в роли таких устройств выступают промышленные блоки питания, либо консоли управления серверов. Мы у себя используем оборудование HP. Fencing производится с использованием iLO-карточек.

До тех пор, пока fenced-демон не получил от агента подтверждения о том, что нода благополучно "fenced" — все операции ввода-вывода в кластере будут приостановлены. Это сделано для того, чтобы свести к минимуму риск порчи данных в хранилище. Так как сбойная нода перестала следовать общепринятым правилам поведения, от нее можно ожидать чего угодно. Например, попыток несанкционированной (и не протоколируемой) записи на диск. Поэтому любое общение с хранилищем в этой ситуации увеличивает риск порчи данных.

Если же для ноды не настроен fence-агент, то fenced не сможет пнуть ее в случае проблем, и кластер будет находиться в замороженном состоянии до тех пор, пока ситуация не разрешится. В такой ситуации есть несколько сценариев дальнейшего развития событий:

1. Нода может прийти в себя, вернуться в кластер и сказать, что она больше так не будет. Ее пустят, и работа кластера будет возобновлена.
2. Нода может перезагрузиться (или ее кто-то перезагрузит), и попроситься в кластер. Факт новой попытки подключения к кластеру считается сигналом о том, что нода исправна, и можно продолжать работу.
3. Нода может умереть. Эта ситуация требует ручного вмешательства. Нужно дать понять fenced-демону, что нода уже не представляет опасности для кластера. И вообще, может быть, больше не вернется. Для этой цели существует утилита "fence_ack_manual". При этом оператор берет на себя ответственность за принятие решения о возобновлении работы кластера.

Если же хост завершает работу в штатном режиме — он просто просит вычеркнуть себя из fence-домена, после чего теряет возможность общаться с хранилищем.

Наличие хоста в fence-домене является необходимым условием для проведения любых операций с общим хранилищем при помощи кластерного ПО.

Рассмотрим настройку fenced с использованием агента "fence-ilo" (настройка производится на каждой ноде кластера):

В файле /etc/default/redhat-cluster-pve выставляем

```
FENCE_JOIN="yes"
```

Теперь при старте системы нода будет подключаться к fence-домену. Перезагружаться мы не хотим, поэтому добавляем ноду к домену вручную:

```
root@srv03-vmx-02:~# fence_tool join
```

Посмотреть состояние fenced-демона можно так:

```
root@srv03-vmx-02:~# fence_tool ls  fence domain  member count  4  victim count  0  victim now  0  master nodeid 1  wait state  none  members  1 2 3 4
```

**Тестируем fence-agent**

Для разных версий iLO существуют разные агенты:

```
root@tpve01:~# fence_ilo  fence_ilo  fence_ilo2  fence_ilo3  fence_ilo_mp
```

Для начала — опросим через iLO состояние интересующей нас ноды:

```
root@tpve01:~# fence_ilo -a ILO_IP -l LOGIN -p PASSWORD -o status  Status: ON
```

Status: ON. Можно вместо "-o status" сказать "-o reboot". Подопытная машина получит ресетом поддых.

Таким же образом проверяем работоспособность iLO на всех нодах.

Теперь настроим кластер для корректной работы fence-агентов. Про настройку fenced в Proxmox есть хорошая статья , и я не буду здесь пересказывать то, что там написано,

приведу лишь конечный конфиг нашего кластера:

```
root@tpve01:~# cat /etc/pve/cluster.conf.new <?xml version="1.0"?> <cluster name="tpve" config_version="5"> <cman keyfile="/var/lib/pve-cluster/corosync.authkey"> </cman> <fencedevices> <fencedevice agent="fence_ilo" ipaddr="IP_ILO_TPVE01" name="tpve01" passwd_script="/usr/local/pvesync/ilo_pass/tpve01" login="LOGIN"/> <fencedevice agent="fence_ilo" ipaddr="IP_ILO_TPVE02" name="tpve02" passwd_script="/usr/local/pvesync/ilo_pass/tpve02" login="LOGIN"/> </fencedevices> <clusternodes> <clusternode name="tpve01" votes="1" nodeid="1"> <fence> <method name="power"> <device name="tpve01"/> </method> </fence> </clusternode> <clusternode name="tpve02" votes="1" nodeid="2"> <fence> <method name="power"> <device name="tpve02"/> </method> </fence> </clusternode> <clusternode name="tpve03" votes="1" nodeid="3"/> </clusternodes> </cluster>
```

В нашем примере нода "tpve03" не имеет настроенного fence-агента, и при проблемах с ней конфликт придется разрешать вручную.

Чтобы не светить в конфиге пароли от iLO, вместо пароля в настройках агента указывается параметр

```
passwd_script="/usr/local/pvesync/ilo_pass/tpve01"
```

Это путь к скрипту, который выдает пароль. Скрипт примитивный:

```
root@tpve01:~# cat /usr/local/pvesync/ilo_pass/tpve01 #!/bin/bash echo "DERPAROL"
```

Эти скрипты должны существовать на всех нодах кластера.

После того, как все изменения в конфигурацию кластера внесены и отвалидированы — применяем наш конфиг. В веб-интерфейсе Proxmox идем в настройки HA и говорим "Активировать". Если все произошло без ошибок, а серийный номер конфигурации кластера был увеличен, то изменения вступят в силу, и можно попробовать повыпинывать ноды. Вообще, крайне рекомендуется устроить фенсинг каждой ноде кластера для того, чтобы быть уверенным, что все действительно работает.

Для начала пнем ноду руками:

```
root@tpve01:~# fence_node tpve02  fence tpve02 success
```

Нода словила ресет.

Теперь попробуем сэмулировать проблему. На одной из нод складываем сетевой интерфейс, по которому происходит общение узлов кластера между собой:

```
root@tpve02:~# ifdown vmbr0
```

Через какое-то время на живой ноде опрос состояния fenced-демона покажет следующее:

```
root@tpve01:~# fence_tool ls  fence domain  member count  2  victim count  1  victim now    2
master nodeid 1  wait state    fencing  members       1 2 3
```

"wait state fencing" говорит о том, что прямо сейчас происходит исключение проблемной ноды из кластера, ифenced-демон ждет вестей от fence-агента.

После получения подтверждения от агента:

```
root@tpve01:~# fence_tool ls  fence domain  member count  2  victim count  0  victim now    0
master nodeid 1  wait state    none  members       1 3
```

Ноду убили, работаем дальше.

Когда нода поднимется, она подключится к кластеру.

Теперь наш кластер готов к работе с общим хранилищем.

Пожалуй, на этом остановимся. В следующей статье, которая выйдет, видимо, уже в январе, я расскажу о подключении хранилища и о работе с ним.

Автор:
29.12.12 21:31 - Последнее обновление 29.12.12 21:35

Источник: http://habrahabr.ru/company/alawar/blog/163297/

ссылка на материал: http://thin.kiev.ua/linux/39-linux/718-proxmox-fencing.html

{jcomments on}