Работа с OCFS
в платформе виртуализации РУСТЭК

2022

Содержание

1 Увеличение файловой системы

При увеличении размера луна необходимо увеличить размер файловой системы (ФС) для отображения нового пространства.

Увеличение размера кластерной ФС OCFS2 возможно на лету, без остановки кластера.

Для начала убедимся, что блочное устройство увеличилось и операционная система (ОС) видит весь требуемый размер. Для этого на узле с ролью *Управление дисками* выполним команду, показанную на рисунке 1.

```
# lsblk -f  | grep ocfs2
sdb                                 ocfs2         rustack0-0     f9bdf0f3-f392-4b79-94ff-508f684b6054
└─360000000000000000e00000000020001 ocfs2         rustack0-0     f9bdf0f3-f392-4b79-94ff-
508f684b6054   260.6G    11% /mnt/ocfs2-360000000000000000e00000000020001
sdc                                 ocfs2         rustack0-0     f9bdf0f3-f392-4b79-94ff-508f684b6054
└─360000000000000000e00000000020001 ocfs2         rustack0-0     f9bdf0f3-f392-4b79-94ff-
508f684b6054   260.6G    11% /mnt/ocfs2-360000000000000000e00000000020001
sdd                                 ocfs2         rustack0-1     a9253df7-1ac0-44ca-bfad-ee3aba095b6a
└─360000000000000000e00000000030001 ocfs2         rustack0-1     a9253df7-1ac0-44ca-bfad-
ee3aba095b6a    700.9G    21% /mnt/ocfs2-360000000000000000e00000000030001
sde                                 ocfs2         rustack0-1     a9253df7-1ac0-44ca-bfad-ee3aba095b6a
└─360000000000000000e00000000030001 ocfs2         rustack0-1     a9253df7-1ac0-44ca-bfad-
ee3aba095b6a    700.9G    21% /mnt/ocfs2-360000000000000000e00000000030001
```

Рисунок 1. Команда для ОС

В случае подключения блочного устройства по протоколу iSCSI возможно потребуется перечитать таргеты для обновления информации со стороны ОС на узлах, где используется ФС, с помощью команды на рисунке 2.

```
# iscsiadm -m node --targetname target_name -R
```

Рисунок 2. Команда для обновления информации со стороны ОС на узлах

Для рескана iSCSI на доступных узлах с ролями "Управление дисками" и "Вычислительный узел" можно воспользоваться командой:

```
ansible –i /var/lib/rustack-ansible/inventory.yml '!unreachable,cinder,compute'-m shell -a 'iscsiadm -m node -R'
```

Далее увеличим размер ФС, выполнив команду на любом узле с ролью "Управление дисками" на рисунке 3.

```
# tunefs.ocfs2 -S /dev/sdb
tunefs.ocfs2 1.8.7
Changing volume size from 1048572 blocks to 2097148 blocks
Proceed (y/N): y
Resized volume
Wrote Superblock
```

Рисунок 3. Команда для увеличения размера ФС

2 Подключение дополнительных лунов (бэкендов)

Для автоматического подключения дополнительных лунов необходимо перечислить их в поле **Список WWID для OCFS2** Конфигуратора и далее **Применить конфигурацию РУСТЭК**.

При наличии типа диска по-умолчанию "ocfs2" новый лун будет добавлен в данный тип диска автоматически при добавлении в поле настроек **Список WWID для OCFS2** и прогоне Конфигуратора.

Для ручного добавления **нового бэкенда** необходимо создать файл конфигурации с расширением .conf по пути /etc/openstack/cinder_backends.

Заголовок секции конфигурации должен совпадать с именем файла. Пример с **mybackend** (/etc/openstack/cinder_backends/mybackend.conf) представлен на рисунке 4.

/etc/openstack/cinder_backends/mybackend.conf

```
[mybackend]
backend_host = ohost1
volume_backend_name = newocfs
volume_driver = crutches.cinder.volume.drivers.sharedfs.SharedFS
nas_secure_file_permissions = true
nas_secure_file_operations = true
volume_clear = none
volume_clear_ionice = -c3
sharedfs_shares_config = /etc/cinder/mybackend_ocfs_shares
sharedfs_sparsed_volumes = true
sharedfs_reflink_cmd = cp --reflink
```

Рисунок 4. Пример с mybackend

Настройки в файле зависят от выбранного драйвера (volume_driver) бэкенда. Для пояснения настроек обратитесь к документации.

Права на файл конфигурации должны принадлежать пользователю и группе cinder. Для установки прав можно воспользоваться командой:

```
host1 ~ # chown -R cinder.cinder /etc/openstack/cinder_backends/*
host1 ~ # chmod -R 0640 /etc/openstack/cinder_backends/*
```

Далее необходимо **Применить конфигурацию РУСТЭК** через Конфигуратор. Если в новом бэкенде также используется новый **volume_backend_name**, то необходимо его связать с существующим или новым дисковым типом. Подробнее в разделе 3.

3 Создание дополнительных дисковых типов

Создать дополнительный дисковый тип возможно только из CLI, используя команду на рисунке 5.

```
host1 ~ # openstack volume type create mynewtype
```

Рисунок 5. Команда для создания дополнительного дискового типа

Далее новый дисковый тип необходимо связать с бэкендом, используя переменную **volume_backend_name** из конфигурационного файла бэкенда с помощью команды на рисунке 6.

```
host1 ~ # openstack volume type set lvm --property volume_backend_name=newocfs
```

Рисунок 6. Дополнительная команда

4 Объединение нескольких лунов (бэкендов) в один дисковый тип

Для объединения нескольких бэкендов в один дисковый тип необходимо перечислить точки монтирования данных лунов в файле, на который ссылается переменная **sharedfs_shares_config** (рисунок 7).

Содержимое /etc/cinder/mybackend_ocfs_shares

```
/mnt/ocfs2-3600000000000000000e00000000020001
/mnt/ocfs2-3600000000000000000e00000000030003
```

Рисунок 7. Команды для объединения лунов

Данный файл необходимо изменить на всех узлах с ролью **Управление дисками**. В данном случае диски ВМ будут создаваться на каждом из перечисленных лунов по очереди.

Если потребуется работа кэширования для всех лунов или различные (более гибкие) настройки балансировки между лунами, то необходимо создать несколько конфигурационных файлов для каждого бэкенда, объединённых одним **volume_backend_name** (рисунки 8-11).

/etc/openstack/cinder_backends/mybackend1.conf

```
[mybackend1]
backend_host = ohost1
volume_backend_name = newocfs
volume_driver = crutches.cinder.volume.drivers.sharedfs.SharedFS
nas_secure_file_permissions = true
nas_secure_file_operations = true
volume_clear = none
volume_clear_ionice = -c3
sharedfs_shares_config = /etc/cinder/mybackend1_ocfs_shares
sharedfs_sparsed_volumes = true
sharedfs_reflink_cmd = cp --reflink
```

Рисунок 8. Команды для /etc/openstack/cinder_backends/mybackend1.conf

Содержимое /etc/cinder/mybackend1_ocfs_shares

```
/mnt/ocfs2-360000000000000000e00000000020001
```

Рисунок 9. Команды для /etc/cinder/mybackend1_ocfs_shares

/etc/openstack/cinder_backends/mybackend2.conf

```
[mybackend2]
backend_host = ohost1
volume_backend_name = newocfs
volume_driver = crutches.cinder.volume.drivers.sharedfs.SharedFS
nas_secure_file_permissions = true
nas_secure_file_operations = true
volume_clear = none
volume_clear_ionice = -c3
sharedfs_shares_config = /etc/cinder/mybackend2_ocfs_shares
sharedfs_sparsed_volumes = true
sharedfs_reflink_cmd = cp --reflink
```

Рисунок 10. Команды для /etc/openstack/cinder_backends/mybackend2.conf

Содержимое /etc/cinder/mybackend2_ocfs_shares

```
/mnt/ocfs2-3600000000000000000e00000000030003
```

Рисунок 11. Команды для /etc/cinder/mybackend2_ocfs_shares

Обратите внимание, что в представленном примере различаются имена файла, заголовок конфигурации и файлы, описывающие луны **sharedfs_shares_config**. Переменная **volume_backend_name** в двух файлах одинаковая. При такой настройке в каждый бэкенд включается по одному луну, в дальнейшем эти бэкенды объединяются в один тип диска.

5 Настройка балансировки

Для различной балансировки между бэкендами используются различные стратегии и свойства драйвера. По умолчанию используется стратегия **CapacityWeigher**, которая взвешивает узлы по их виртуальной или фактической свободной емкости. В случае тонких дисков бэкенды взвешиваются по их виртуальной свободной емкости, рассчитанной как общая емкость, умноженная на коэффициент переподписки за вычетом выделенной емкости. В случае толстых дисков бэкенды взвешиваются по их фактической свободной емкости с учетом зарезервированного пространства.

Для примера будем использовать те же настройки бэкендов в предыдущих разделах и поставим задачу размазывать диски ВМ по двум лунам (бэкендам) в зависимости от размера диска. Диски с (логическим) размером равным или менее 5 Гб будут создаваться на mybackend2 до исчерпания места на бэкенде, остальные на mybackend1.

Для начала переопределим стратегии и правила балансировки в любом конфиге бэкенда, добавив секцию (рисунок 12).

/etc/openstack/cinder_backends/mybackend1.conf

```
[DEFAULT]
scheduler_default_weighers = GoodnessWeigher
```

Рисунок 12. Команда для /etc/openstack/cinder_backends/mybackend1.conf

Далее в настройки бэкенда mybackend1.conf в секцию [mybackend1] добавим строку (рисунок 13).

/etc/openstack/cinder_backends/mybackend1.conf

```
goodness_function = "50"scheduler_default_weighers = GoodnessWeigher
```

Рисунок 13. Команда для /etc/openstack/cinder_backends/mybackend1.conf

В настройки бэкенда mybackend2.conf в секцию [mybackend2] добавим строку (рисунок 14).

/etc/openstack/cinder_backends/mybackend1.conf

```
goodness_function = "(volume.size <= 5) ? 100 : 55"
```

Рисунок 14. Команда для /etc/openstack/cinder_backends/mybackend1.conf

Теперь mybackend2 при создании диска объемом меньшим или равным 5 Гб будет иметь приоритет 55 из 100 над бэкендом mybackend1, который имеет приоритет 50 из 100.

Перезапустим работающие процессы **cinder-volume** и **cinder-scheduler** на узлах платформы на всех узлах с ролью **Управление дисками** или воспользуемся пунктом **Применить конфигурацию РУСТЭК** через Конфигуратор.

6 Настройка кэша образов

Для настройки кэша образов в конфигурацию бэкенда при её отсутствии необходимо добавить секцию [DEFAULT] следующего содержания (рисунок 15).

/etc/openstack/cinder_backends/mybackend1.conf

```
[DEFAULT]
cinder_internal_tenant_project_id = 535f076c4dc04a0cba94ce7ed985eb58
cinder_internal_tenant_user_id = 1971fa645626446fae1b790d1ecd440d
```

Рисунок 15. Команды для настройки кэша

Значение переменной **cinder_internal_tenant_project_id** задаётся ID проекта service, который можно посмотреть в выводе команды, показанной на рисунке 16.

```
# openstack --os-cloud rustack_system project show service -c id -f value
535f076c4dc04a0cba94ce7ed985eb58
```

Рисунок 16. Команда для ID проекта service

Значение переменной **cinder_internal_tenant_user_id** задаётся ID пользователя glance, который можно посмотреть в выводе команды, показанной на рисунке 17.

```
# openstack --os-cloud rustack_system user show glance -c id -f value
1971fa645626446fae1b790d1ecd440d
```

Рисунок 17. Команда для ID пользователя glance

Затем в конфиге бэкенда в секции настроек (в примере [mybackend]) нужно активировать кэширование, установив переменную **image_volume_cache_enabled**. При необходимости можно перечислить остальные настройки кэширования (рисунок 18).

```
[mybackend]
...вырезано...
image_volume_cache_enabled = True
image_volume_cache_max_size_gb = 200
image_volume_cache_max_count = 50
```

Рисунок 18. Команды для настройки кэширования

7 Настройка ISCSI на определенном интерфейсе

Для настройки ISCSI с использованием определенного сетевого интерфейса необходимо его сконфигурировать.

В конфигурационном файле /etc/conf.d/net пример конфигурации с двумя интерфейсами eno16780032 и eno16780033 в active-backup и vlan 1111 (рисунок 19).

```
config_eno16780032="null"
config_eno16780033="null"

slaves_bond1="eno16780032 eno16780033"
mode_bond1="active-backup"
miimon_bond1="100"

config_bond1="null"
mtu_bond1="9000"

vlans_bond1="1111"
config_bond1_1111="10.0.0.2/24"
routes_bond1_1111="default via 10.0.0.1"
```

Рисунок 19. Пример конфигурации с интерфейсами eno16780032 и eno16780033

Добавить интерфейс bond1 в автозагрузку и активировать в системе (Рисунок 20).

```
# ln -s /etc/init.d/net.lo /etc/init.d/net.bond1
# rc-update add net.bond1
# /etc/init.d/net.bond1 start
```

Рисунок 20. Команда для добавления bond1 в автозагрузку и активации в системе

Проверить и настроить при отсутствии параметра InitiatorName имя инициатора ISCSI в файле /etc/iscsi/initiatorname.iscsi (Рисунок 21).

```
# cat /etc/iscsi/initiatorname.iscsi
InitiatorName=
# echo "InitiatorName=$(iscsi-iname)" > /etc/iscsi/initiatorname.iscsi
```

Рисунок 21. Команда для проверки и настройки InitiatorName

Запустить службу ISCSI (Рисунок 22).

```
# rc-update add iscsid
# /etc/init.d/iscsid start
```

Рисунок 22. Команда для запуска службы ISCSI

Настроить ISCSI с адресов сервера 192.168.1.10 и 192.168.1.11 на использование через bond1 и vlan 1111 (Рисунок 23).

```
# iscsiadm --mode discoverydb --type sendtargets --interface=bond1.1111 --portal 192.168.1.10 --discover
# iscsiadm --mode discoverydb --type sendtargets --interface=bond1.1111 --portal 192.168.1.11 --discover
# iscsiadm --mode node --targetname iqn.2001-05.com.doe:test --interface=bond1.1111 --portal 192.168.1.10:3260 --login
# iscsiadm --mode node --targetname iqn.2001-05.com.doe:test --interface=bond1.1111 --portal 192.168.1.11:3260 --login
# iscsiadm --mode node --targetname iqn.2001-05.com.doe:test --interface=bond1.1111 --portal 192.168.1.10:3260 -o
update -n node.startup -v automatic
# iscsiadm --mode node --targetname iqn.2001-05.com.doe:test --interface=bond1.1111 --portal 192.168.1.11:3260 -o
update -n node.startup -v automatic
```

Рисунок 23. Команды для настройки ISCSI