Высокая степень доступности (HA)

Высокая степень доступности (HA)

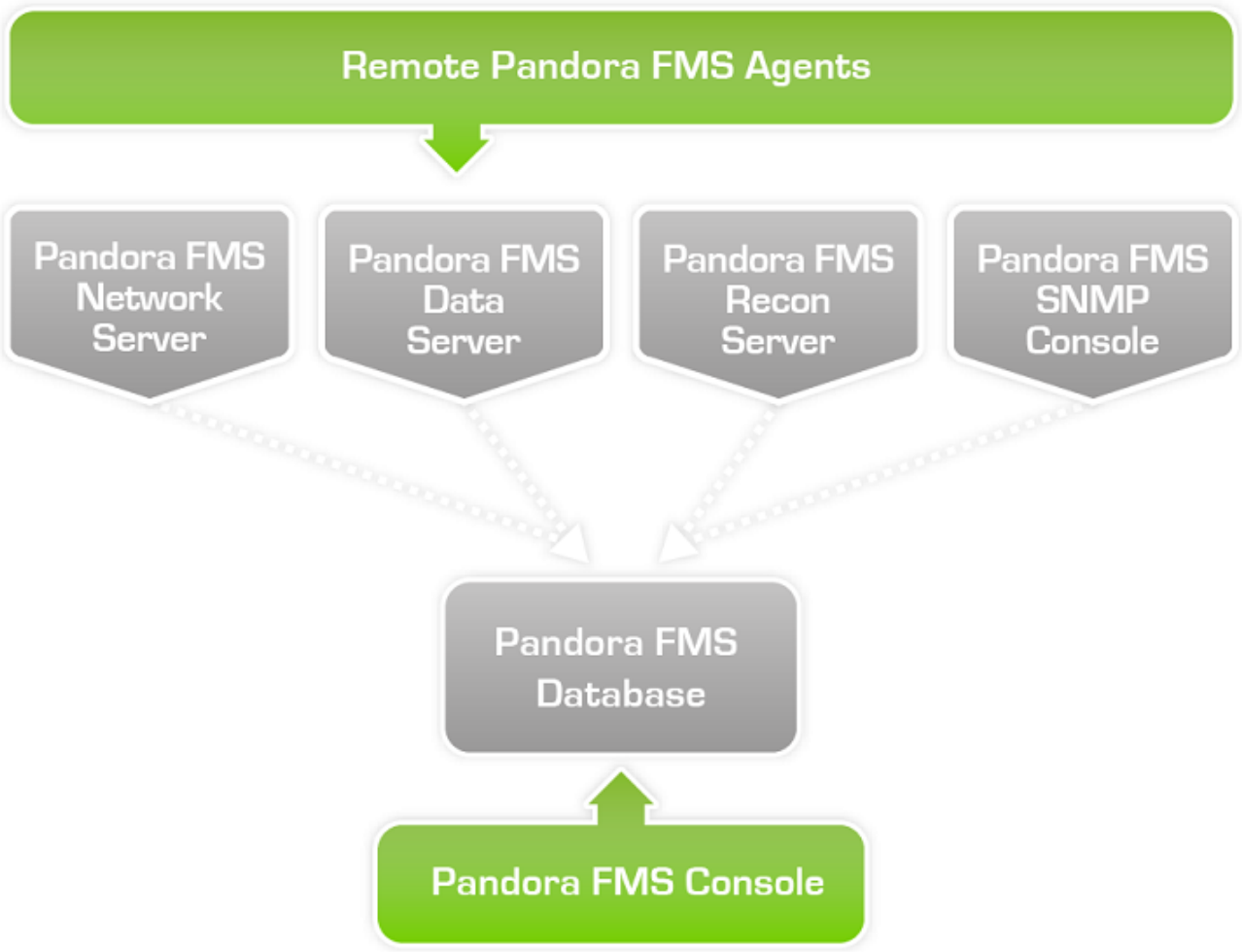
Вернуться в оглавление Документации Pandora FMS

Введение

Pandora FMS - очень стабильное приложение (благодаря тестам и улучшениям, вносимым в каждую версию, а также исправлению некоторых ошибок, обнаруженных пользователями), тем не менее, в критических средах и/или при большой нагрузке может возникнуть необходимость распределить нагрузку на несколько машин и быть уверенным, что при отказе любого компонента Pandora FMS система не рухнет.

Pandora FMS была разработана по модульному принципу, и любой из ее модулей может работать независимо. Но он также предназначен для совместной работы с другими компонентами и способен принять на себя нагрузку от тех компонентов, которые упали.

Стандартно Pandora FMS может выглядеть так, как показано на рисунке ниже.



Очевидно, что агенты не являются избыточными. Если агент выходит из строя, нет смысла запускать другого, поскольку единственной причиной выхода агента из строя является невозможность получения данных из-за сбоя в работе какого-либо модуля, который не может быть устранен другим агентом, работающим параллельно, или из-за отсутствия связи или сбоя системы. Очевидным решением является резервирование критических систем независимо от того, работают ли в них агенты Pandora FMS или нет, и, таким образом, резервирование мониторинга этих систем.

Использование HA может обсуждаться в различных сценариях:

- Балансировка и HA серверов данных.
- Балансировка и HA сетевых серверов, WMI, plugin, web, prediction, recon и подобные.
- Балансировка и HA в базах данных.
- Балансировка и HA консоли Pandora FMS.

Архитектурное проектирование HA и определение размеров

Наиболее важными компонентами Pandora FMS являются:

1. База данных

2. Сервер
3. Консоль

Каждый из компонентов может быть реплицирован для защиты системы мониторинга от любых проблем.

Чтобы определить количество узлов, необходимых для балансировки нагрузки, мы рассмотрим количество целей для мониторинга, а также объем, тип и частоту захвата метрик для сбора.

В зависимости от потребностей мониторинга, мы определим различные архитектуры.

Примечание: Испытания, проведенные для определения архитектур, были выполнены с использованием различного оборудования:

Intel(R) Core(TM) i5-8600K CPU @ 3.60GHz

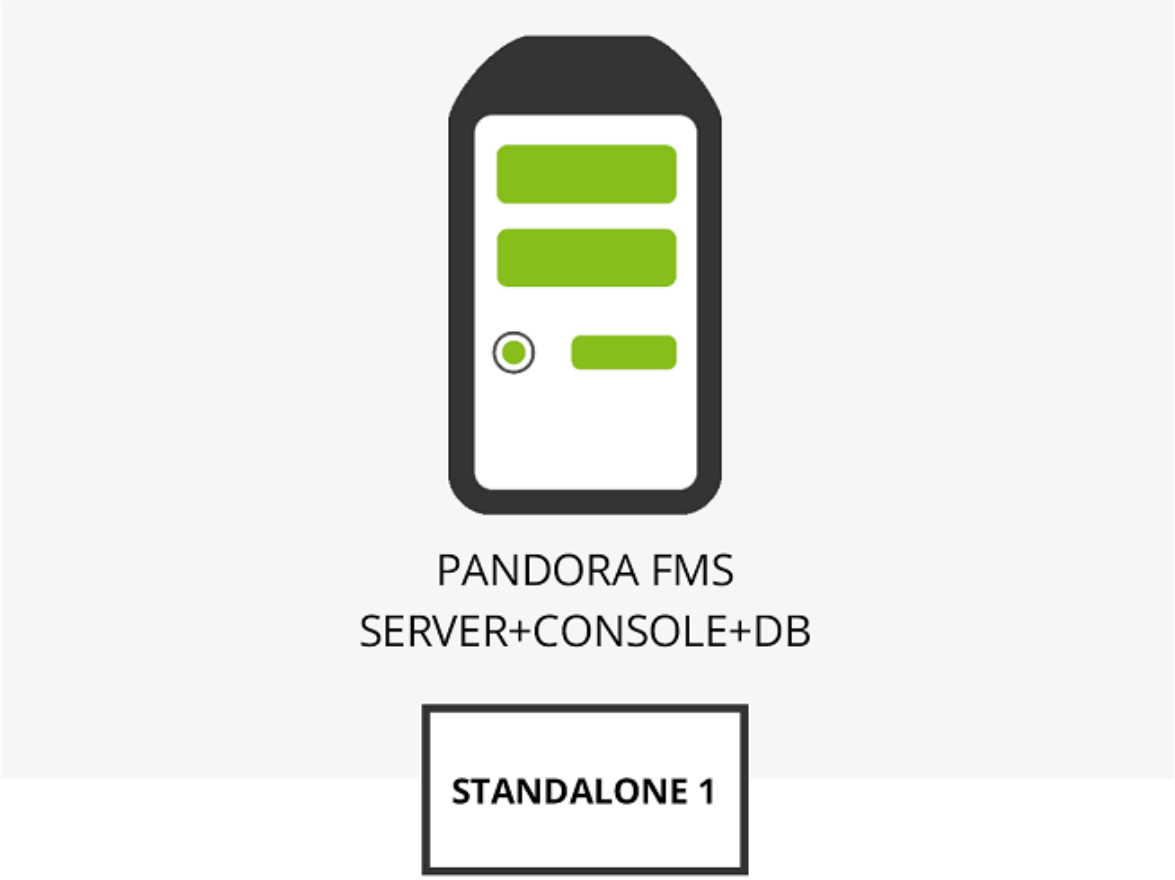
Экземпляр *t2.large* от Amazon [1)]

Определение размеров

В зависимости от потребностей:

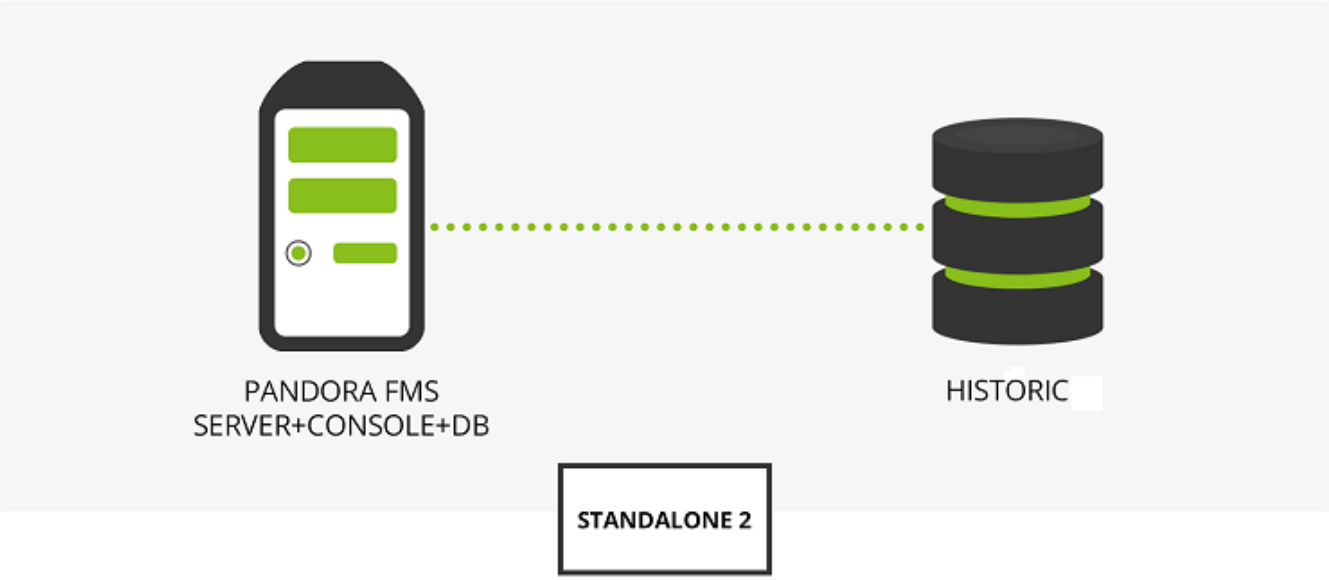
1. Автономно (без высокой доступности) до 2500 агентов / 50000 модулей каждые 5 минут, однородные данные, без исторических данных

```
Серверы: 1 (общий)

Главный:
 ----------
 CPU: 6 cores
 RAM: 8 ГБ
Диск: 100 ГБ
```



2. Автономно (без высокой доступности) до 2500 агентов / 50000 модулей каждые 5 минут, однородные данные, с историческими данными (1 год)

```
Серверы: 2 (1 общий, 1 история)

Главный:
 ----------
 CPU: 6 cores
 RAM: 8 ГБ
Диск: 100 ГБ

История:
 ----------
 CPU: 2 cores
 RAM: 4 ГБ
Диск: 200 ГБ
```



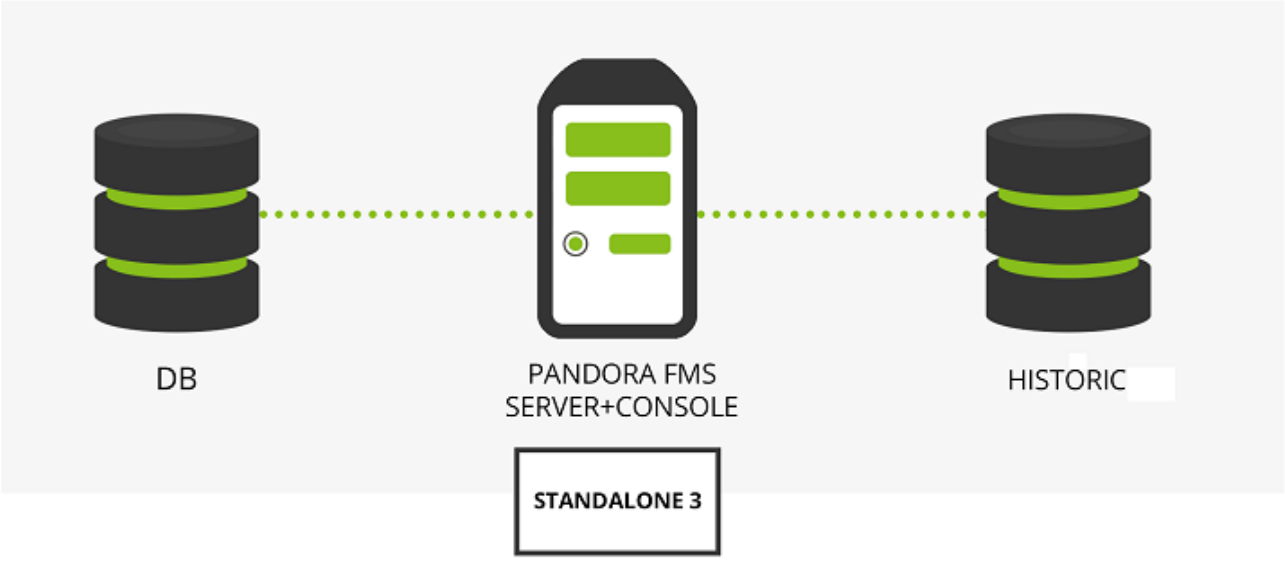
3. Автономно (без высокой доступности) до 5000 агентов / 100000 модулей каждые 5 минут, однородные данные, с историческими данными (1 год)

Серверы: 3 (1 сервер + консоль, 1 основная база данных, 1 история)

Сервер + консоль:
```
 --------------------
 CPU: 6 cores
 RAM: 8 ГБ
```
Диск: 40 ГБ

Главная база данных:
```
 ------------------------
 CPU: 4 cores
 RAM: 8 ГБ
```
Диск: 100 ГБ

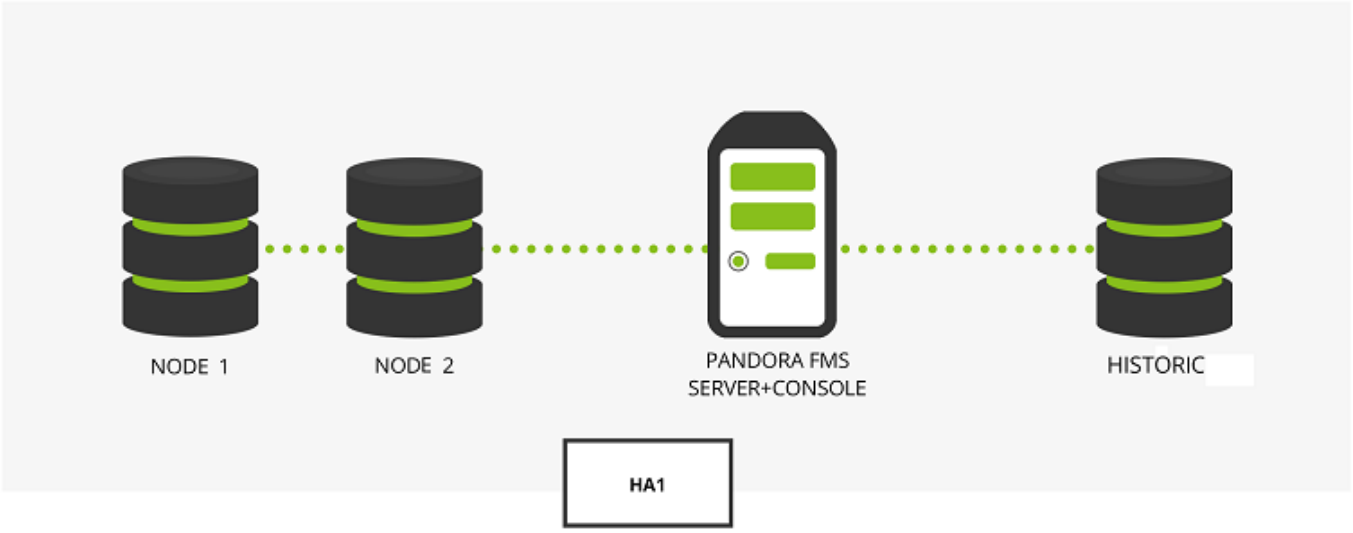
История:
```
 ----------
 CPU: 2 cores
 RAM: 4 ГБ
```
Диск: 200 ГБ



Создание архитектуры HA

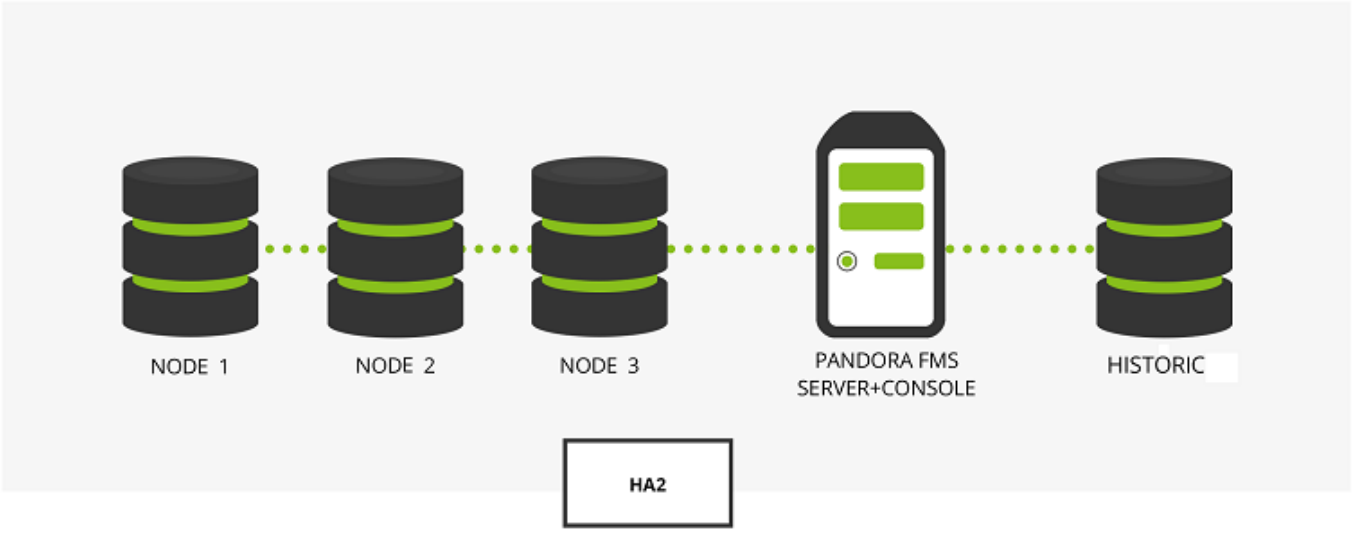
1. Простая база данных HA, до 7500 агентов / 125000 модулей каждые 5 минут, однородные данные, с историческими данными (1 год).

```
Серверы: 4 (1 сервер + консоль, 2 базы данных, 1 история)

Сервер + консоль:
 -------------------
 CPU: 6 cores
 RAM: 8 ГБ
Диск: 40 ГБ

База данных узел 1:
 ----------------------
 CPU: 6 cores
 RAM: 8 ГБ
Диск: 100 ГБ

База данных узел 2:
 ----------------------
 CPU: 6 cores
 RAM: 8 ГБ
Диск: 100 ГБ

История:
 ----------
 CPU: 2 cores
 RAM: 4 ГБ
Диск: 300 ГБ
```



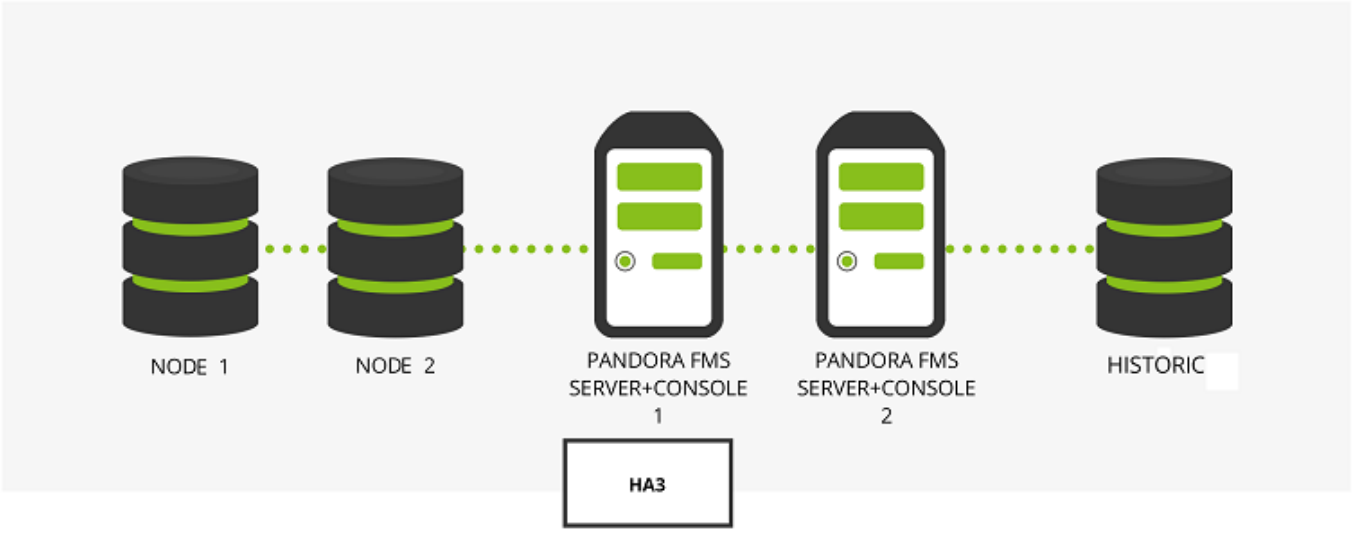
2. База данных с полной HA (с кворумом), до 7500 агентов / 125000 модулей каждые 5 минут, однородные данные, с историческими данными (1 год).

```
Серверы: 5 (1 сервер + консоль, 3 базы данных, 1 история)

Сервер + консоль:
 --------------------
 CPU: 6 cores
 RAM: 8 ГБ
Диск: 40 ГБ

База данных узел 1:
 ----------------------
 CPU: 6 cores
 RAM: 8 ГБ
Диск: 100 ГБ

База данных узел 2:
 ----------------------
 CPU: 6 cores
 RAM: 8 ГБ
Диск: 100 ГБ

База данных узел 3:
 ----------------------
 CPU: 6 cores
 RAM: 8 ГБ
Диск: 100 ГБ

История:
 ----------
 CPU: 2 cores
 RAM: 4 ГБ
Диск: 200 ГБ
```



3. База данных в простой HA и Pandora FMS в сбалансированной HA, до 7500 агентов / 125000 модулей каждые 5 минут, однородные данные, с историческими данными (1 год).

```
Серверы: 5 (2 сервера + консоль, 2 базы данных, 1 история)

Сервер + консоль:
 --------------------
 CPU: 6 cores
 RAM: 8 ГБ
Диск: 40 ГБ

Сервер + консоль:
 --------------------
 CPU: 6 cores
 RAM: 8 ГБ
Диск: 40 ГБ

База данных узел 1:
 ----------------------
 CPU: 6 cores
 RAM: 8 ГБ
Диск: 100 ГБ

База данных узел 2:
 ----------------------
 CPU: 6 cores
 RAM: 8 ГБ
Диск: 100 ГБ

История:
 ----------
 CPU: 2 cores
 RAM: 4 ГБ
Диск: 200 ГБ
```



4. Базовая HA сбалансирована на сервере, основной и базовой базе данных, до 4000 агентов / 90000 модулей каждые 5 минут, однородные данные, с историческими данными (1 год).

```
Серверы: 3 (2 общих, 1 история)

Главный: (консоль + сервер + узел базы данных 1).
 ----------
 CPU: 8 cores
 RAM: 12 ГБ
Диск: 100 ГБ

Вторичный: (консоль + сервер + узел базы данных 2).
 ----------
 CPU: 8 cores
 RAM: 12 ГБ
Диск: 100 ГБ

История:
 ----------
 CPU: 2 cores
 RAM: 4 ГБ
Диск: 200 ГБ
```

В этой схеме узлы базы данных Pandora FMS настроены на каждом из двух доступных серверов (основном и вторичном).

Примечание: Для больших сред мы определим каждую из ранее описанных схем конфигурации как *вычислительные узлы*.

Пример

Если вам нужно контролировать 30 000 агентов с 500 000 модулей, вы должны настроить столько узлов, сколько необходимо для выполнения этих требований. Следуя примеру:

Если мы выберем дизайн HA#1 (1 сервер+консоль, 2 узла базы данных в HA и база данных истории), нам придется настроить 30 000 / 7500 = 4 узла.

Для управления всей средой необходимо установить метаконсоль, откуда можно настроить всю инфраструктуру мониторинга.

Метаконсоль потребует:

```
Серверы: 1 (общий)

Главный:
 ----------
 CPU: 8 cores
 RAM: 12 ГБ
Диск: 100 ГБ
```

Общее количество серверов с независимыми историческими базами данных: 17

Общее количество серверов с объединенными историческими базами данных: 13

Примечание> Чтобы объединить все исторические базы данных (4) в одной машине, вам придется изменить размер их функций, чтобы машина смогла принять на себя

дополнительную нагрузку:

```
Объединенная история:
 --------------------
 CPU: 8 cores
 RAM: 12 ГБ
Диск: 1200 ГБ
```

Высокая доступность сервера данных

Самый простой способ - использовать HA, реализованный в агентах (который позволяет связаться с альтернативным сервером, если основной сервер не отвечает). Однако, поскольку сервер данных обслуживает через порт 41121 и это стандартный TCP-порт, можно использовать любое коммерческое решение, позволяющее балансировать или *кластеризовать* обычный TCP-сервис.

Для сервера данных Pandora FMS необходимо установить две машины с настроенным сервером данных Pandora FMS (и разными именами хоста и сервера). На каждой из них нужно будет установить сервер Tentacle. Каждая машина будет иметь свой IP-адрес. Если мы собираемся использовать внешний балансировщик, он предоставит один IP-адрес, к которому агенты будут подключаться для отправки своих данных.

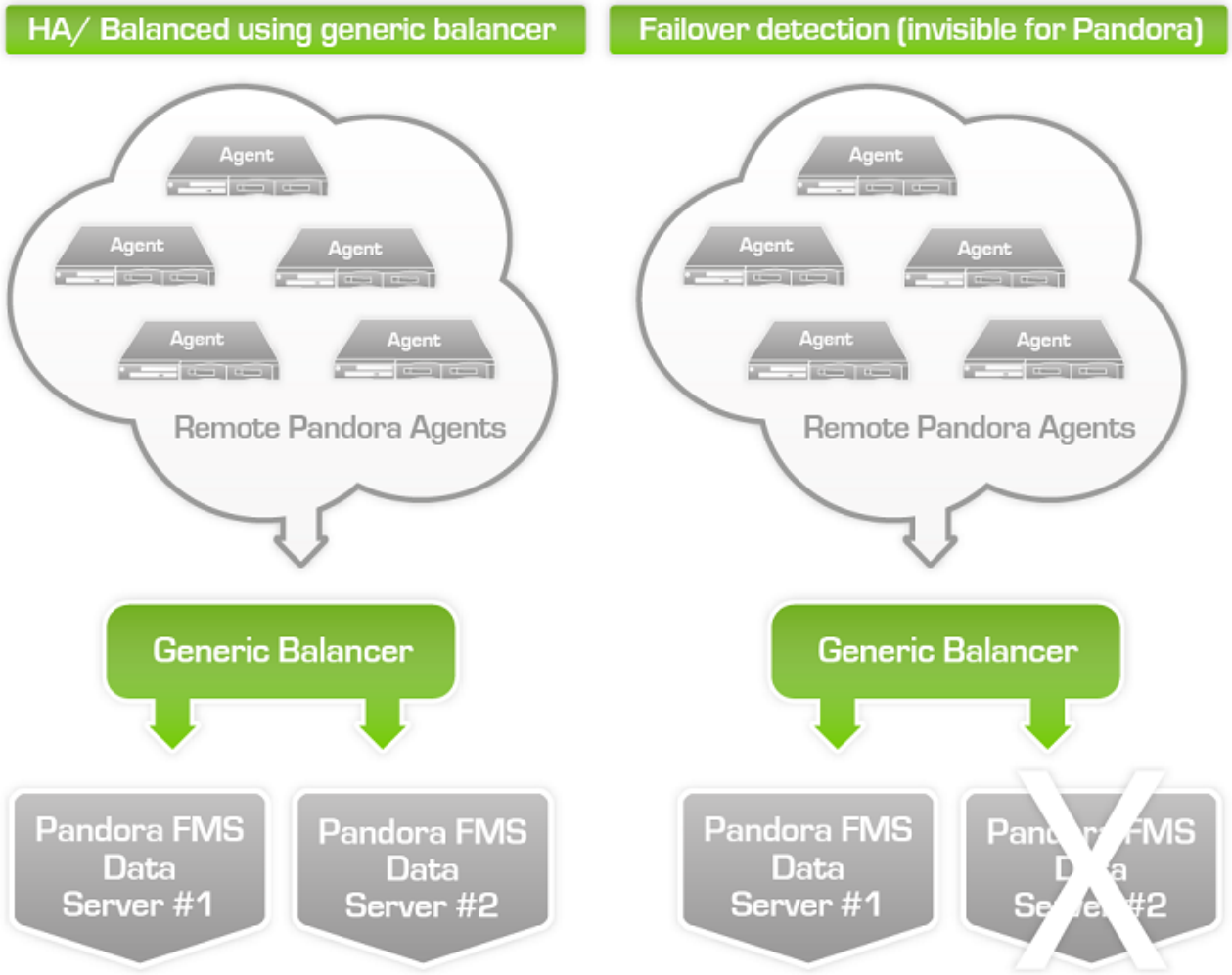
Если мы используем внешний балансировщик нагрузки, и один из серверов выходит из строя, механизм HA включает один из доступных активных серверов, и агенты Pandora FMS будут продолжать подключаться с тем же адресом, что и раньше, не замечая изменений, но в этом случае балансировщик нагрузки будет отправлять данные не на сервер, который вышел из строя, а на другой активный сервер. Нет необходимости менять что-либо в каждом сервере данных Pandora FMS, даже каждый сервер может сохранять свое собственное имя, полезное для того, чтобы знать, если какой-либо из них не работает, в при просмотре состояния сервера. Модули данных Pandora FMS могут обрабатываться любым сервером без необходимости предварительного распределения. Он разработан таким образом, что HA может быть реализован более удобным для пользователя способом..

В случае использования механизма HA агентов, будет наблюдаться небольшая задержка в отправке данных, так как при каждом выполнении агента, он будет пытаться соединиться с первичным сервером, и если тот не ответит, он сделает это со вторичным (если он был настроен таким образом). Это описано ниже как «Балансировка на программных агентах».

Если вы хотите использовать два сервера данных и оба должны обрабатывать удаленные политики, коллекции и конфигурации, вы должны совместно использовать ключевые каталоги чтобы все экземпляры сервера данных могли читать и записывать в эти каталоги. Консоли также должны иметь доступ к этим общим каталогам.

- /var/spool/pandora/data_in/conf

- /var/spool/pandora/data_in/collections
- /var/spool/pandora/data_in/md5
- /var/spool/pandora/data_in/netflow
- /var/www/html/pandora_console/attachment



Высокая доступность программных агентов

С помощью программных агентов можно выполнять балансировку серверов данных, поскольку можно настроить основной сервер данных и резервный сервер. В файле конфигурации агента *pandora_agent.conf* должна быть настроена и не комментированной следующая часть файла конфигурации агента:

```
# Secondary server configuration
# ==============================
# If secondary_mode is set to on_error, data files are copied to the secondary
# server only if the primary server fails. If set to always, data files are
# always copied to the secondary server
secondary_mode on_error
secondary_server_ip localhost
secondary_server_path /var/spool/pandora/data_in
```

```
secondary_server_port 41121
secondary_transfer_mode tentacle
secondary_server_pwd mypassword
secondary_server_ssl no
secondary_server_opts
```

Доступны следующие опции (для получения дополнительной информации см. главу о конфигурации агента).

- secondary_mode: Режим, в котором должен находиться вторичный сервер. Он может иметь два значения:
  - on_error: Он будет отправлять данные на вторичный сервер только в том случае, если не может отправить их на первичный сервер.
  - always: Он всегда отправляет данные на вторичный сервер, независимо от того, может ли он связаться с первичным сервером или нет.
- secondary_server_ip: IP-адрес вторичного сервера.
- secondary_server_path: Путь, по которому XML копируется на вторичный сервер, обычно /var/spool/pandora/data_in
- secondary_server_port: Порт, через который XML будет скопирован на вторичный сервер, в tentacle 41121, в ssh 22 и в ftp 21.
- secondary_transfer_mode: Режим передачи, который будет использоваться для копирования XML на вторичный сервер, Tentacle, ssh, ftp, ...
- secondary_server_pwd: Параметр пароля для передачи данных по FTP
- secondary_server_ssl: Установите значение yes или no в зависимости от того, хотите ли вы использовать ssl для передачи данных через Tentacle.
- secondary_server_opts: Другие параметры, необходимые для трансфера, будут помещены в это поле.

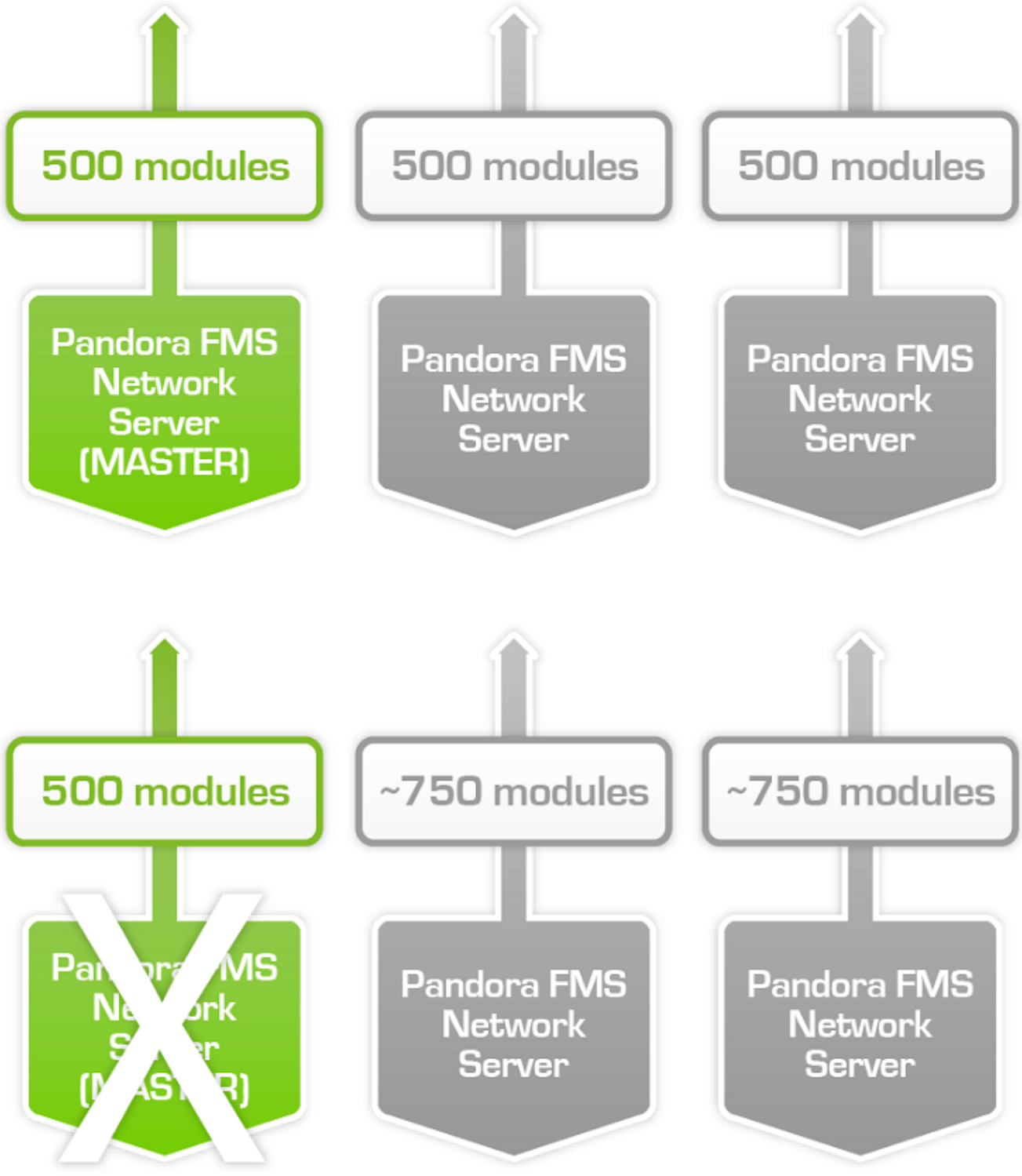
На главном сервере работает только удаленная конфигурация агента, если она включена.

Высокая доступность сетевых серверов, WMI, плагинов, web, prediction и т.п.

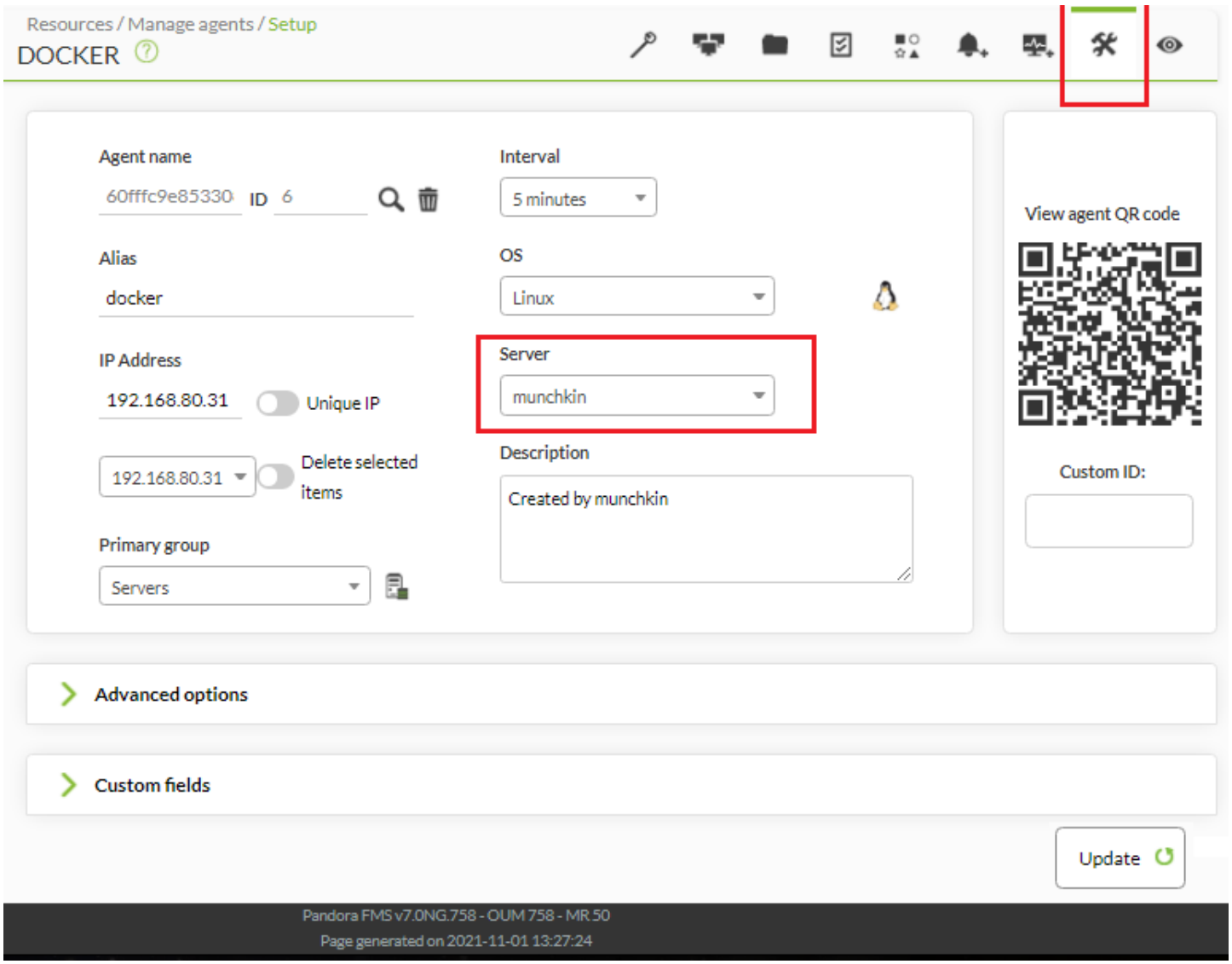
Вам необходимо установить несколько сетевых серверов. WMI, Plugin, Web или prediction, на нескольких машинах в сети (все с одинаковой видимостью в отношении систем, подлежащих мониторингу) и все в одном сегменте (чтобы данные о сетевой задержке были согласованными).

Серверы могут быть помечены как основные. Эти серверы будут автоматически собирать данные со всех модулей, назначенных на сервер, который помечен как «неработающий». В самих серверах Pandora FMS реализован механизм, позволяющий обнаружить, что один из них вышел из строя, путем проверки даты последнего контакта. (server threshold x 2). Достаточно, чтобы был активен только один сервер Pandora FMS, чтобы он смог обнаружить падение остальных. Если все серверы Pandora FMS не работают, нет возможности обнаружить или реализовать HA.

Очевидным способом реализации HA и балансировки нагрузки в двухузловой системе является назначение 50% модулей на каждый сервер и маркировка обоих серверов как ведущих. В случае более чем двух главных серверов и третьего упавшего сервера с большим количеством модулей, ожидающих выполнения, первый из главных серверов, выполняющий модуль, «самоназначается» модулем упавшего сервера. В случае восстановления одного из вышедших из строя серверов, модули, которые были назначены основному серверу, автоматически переназначаются.



Балансировка нагрузки между различными серверами осуществляется в административной части агента в меню «setup».



В поле «server» есть комбинированное поле, где вы выбираете сервер, который будет выполнять проверки.

Конфигурация на серверах

Сервер Pandora FMS может работать в двух различных режимах:

- Главный режим.
- Не главный режим.

Если сервер выходит из строя, его модули будут выполняться главным сервером, так что данные не будут потеряны.

В любой момент времени может быть только один главный сервер, который выбирается среди серверов, имеющих параметр конфигурации *master* в */etc/pandora/pandora_server.conf*

со значением больше 0:

```
master [1..7]
```

Если текущий главный сервер выходит из строя, выбирается новый главный сервер. Если кандидатов несколько, выбирается кандидат с наибольшим значением *master*.

Будьте осторожны при отключении серверов. Если сервер с сетевыми модулями выйдет из строя, а сетевой сервер главного сервера будет отключен, эти модули не будут работать.

Например, если у вас есть три сервера Pandora FMS в статусе *master*, установленным на 1, один главный сервер будет выбран случайным образом, а два других будут работать во второстепенном режиме. Если главный сервер выходит из строя, новый главный сервер выбирается случайным образом.

В pandora_server.conf были введены следующие параметры:

- ha_file: Адрес временного двоичного файла HA.
- ha_pid_file: Текущий процесс HA.
- pandora_service_cmd: Контроль состояния службы Pandora FMS.

Высокая доступность консоли Pandora FMS

Только нужно установить другую консоль указывающую на базу данных. Любая из консолей может использоваться одновременно из разных мест разными пользователями. Мы сможем использовать веб-балансировщик для консолей, если нам понадобится горизонтальный рост для балансировки нагрузки на консоли. Управление системой сессий осуществляется с помощью файлов cookie, которые хранятся в браузере.

В случае использования удаленной конфигурации и для управления ею со всех консолей, оба сервера данных и консоли должны совместно использовать каталог входных данных (по умолчанию: /var/spool/pandora/data_in) для удаленной конфигурации агентов, коллекций и других каталогов.

В этом руководстве показано, как предоставлять общий доступ к этим каталогам с помощью NFS и GlusterFS.

Важно предоставлять общий доступ только к подкаталогам внутри data_in, а не к самим

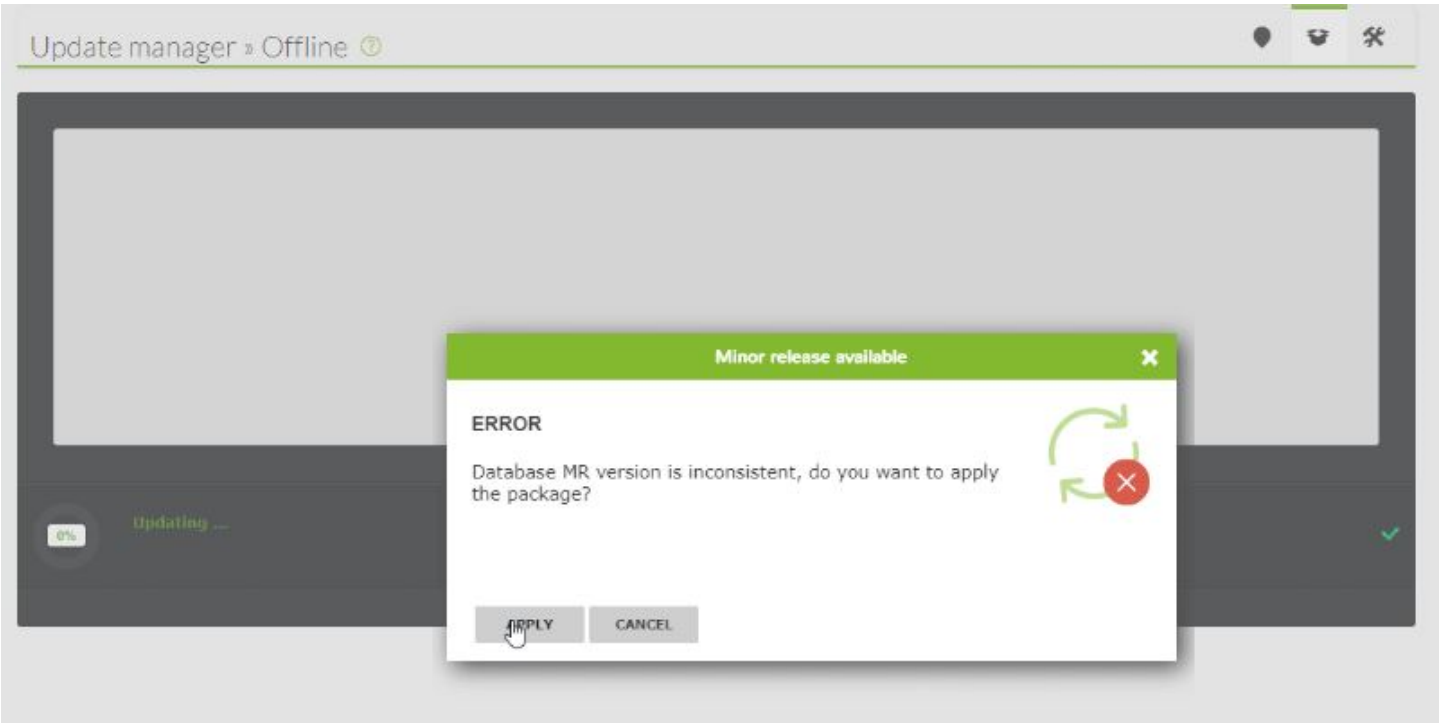
data_in, так как это негативно скажется на производительности сервера.

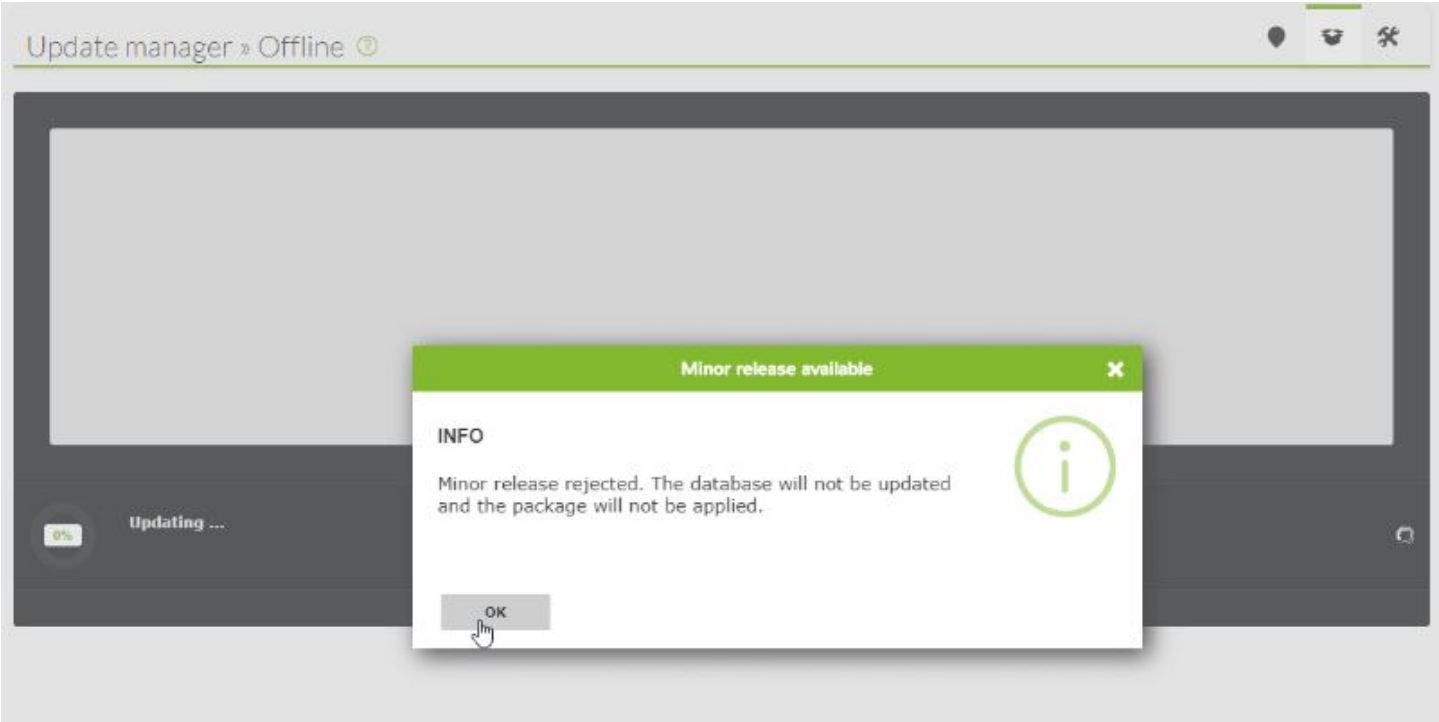
Обновление

При обновлении консоли Pandora FMS в среде высокой доступности важно учитывать следующие соображения при обновлении через OUM посредством Update Manager → Update Manager offline.

Пользователи версии Enterprise могут загрузить пакет OUM с сайта поддержки Pandora FMS.

В сбалансированной среде с общей базой данных при обновлении первой консоли к базе данных применяются соответствующие изменения. Это приводит к тому, что при обновлении второй консоли Pandora FMS выдает ошибку при поиске информации, уже вставленной в базу данных. Однако обновление консоли будет осуществляться тем же способом.





Высокая доступность в базе данных

Цель данного раздела - предложить комплексное решение для HA в среде Pandora FMS. Это единственная модель HA с официальной поддержкой Pandora FMS. Это решение поставляется - предварительно установленным - из OUM 724. Эта система заменяет DRBD и другие системы HA, рекомендованные в прошлом.

Это первое применение базы данных HA в Pandora FMS, и процесс установки почти полностью ручной, с использованием консоли GNU/Linux в качестве root. В будущих версиях мы облегчим настройку из графического интерфейса.

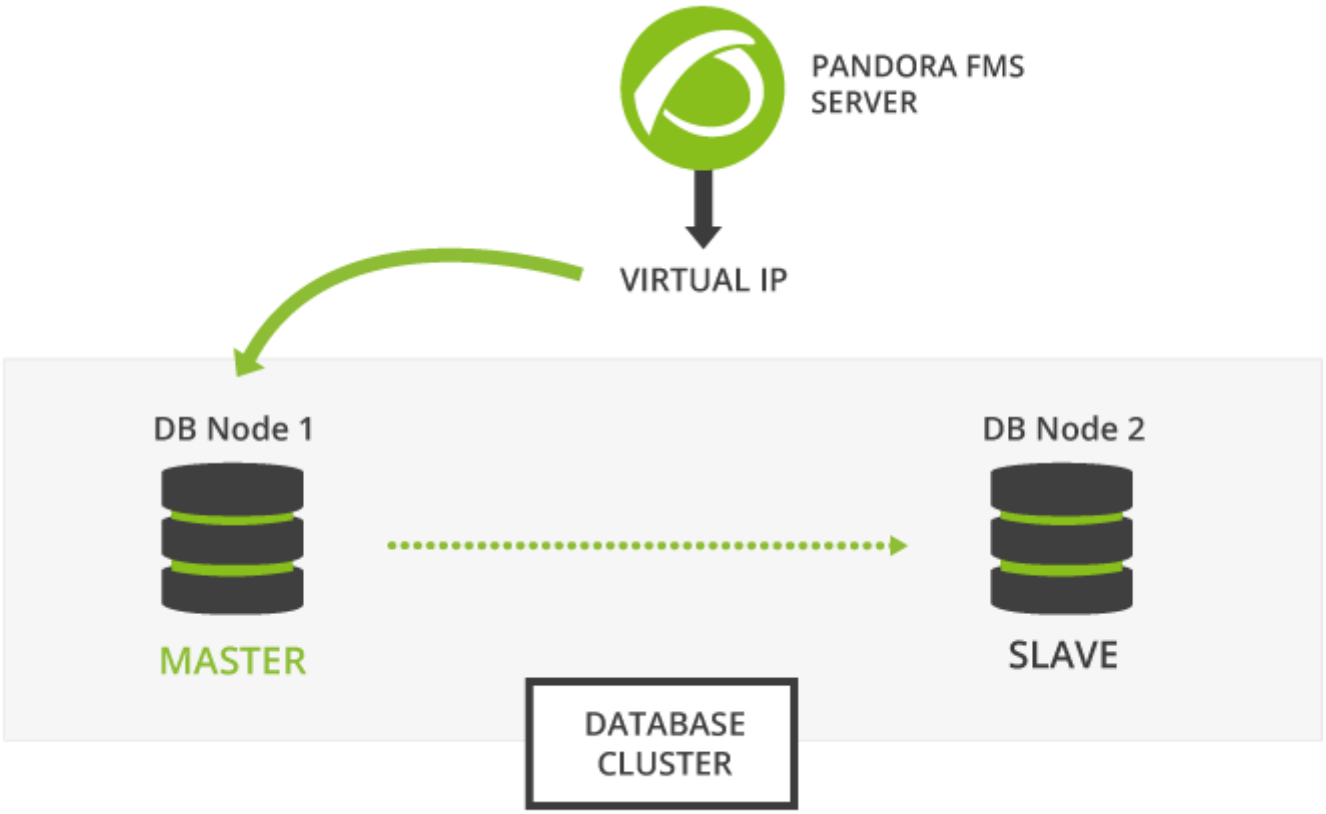
Pandora FMS основана на базе данных MySQL для конфигурирования и хранения данных. Сбой базы данных может на мгновение парализовать работу инструмента мониторинга. Кластер базы данных высокой доступности Pandora FMS позволяет нам легко развернуть надежную и отказоустойчивую архитектуру.

Это продвинутая функция, требующая знания систем GNU/Linux.

Управление ресурсами кластера осуществляется с помощью Pacemaker, усовершенствованного и масштабируемого менеджера ресурсов кластера высокой доступности. Corosync предоставляет модель связи закрытой группы процессов для

создания реплицированных машин состояний. Percona была выбрана в качестве РСУБД по умолчанию благодаря своей масштабируемости, доступности, безопасности и функциям резервного копирования.

Активная/пассивнаярепликация осуществляется с одного ведущего узла (с разрешением на запись) на любое количество ведомых узлов (только для чтения). Виртуальный IP-адрес всегда ведет к текущему главному узлу. Если ведущий узел выходит из строя, один из ведомых узлов становится ведущим, а виртуальный IP-адрес обновляется соответствующим образом.



Инструмент базы данных Pandora FMS HA, *pandora_ha*, осуществляет мониторинг кластера и следит за тем, чтобы сервер Pandora FMS непрерывно работал, перезапуская его при необходимости. *pandora_ha* в свою очередь контролируется systemd.

Рекомендуется хранить максимум 15 дней данных и событий, для более длительного хранения следует создать историческую базу данных. См. также тему «Управление и администрирование серверов PFMS».

Установка для RHEL 8 и Rocky Linux 8

Версия 760 или более поздняя

В этом примере будет настроен *cluster* из двух узлов, с *hosts*: node1 и node2.

Измените имена *hosts*, пароли и т.д. по мере необходимости в соответствии с развертываемой средой.

Команды, которые должны быть выполнены на любом узле, будут иметь следующий синтаксис (пример для node1):

```
node1#
<command>
<command>
<command>
```

Команды, которые должны быть выполнены на всех узлах, будут предваряться словом all:

```
all#
<command>
<command>
<command>
```

Существует также дополнительный *host* под названием pandorafms, на котором будет установлен Pandora FMS. Когда ссылаются на all, это относится только к узлам базы данных, дополнительный узел Pandora FMS всегда будет упоминаться как pandorafms и *не является частью* all.

Пререквизиты

RHEL версии 8 или Rocky Linux версии 8 должны быть установлены на всех *hosts*, и должны иметь возможность разрешать *hostnames* других *hosts*.

На каждом *host* должен быть установлен и запущен сервер OpenSSH. Подавление предупреждения, которое отображает OpenSSH:

```
all#
[ -f /etc/cron.hourly/motd_rebuild ] && rm -f
/etc/cron.hourly/motd_rebuild sed -i -e 's/^Banner.*//g' /etc/ssh/sshd_config
systemctl restart sshd
```

Инструмент базы данных Pandora FMS HA не будет работать должным образом, если в OpenSSH настроено предупреждение.

Сгенерируйте новые ключи аутентификации SSH для каждого *host* и скопируйте открытый

ключ для каждого из *hosts*.

Вы можете сгенерировать ключи для пользователя, отличного от *root*, для последующей установки кластера с пользователем «*non-root*».

```
all#
printf "\n\n\n" | ssh-keygen -t rsa -P ''
ssh-copy-id -p22 root@node1
ssh-copy-id -p22 root@node2
```

```
pandorafms#
printf "\n\n\n" | ssh-keygen -t rsa -P ''
ssh-copy-id -p22 root@node1
ssh-copy-id -p22 root@node2
```

На узле Pandora FMS скопируйте пару ключей в следующие каталоги httpd , ssh. Консоль Pandora FMS (httpd) должна получить статус *cluster*:

```
pandorafms#
cp -r /root/.ssh/ /usr/share/httpd/
chown -R apache:apache /usr/share/httpd/.ssh/
```

Следующие шаги необходимы только в том случае, если узлы используют SSH на нестандартном порту.

Вы должны заменить 22 на номер порта, который вы используете:

```
all#
echo -e "Host node1\n Port 22">> /root/.ssh/config
echo -e "Host node2\n Port 22">> /root/.ssh/config
```

Вы должны протестировать аутентификацию без пароля от каждого узла к другим:

```
all#
ssh node1
ssh node2
```

```
pandorafms#
ssh node1
ssh node2
```

Установка Percona

Установите необходимый пакет:

```
all#
dnf install dnf-utils \
 https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm \
 https://repo.percona.com/yum/percona-release-latest.noarch.rpm"

dnf module disable -y mysql

dnf install -y Percona-Server-server-57 percona-xtrabackup-24
```

Для получения дополнительной информации о процессе установки Percona вы можете обратиться к официальной документации по продукту:

https://www.percona.com/doc/percona-server/5.7/installation/yum_repo.html

После установки пакетов убедитесь, что служба Percona отключена, так как ею будет управлять *cluster*:

```
all#
systemctl disable mysqld
```

Если системная служба не отключена, диспетчер
ресурсов *cluster* не будет работать правильно.

Затем запустите сервер Percona:

```
all#
systemctl start mysqld
```

Будет сгенерирован новый временный пароль для подключения к `/var/log/mysqld.log`. Подключитесь к серверу Percona и измените пароль для *root*:

```
all#
export MYSQL_PWD=$(grep "temporary password" /var/log/mysqld.log | rev | cut -d'
' -f1 | rev)

echo """
  SET PASSWORD FOR 'root'@'localhost' = PASSWORD('Pandor4!');
  UNINSTALL PLUGIN validate_password;
  SET PASSWORD FOR 'root'@'localhost' = PASSWORD('pandora');
```

```
  """ | mysql --connect-expired-password -uroot
```

При выполнении настройки MySQL это можно сделать с помощью следующего автогенератора, который уже включен в установочный пакет сервера Pandora FMS Enterprise по умолчанию.

После установки *server*, вы найдете построитель конфигурации для репликации базы данных по пути:

```
/usr/share/pandora_server/util/myconfig_ha_gen.sh
```

```
Example: ./myconfig_ha_gen.sh -i serverid [-l file_location] [-d database] [-b
binlog] [-u dbuser] [-p dbpass] [-s poolsize] [-h help]
Mandatory parameters:
    -i --serverid Set the server id for the database (Mandatory)
Optional parameters:
    -l --location Set my.cnf custom location including filename. [ default
value: /etc/my.cnf ] (optional)
    -d --database Set the database to be replicated. [ default value: pandora ]
(optional)
    -b --binlog Set binlog file. [ default value: mysql-bin ] (optional)
    -u --dbuser Set dbuser for mysql connection and backups. [ default value:
root ] (optional)
    -p --dbpass Set dbpassword for mysql connection and backups. [ default
value: pandora ] (optional)
    -s --poolsize Set innodb_buffer_pool_size static size in M (Megabytes) or G
(Gigabytes). [ default value: autocalculated ] (optional)
    -h --help Print help.
```

В текущем случае, когда базы данных не находятся на том же сервере, что и приложение, необходимо будет скопировать *script* в стандартных средах некоторые необязательные параметры для пользовательских сред.

```
pandorafms#
scp /usr/share/pandora_server/util/myconfig_ha_gen.sh root@node1:/root/
scp /usr/share/pandora_server/util/myconfig_ha_gen.sh root@node2:/root/
```

Необходимо будет только передать параметр serverid (обязательный) в стандартных средах и некоторые необязательные параметры для пользовательских сред.

Если заданный по умолчанию или заданный пользователь не подключается к базе данных, сценарий завершится с ошибкой подключения.

У вас также есть возможность передать имя базы данных, пользователя и пароль в качестве аргументов. В противном случае будут использоваться настройки по умолчанию.

В этом случае он выполнит сценарий на обоих узлах, передав `server id` только в том

случае, если он имеет учетные данные по умолчанию, в противном случае он должен определить необходимые параметры.

```
node1#
/root/myconfig_ha_gen.sh -i 1
```

```
node2#
/root/myconfig_ha_gen.sh -i 2
```

Каждый узел должен иметь уникальный идентификатор.

Файл конфигурации Percona будет записан в `/etc/my.cnf` , где будут определены идентификатор сервера и рекомендуемая конфигурация для репликации базы данных. Вы должны перезапустить службу mysqld, чтобы убедиться, что конфигурация применена правильно.

```
all#
systemctl restart mysqld
```

Установка Pandora FMS

Вы можете либо выполнить полностью новую установку, либо перенести имеющиеся данные из существующего экземпляра.

Новая установка Pandora FMS

Установите Pandora FMS на вновь созданную базу данных. Остановите сервер Pandora FMS:

```
pandorafms#
/etc/init.d/pandora_server stop
```

Начиная с версии NG 754 у вас появились дополнительные возможности по ручному запуску и остановке сред высокой доступности (HA).

Существующая установка Pandora FMS

Остановите сервер Pandora FMS:

```
pandorafms#
```

```
/etc/init.d/pandora_server stop
```

Создайте резервную копию базы данных Pandora FMS и перенесите ее на узел 1:

```
pandorafms# mysqldump -uroot -ppandora --databases pandora> /tmp/pandoradb.sql
scp /tmp/pandoradb.sql node1:/tmp/
```

Теперь загрузите информацию в новую базу данных на узле (если вы не используете учетные данные и имя базы данных по умолчанию, измените их в следующей команде):

```
node1# mysql -uroot -ppandora pandora -e source "/tmp/pandoradb.sql"
```

Конфигурация репликации

Предоставьте необходимые привилегии для репликации, чтобы она работала на всех базах данных:

```
all#
mysql -uroot -ppandora
  GRANT ALL ON pandora.* TO 'root'@'%' IDENTIFIED BY 'pandora';
  GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'root'@'%' IDENTIFIED BY
'pandora';
  GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.* TO 'pandora'@'%' IDENTIFIED
BY 'pandora';
  GRANT REPLICATION CLIENT, REPLICATION SLAVE, SUPER, PROCESS, RELOAD ON *.* TO
'root'@'localhost' IDENTIFIED BY 'pandora';
  GRANT select ON mysql.user TO 'root'@'%' IDENTIFIED BY 'pandora'; FLUSH
PRIVILEGES; quit;
```

Остановите базу данных node2:

```
node2#
systemctl stop mysqld
```

Сделайте резервную копию базы данных первого узла (node1) и запишите имя и позицию главного *log* файла (в данном примере `mysql-bin.000001` и 785):

```
node1#
[ -e /root/pandoradb.bak ] && rm -rf /root/pandoradb.bak
innobackupex --no-timestamp /root/pandoradb.bak/
innobackupex --apply-log /root/pandoradb.bak/
rsync -avpP -e ssh /root/pandoradb.bak/ node2:/var/lib/mysql/
cat /root/pandoradb.bak/xtrabackup_binlog_info
```

Загрузите базу данных на второй узел (node2) и настройте ее на репликацию с первого узла (необходимо установить `MASTER_LOG_FILE` и `MASTER_LOG_POS` на значения из предыдущего шага):

```
node2#
chown -R mysql:mysql /var/lib/mysql
chcon -R system_u:object_r:mysqld_db_t:s0 /var/lib/mysql

systemctl start mysqld

mysql -uroot -ppandora
  CHANGE MASTER TO MASTER_HOST='node1',
  MASTER_USER='root', MASTER_PASSWORD='pandora',
  MASTER_LOG_FILE ='mysql-bin.000001', MASTER_LOG_POS =785;
  START SLAVE;
  SHOW SLAVE STATUS \G
```

Вы получите результат, похожий на:

```
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: node1
                  Master_User: root
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000002
          Read_Master_Log_Pos: 785
               Relay_Log_File: node2-relay-bin.000003
                Relay_Log_Pos: 998
        Relay_Master_Log_File: mysql-bin.000002
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB: pandora
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
                 Skip_Counter: 0
          Exec_Master_Log_Pos: 785
              Relay_Log_Space: 1252
              Until_Condition: None
               Until_Log_File:
                Until_Log_Pos: 0
           Master_SSL_Allowed: No
           Master_SSL_CA_File:
           Master_SSL_CA_Path:
              Master_SSL_Cert:
            Master_SSL_Cipher:
               Master_SSL_Key:
        Seconds_Behind_Master: 0
Master_SSL_Verify_Server_Cert: No
                Last_IO_Errno: 0
                Last_IO_Error:
               Last_SQL_Errno: 0
               Last_SQL_Error:
  Replicate_Ignore_Server_Ids:
             Master_Server_Id: 1
                  Master_UUID: 580d8bb0-6991-11e8-9a22-16efadb2f150
             Master_Info_File: mysql.slave_master_info
                    SQL_Delay: 0
          SQL_Remaining_Delay: NULL
      Slave_SQL_Running_State: Slave has read all relay log; waiting
for more updates
           Master_Retry_Count: 86400
                  Master_Bind:
      Last_IO_Error_Timestamp:
     Last_SQL_Error_Timestamp:
               Master_SSL_Crl:
           Master_SSL_Crlpath:
           Retrieved_Gtid_Set:
            Executed_Gtid_Set:
                Auto_Position: 0
         Replicate_Rewrite_DB:
                 Channel_Name:
           Master_TLS_Version:
1 row in set (0.00 sec)
```

Вы должны убедиться, что `Slave_IO_Running` и `Slave_SQL_Running` показывают Yes. *Другие значения могут отличаться от приведенных в примере.*

Если все в порядке, выйдите из интерфейса базы данных:

```
#node2
mysql> QUIT
```

Конфигурация кластера с двумя узлами

Установите необходимые пакеты: Для Rocky Linux™ достаточно выполнить следующую команду.

```
all#
dnf install -y --enablerepo='ha' chrony epel-release corosync pacemaker pcs
```

В случае RedHat перед установкой необходимо включить репозиторий rhel-8-for-x86_64-highavailability-rpms из *subscription manager*.

```
all#
subscription-manager repos --enable=rhel-8-for-x86_64-highavailability-rpms
dnf install -y --enablerepo='rhel-8-for-x86_64-highavailability-rpms' chrony
epel-release corosync pacemaker pcs
```

Теперь определите конфигурационный файл и включите службы corosync, pscd и chrony (замена старого ntpd).

```
all#
cp /etc/corosync/corosync.conf.example /etc/corosync/corosync.conf

systemctl enable chronyd --now

systemctl enable pcsd --now

systemctl enable corosync --now
```

При попытке запустить службу corosync: вы можете увидеть сообщение об ошибке: это связано с тем, что она еще не настроена, проигнорируйте его и продолжите следующие шаги.

Остановите сервер Percona:

```
all#
systemctl stop mysqld
```

Аутентификация всех узлов в кластере

Определите пароль пользователя `hacluster`:

```
all#
echo hapass | passwd hacluster --stdin
```

Создайте и запустите *cluster*, эти шаги будут необходимы только для того, чтобы сделать это в node1:

```
node1#
pcs host auth node1 node2 -u hacluster -p hapass

pcs cluster setup pandorafms node1 node2 --force

pcs cluster start --all
pcs cluster enable --all

pcs property set stonith-enabled=false

pcs property set no-quorum-policy=ignore
```

Проверьте состояние *cluster*:

```
node1#
pcs status
```

Вы увидите результат, аналогичный следующему:

```
   Cluster name: pandoraha
   Stack: corosync
   Current DC: node1 (version 1.1.18-11.el7_5.2-2b07d5c5a9) - partition
with quorum
   Last updated: Fri Jun  8 12:53:49 2018
   Last change: Fri Jun  8 12:53:47 2018 by root via cibadmin on node1

   2 nodes configured
   0 resources configured

   Online: [ node1 node2 ]

   No resources

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

Оба узла должны быть в режиме онлайн

(`Online: [ node1 node2 ]`).

Другие значения могут отличаться от приведенных в примере.

Установка агента репликации Pacemaker от Percona

Кардиостимулятор можно загрузить вручную из библиотеки PFMS.

Если у вас есть доступ в Интернет, вы можете установить его, запустив:

```
all#
cd /usr/lib/ocf/resource.d/
mkdir percona
cd percona
curl -L -o pacemaker_mysql_replication.zip
https://pandorafms.com/library/wp-content/uploads/2019/12/pacemaker_mysq
l_replication.zip
unzip pacemaker_mysql_replication.zip
rm -f pacemaker_mysql_replication.zip
chmod u+x mysql
cd
```

Настройте ресурсы *cluster*.

Если пароль по умолчанию, используемый в данном руководстве для пользователя базы данных *root*, был изменен, рекомендуется обновить `replication_passwd` и `test_passwd` соответственно.
Имена ресурсов *cluster* должны быть точно такими, как указано в

данном руководстве (pandoraip и pandoradb).

Замените <VIRT_IP> на предпочитаемый виртуальный IP-адрес:

```
#node1
export VIP='<VIRT_IP>'
pcs resource create pandoradb ocf:percona:mysql config="/etc/my.cnf" \
pid="/var/run/mysqld/mysqld.pid" socket="/var/lib/mysql/mysql.sock" \
replication_user="root" replication_passwd="pandora" max_slave_lag="60" \
evict_outdated_slaves="false" binary="/usr/sbin/mysqld" datadir="/var/lib/mysql"
\
test_user="root" test_passwd="pandora" op start interval="0" timeout="60s" \
op stop interval="0" timeout="60s" op promote timeout="120" op demote
timeout="120" \
op monitor role="Master" timeout="30" interval="5" on-fail="restart" op monitor
role="Slave" \
timeout="30" interval="10"

pcs resource create pandoraip ocf:heartbeat:IPaddr2 ip=$VIP cidr_netmask=24 \
op monitor interval=20s

pcs resource promotable pandoradb meta master-max="1" master-node-max="1" clone-
max="2" clone-node-max="1" notify="true" \
globally-unique=" false" target-role="Master" is-managed="true"

pcs constraint colocation add master pandoradb-clone with pandoraip

pcs constraint order promote pandoradb-clone then start pandoraip

sleep 5 ; pcs resource cleanup
```

Проверка состояния *cluster*:

```
node1#
pcs status
```

Вы увидите результат, аналогичный следующему:

```
Cluster name: pandorafms
Cluster Summary:
  * Stack: corosync
  * Current DC: node1 (version 2.1.0-8.el8-7c3f660707) - partition with
quorum
  * Last updated: Wed Feb  2 16:15:07 2022
  * Last change:  Mon Jan 31 15:49:07 2022 by root via crm_attribute on
node1
  * 2 nodes configured
  * 3 resource instances configured

Node List:
  * Online: [ node1 node2 ]

Full List of Resources:
  * pandoraip   (ocf::heartbeat:IPaddr2):        Started node1
  * Clone Set: pandoradb-clone [pandoradb] (promotable):
    * Masters: [ node1 ]
    * Slaves: [ node2 ]

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

Оба узла должны быть в режиме онлайн

(`Online: [ node1 node2 ]`).

Другие значения могут отличаться от приведенных в примере.

Конфигурация двухузлового кластера с пользователем "не root"

Он будет проводиться аналогично предыдущему.

Учетные данные пользователя должны быть скопированы, как уже объяснялось ранее, и необходимо выполнить следующие действия:

```
# All nodes:
useradd <usuario>
passwd <usuario>
usermod -a -G haclient <usuario>

# Enable PCS ACL system
pcs property set enable-acl=true --force

# Create role
pcs acl role create <rol> description="RW role" write xpath /cib

# Create PCS user - Local user
pcs acl user create <usuario> <rol>
```

```
# Login into PCS from ALL nodes
su - <usuario>
pcs status
Username: <usuario>
Password: *****

# Wait for 'Authorized' message, ignore output. Wait a second and retry 'pcs
status' command
```

Установка для RedHat 7 и CentOS 7

Версия 759 или более ранняя

Мы создадим двухузловой кластер с хостами *node1* и *node2*. Мы меняем имена хостов, пароли и т.д. в зависимости от того, что нам нужно для соответствия нашей среде.

Командам, которые должны быть выполнены на узле, предшествует имя хоста этого узла. Например:

```
node1# <command>
```

Перед командами, которые должны быть выполнены на всех узлах, должно стоять слово all. Например:

```
all# <command>
```

Существует дополнительный хост, называемый pandorafms, на котором находится или установлен Pandora FMS.

Когда all относится только к узлам Базы данных,
дополнительный узел Pandora FMS всегда будет
упоминаться как pandorafms и не является частью all.

Предварительные требования

CentOS версии 7 должна быть установлена на всех хостах и должна иметь возможность разрешать имена других хостов.

```
node1# ping node2
PING node2 (192.168.0.2) 56(84) bytes of data.

node2# ping node1
PING node1 (192.168.0.1) 56(84) bytes of data.
```

```
pandorafms# ping node1
PING node1 (192.168.0.1) 56(84) bytes of data.

pandorafms# ping node2
PING node2 (192.168.0.2) 56(84) bytes of data.
```

На каждом хосте должен быть установлен и запущен сервер Open SSH. Уберите баннер с надписью Open SSH:

```
all# [ -f /etc/cron.hourly/motd_rebuild ] && rm -f /etc/cron.hourly/motd_rebuild
all# sed -i -e 's/^Banner.*//g' /etc/ssh/sshd_config
all# systemctl restart sshd
```

Инструмент базы данных Pandora FMS HA не будет работать правильно, если баннер настроен на Open SSH.

Мы генерируем новые ключи аутентификации SSH для каждого узла и копируем открытый ключ для каждого узла:

Ключи могут быть созданы для пользователя NO root, для последующей установки кластера с пользователем «no root».

```
node1# echo -e "\n\n\n" | ssh-keygen -t rsa
node1# ssh-copy-id -p22 root@node2
node1# ssh node2

node2# echo -e "\n\n\n" | ssh-keygen -t rsa
node2# ssh-copy-id -p22 root@node1
node2# ssh node1

pandorafms# echo -e "\n\n\n" | ssh-keygen -t rsa
pandorafms# ssh-copy-id -p22 root@node1
pandorafms# ssh-copy-id -p22 root@node2
pandorafms# ssh node1
pandorafms# ssh node2
```

На узле Pandora FMS скопируйте пару ключей в */usr/share/httpd/.ssh* /. Консоль Pandora FMS должна восстановить состояние кластера:

```
pandorafms# cp -r /root/.ssh/ /usr/share/httpd/
pandorafms# chown -R apache:apache /usr/share/httpd/.ssh/
```

Следующие шаги необходимы только в том случае, если узлы используют SSH на нестандартном порту. Заменим *22* на правильный номер порта:

```
all# echo -e "Host node1\n    Port 22">> /root/.ssh/config
all# echo -e "Host node2\n    Port 22">> /root/.ssh/config
```

Установка Percona

Установите необходимый пакет:

```
all# yum install https://repo.percona.com/yum/percona-release-latest.noarch.rpm

all# yum install -y Percona-Server-server-57 percona-xtrabackup-24
```

Для получения дополнительной информации о процессе установки Percona, пожалуйста, обратитесь к официальной документации продукта:
https://www.percona.com/doc/percona-server/5.7/installation/yum_repo.html

После установки пакетов мы убеждаемся, что служба Percona деактивирована, поскольку она управляется кластером:

```
all# systemctl disable mysqld
```

Если системная служба не отключена, менеджер
ресурсов кластера будет работать неправильно.

Далее мы запускаем сервер Percona:

```
all# systemctl start mysqld
```

Будет сгенерирован новый временный пароль, подключенный к /var/log/mysqld.log. Мы подключаемся к серверу Percona и меняем пароль root:

```
all# mysql -uroot -p$(grep "temporary password" /var/log/mysqld.log | \
rev | cut -d' ' -f1 | rev)
mysql> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('Pandor4!');
mysql> UNINSTALL PLUGIN validate_password;
mysql> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('pandora');
mysql> quit
```

Переустановите с помощью опции ha.

```
pandorafms# ./pandora_server_installer --install --ha
```

После установки сервера с включенными инструментами HA вы найдете генератор конфигурации для репликации базы данных по пути: */usr/share/pandora_server/util/myconfig_ha_gen.sh*.

```
Пример: ./myconfig_ha_gen.sh -i serverid [-l file_location] [-d database] [-b
binlog] [-u dbuser] [-p dbpass] [-s poolsize] [-h help]
Mandatory parameters:
       -i --serverid   Set the server id for the database (Mandatory)
Optional parameters:
       -l --location   Set my.cnf custom location including filename. [ default
value: /etc/my.cnf ] (optional)
       -d --database   Set the database to be replicated. [ default value:
pandora ] (optional)
       -b --binlog     Set binlog file. [ default value: mysql-bin ] (optional)
       -u --dbuser     Set dbuser for mysql connection and backups. [ default
value: root ] (optional)
       -p --dbpass     Set dbpassword for mysql connection and backups. [
default value: pandora ] (optional)
       -s --poolsize   Set innodb_buffer_pool_size static size in M (Megabytes)
or G (Gigabytes). [ default value: autocalculated ] (optional)
       -h --help       Print help.
```

В текущем случае, когда базы данных не находятся на том же сервере, что и приложение, необходимо скопировать сценарий на узлы для локального выполнения.

```
pandorafms# scp /usr/share/pandora_server/util/myconfig_ha_gen.sh
root@node1:/root/
pandorafms# scp /usr/share/pandora_server/util/myconfig_ha_gen.sh
root@node2:/root/
```

Как мы видим в примере, необходимо передать только параметр serverid (обязательный) в стандартных средах и некоторые необязательные параметры для пользовательских сред.

Если пользователь по умолчанию или определенный пользователь не подключится к базе данных, сценарий завершится с ошибкой подключения.

Также можно передать базу данных, пользователя и пароль в качестве аргументов. В противном случае будут использоваться настройки по умолчанию.

Для этого случая мы запустим скрипт на обоих узлах, передав только id сервера, если у нас есть учетные данные по умолчанию, в противном случае определим необходимые параметры.

```
 node1# /root/myconfig_ha_gen.sh -i 1
 node2# /root/myconfig_ha_gen.sh -i 2
```

ВАЖНО: Каждый узел должен иметь уникальный идентификатор.

Конфигурационный файл Percona записывается в /etc/my.cnf, где определяется идентификатор сервера и рекомендуемая конфигурация для репликации базы данных.

Перезапуск службы mysqld, чтобы убедиться, что конфигурация была применена правильно.

```
all# systemctl restart mysqld
```

Установка Pandora FMS

Новая установка Pandora FMS

Мы устанавливаем Pandora FMS во вновь созданную базу данных. Для получения дополнительной информации:

https://wiki.pandorafms.com/es:documentation:02_installation:01_installing

Остановите сервер Pandora FMS:

```
pandorafms# /etc/init.d/pandora_server stop
```

Начиная с версии NG 754 и выше, доступны дополнительные опции по ручному запуску и выключению сред высокой доступности (HA).

Существующая установка Pandora FMS

Остановите сервер Pandora FMS:

```
pandorafms# /etc/init.d/pandora_server stop
```

Мы создаем резервную копию базы данных Pandora FMS:

```
pandorafms# mysqldump -uroot -ppandora --databases pandora> /tmp/pandoradb.sql
pandorafms# scp /tmp/pandoradb.sql node1:/tmp/
```

Мы загружаем ее в новую базу данных:

```
node1# mysql -uroot -ppandora pandora -e source "/tmp/pandoradb.sql"
```

Конфигурация репликации

Мы предоставляем необходимые привилегии для репликации, чтобы она работала на всех базах данных:

```
all# mysql -uroot -ppandora
mysql> GRANT ALL ON pandora.* TO 'root'@'%' IDENTIFIED BY 'pandora';
mysql> GRANT REPLICATION CLIENT, REPLICATION SLAVE ON *.*
TO 'root'@'%' IDENTIFIED BY 'pandora';
mysql> GRANT REPLICATION CLIENT, REPLICATION SLAVE, SUPER, PROCESS, RELOAD ON
*.*
TO 'root'@'localhost' IDENTIFIED BY 'pandora';
mysql> GRANT select ON mysql.user TO 'root'@'%' IDENTIFIED BY 'pandora';
mysql> FLUSH PRIVILEGES;
mysql> quit
```

Мы создаем резервную копию базы данных первого узла и записываем имя и позицию главного файла журнала (в примере *mysql-bin.000001* и *785*):

```
node1# [ -e /root/pandoradb.bak ] && rm -rf /root/pandoradb.bak
node1# innobackupex --no-timestamp /root/pandoradb.bak/
node1# innobackupex --apply-log /root/pandoradb.bak/
node1# cat /root/pandoradb.bak/xtrabackup_binlog_info
mysql-bin.000001        785
```

Мы загружаем базу данных на второй узел и настраиваем ее на репликацию с первого узла (мы устанавливаем MASTER_LOG_FILE и MASTER_LOG_POS в значения предыдущего шага):

```
node2# systemctl stop mysqld

node1# rsync -avpP -e ssh /root/pandoradb.bak/ node2:/var/lib/mysql/

node2# chown -R mysql:mysql /var/lib/mysql
node2# chcon -R system_u:object_r:mysqld_db_t:s0 /var/lib/mysql
node2# systemctl start mysqld
node2# mysql -uroot -ppandora
mysql> CHANGE MASTER TO MASTER_HOST='node1',
MASTER_USER='root', MASTER_PASSWORD='pandora',
MASTER_LOG_FILE='mysql-bin.000001', MASTER_LOG_POS=785;
mysql> START SLAVE;
mysql> SHOW SLAVE STATUS \G
   *************************** 1. row ***************************
                Slave_IO_State: Waiting for master to send event
                   Master_Host: node1
                   Master_User: root
                   Master_Port: 3306
                 Connect_Retry: 60
               Master_Log_File: mysql-bin.000002
           Read_Master_Log_Pos: 785
```

```
                  Relay_Log_File: node2-relay-bin.000003
                   Relay_Log_Pos: 998
           Relay_Master_Log_File: mysql-bin.000002
                Slave_IO_Running: Yes
               Slave_SQL_Running: Yes
                 Replicate_Do_DB: pandora
             Replicate_Ignore_DB:
              Replicate_Do_Table:
          Replicate_Ignore_Table:
         Replicate_Wild_Do_Table:
     Replicate_Wild_Ignore_Table:
                      Last_Errno: 0
                      Last_Error:
                    Skip_Counter: 0
             Exec_Master_Log_Pos: 785
                 Relay_Log_Space: 1252
                 Until_Condition: None
                  Until_Log_File:
                   Until_Log_Pos: 0
              Master_SSL_Allowed: No
              Master_SSL_CA_File:
              Master_SSL_CA_Path:
                 Master_SSL_Cert:
               Master_SSL_Cipher:
                  Master_SSL_Key:
           Seconds_Behind_Master: 0
   Master_SSL_Verify_Server_Cert: No
                   Last_IO_Errno: 0
                   Last_IO_Error:
                  Last_SQL_Errno: 0
                  Last_SQL_Error:
     Replicate_Ignore_Server_Ids:
                Master_Server_Id: 1
                     Master_UUID: 580d8bb0-6991-11e8-9a22-16efadb2f150
                Master_Info_File: mysql.slave_master_info
                       SQL_Delay: 0
             SQL_Remaining_Delay: NULL
         Slave_SQL_Running_State: Slave has read all relay log; waiting for more
updates
              Master_Retry_Count: 86400
                     Master_Bind:
         Last_IO_Error_Timestamp:
        Last_SQL_Error_Timestamp:
                  Master_SSL_Crl:
              Master_SSL_Crlpath:
              Retrieved_Gtid_Set:
               Executed_Gtid_Set:
                   Auto_Position: 0
            Replicate_Rewrite_DB:
                    Channel_Name:
              Master_TLS_Version:
   1 row in set (0.00 sec)
```

```
mysql> quit
```

Мы должны убедиться, что Slave_IO_Running и Slave_SQL_Running показывают Yes. Другие значения могут отличаться от приведенных в примере.

Рекомендуется не использовать пользователя root для выполнения этого процесса. Рекомендуется предоставить права другому пользователю, отвечающему за управление базой данных, чтобы избежать возможных конфликтов.

Конфигурация кластера с двумя узлами

Нужно установить необходимые пакеты:

```
all# yum install -y epel-release corosync ntp pacemaker pcs

all# systemctl enable ntpd
all# systemctl enable corosync
all# systemctl enable pcsd
all# cp /etc/corosync/corosync.conf.example /etc/corosync/corosync.conf
```

```
all# systemctl start ntpd
all# systemctl start corosync
all# systemctl start pcsd
```

Мы останавливаем сервер Percona:

```
node1# systemctl stop mysqld
```

```
node2# systemctl stop mysqld
```

Аутентифицируем все узлы в кластере

Создаем и запускаем кластер:

```
all# echo hapass | passwd hacluster --stdin
```

```
node1# pcs cluster auth -u hacluster -p hapass --force node1 node2
node1# pcs cluster setup --force --name pandoraha node1 node2
node1# pcs cluster start --all
node1# pcs cluster enable --all
```

```
node1# pcs property set stonith-enabled=false
node1# pcs property set no-quorum-policy=ignore
```

Мы проверяем состояние кластера:

```
node#1 pcs status
   Cluster name: pandoraha
   Stack: corosync
   Current DC: node1 (version 1.1.18-11.el7_5.2-2b07d5c5a9) - partition with
quorum
   Last updated: Fri Jun  8 12:53:49 2018
   Last change: Fri Jun  8 12:53:47 2018 by root via cibadmin on node1

   2 nodes configured
   0 resources configured

   Online: [ node1 node2 ]

   No resources

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

Оба узла должны быть в режиме онлайн

(`Online: [ node1 node2 ]`).

Другие значения могут отличаться от приведенных в
примере.

Установите агент репликации Pacemaker от Percona

вы можете загрузить его вручную с нашего сайта librería PFMS.

```
all# cd /usr/lib/ocf/resource.d/
all# mkdir percona
all# cd percona
all# curl -L -o pacemaker_mysql_replication.zip
https://pandorafms.com/library/wp-content/uploads/2019/12/pacemaker_mysql_replic
ation.zip
all# unzip pacemaker_mysql_replication.zip
all# rm -f pacemaker_mysql_replication.zip
all# chmod u+x mysql
```

Настроим ресурсы кластера. Заменяем <VIRT_IP> на предпочитаемый виртуальный IP-адрес:

Если пароль по умолчанию, используемый в данном руководстве для пользователя root базы данных, был изменен, необходимо соответствующим образом обновить *replication_passwd* и *test_passwd*.

Имена ресурсов кластера должны быть точно такими, как указано в данном руководстве («pandoraip» и «pandoradb»)

```
node1# export VIP='<VIRT_IP>'
node1# pcs resource create pandoradb ocf:percona:mysql config="/etc/my.cnf" \
pid="/var/run/mysqld/mysqld.pid" socket="/var/lib/mysql/mysql.sock" \
replication_user="root" replication_passwd="pandora" max_slave_lag="60" \
evict_outdated_slaves="false" binary="/usr/sbin/mysqld" datadir="/var/lib/mysql"
\
test_user="root" test_passwd="pandora" op start interval="0" timeout="60s" \
op stop interval="0" timeout="60s" op promote timeout="120" op demote
timeout="120" \
op monitor role="Master" timeout="30" interval="5" on-fail="restart" op monitor
role="Slave" \
timeout="30" interval="10"
node1# pcs resource create pandoraip ocf:heartbeat:IPaddr2 ip=$VIP
cidr_netmask=24 \
op monitor interval=20s
node1# pcs resource master master_pandoradb pandoradb meta master-max="1" \
master-node-max="1" clone-max="2" clone-node-max="1" notify="true" \
globally-unique="false" target-role="Master" is-managed="true"
node1# pcs constraint colocation add master master_pandoradb with pandoraip
node1# pcs constraint order promote master_pandoradb then start pandoraip
```

Мы проверяем состояние кластера:

```
node1# pcs status
   Cluster name: pandoraha
   Stack: corosync
   Current DC: node1 (version 1.1.18-11.el7_5.2-2b07d5c5a9) - partition with
quorum
   Last updated: Fri Jun  8 13:02:21 2018
   Last change: Fri Jun  8 13:02:11 2018 by root via cibadmin on node1

   2 nodes configured
   3 resources configured

   Online: [ node1 node2 ]

   Full list of resources:

    Master/Slave Set: master_pandoradb [pandoradb]
```

```
        Masters: [ node1 ]
        Slaves: [ node2 ]
    pandoraip      (ocf::heartbeat:IPaddr2):       Started node1

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

Оба узла должны быть в режиме онлайн

(`Online: [ node1 node2 ]`).

Другие значения могут отличаться от приведенных в примере.

Настройка кластера из двух узлов с пользователем "no root"

Она будет проводиться аналогично предыдущей. Вы должны скопировать учетные данные пользователя, о чем говорилось ранее, и выполнить следующие шаги:

```
# All nodes:
```

```
 useradd <user>
 passwd <user>
 usermod -a -G haclient <user>
```

```
# Enable PCS ACL system
pcs property set enable-acl=true --force
```

```
# Create role
pcs acl role create <rol> description="RW role"  write xpath /cib
```

```
 # Create PCS user - Local user
 pcs acl user create <user> <rol>
```

```
 # Login into PCS from ALL nodes
 su - <user>
 pcs status
 Username: <user>
 Password: *****
```

```
# Wait for 'Authorized' message, ignore output. Wait a second and retry 'pcs
status' command
```

Конфигурация Pandora FMS

Убедитесь, что php-pecl-ssh2 установлен в соответствии с установленной ОС и версией:

RHEL 8

```
 pandorafms# dnf install --disablerepo=rhel-8-for-x86_64-appstream-rpms php-
pecl-ssh2
 pandorafms# systemctl restart php-fpm
 pandorafms# systemctl restart httpd
```

Rocky Linux 8

```
 pandorafms# dnf install php-pecl-ssh2
 pandorafms# systemctl restart php-fpm
 pandorafms# systemctl restart httpd
```

CentOS 7

```
 pandorafms# yum install php-pecl-ssh2
 pandorafms# systemctl restart httpd
```

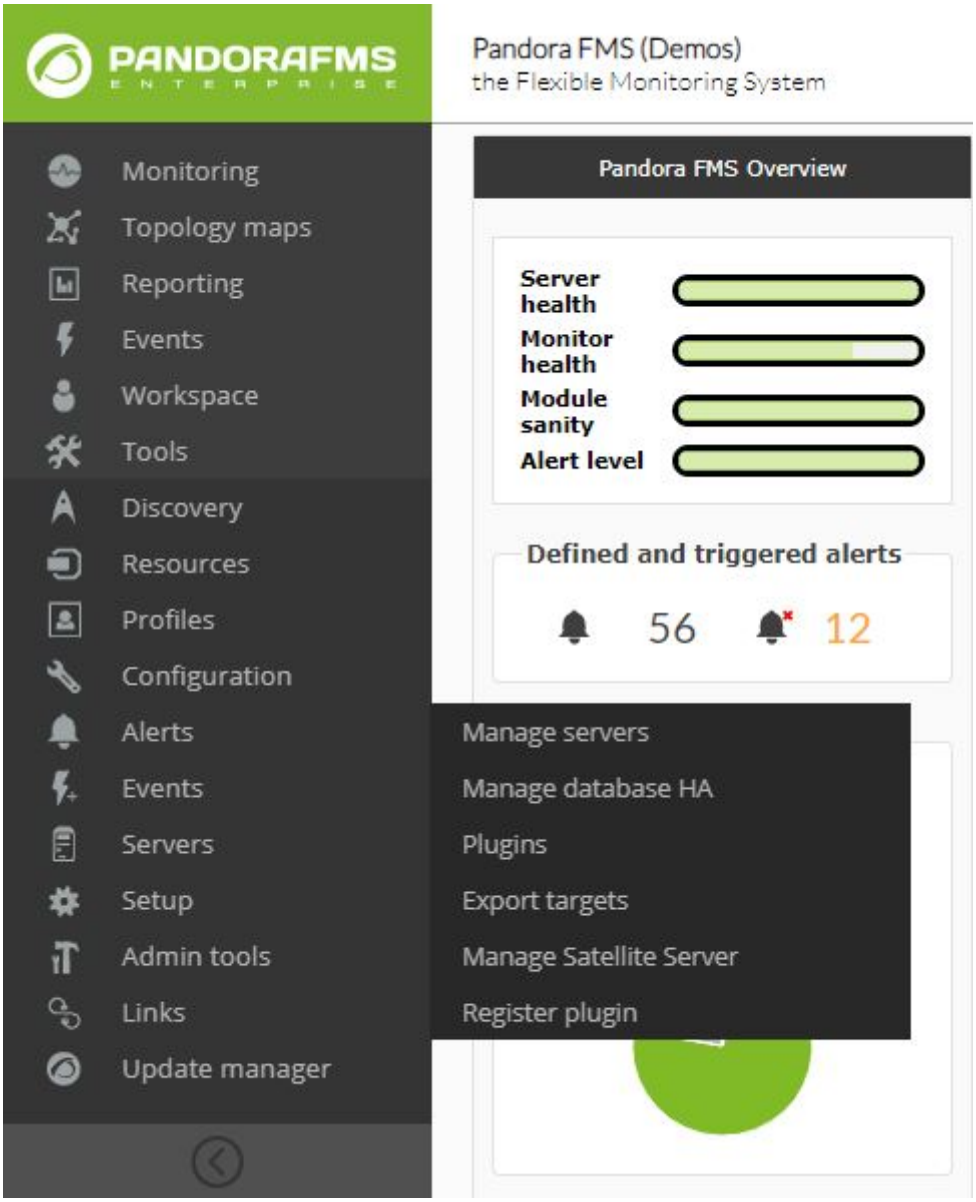
В */etc/pandora/pandora_server.conf* есть два параметра, которые управляют поведением инструмента HA базы данных Pandora FMS. Они могут быть изменены в соответствии с нашими потребностями:

```
# Pandora FMS Database HA Tool execution interval in seconds (PANDORA FMS
ENTERPRISE ONLY).
ha_interval 30

# Pandora FMS Database HA Tool monitoring interval in seconds. Must be a
multiple of ha_interval (PANDORA FMS ENTERPRISE ONLY).
ha_monitoring_interval 60
```

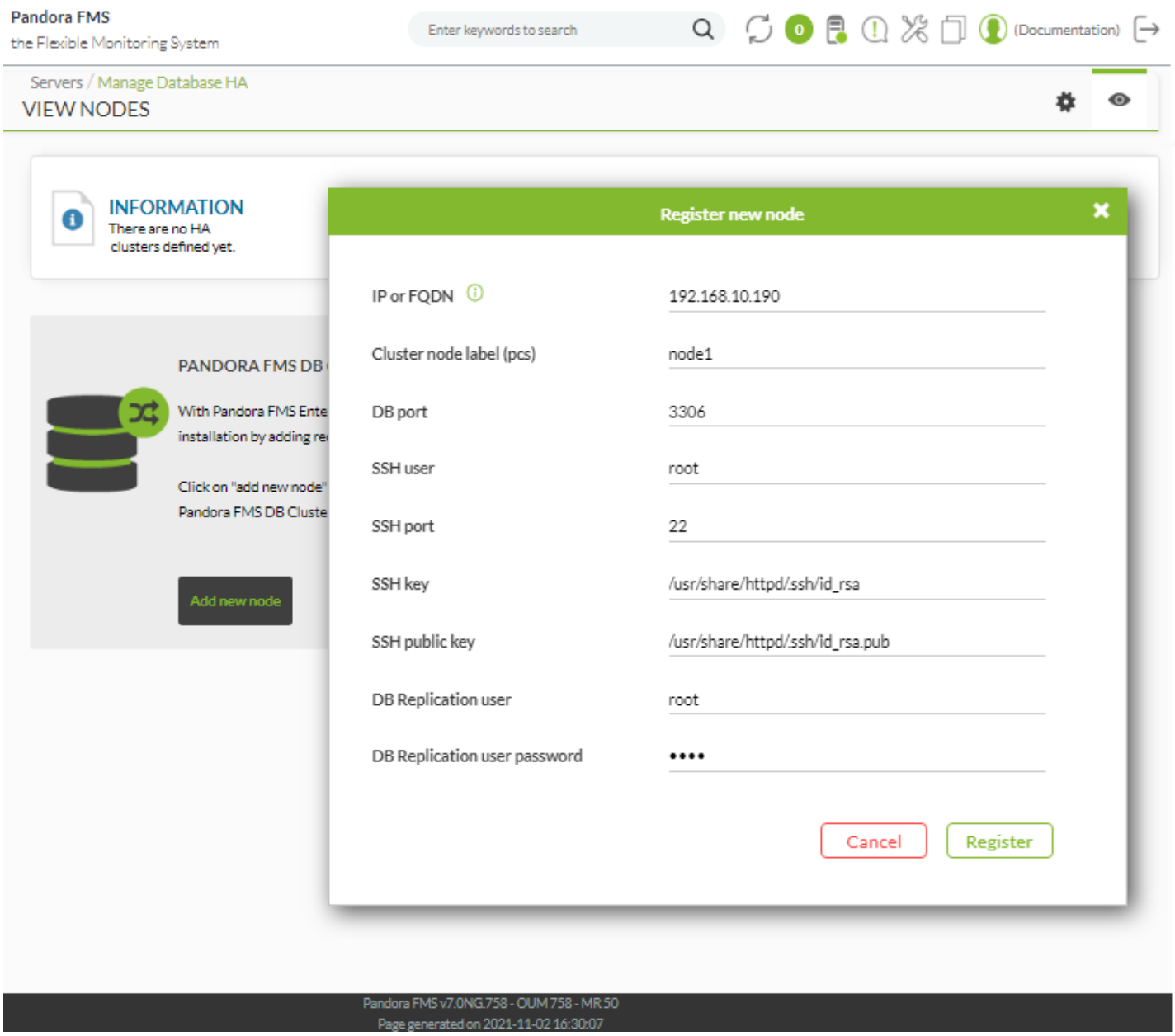
Мы направляем Pandora FMS на главный виртуальный IP-адрес (заменяя <IP> на виртуальный IP-адрес):

```
pandorafms# export VIRT_IP=<IP>
pandorafms# sed -i -e "s/^dbhost .*/dbhost $VIRT_IP/" \
/etc/pandora/pandora_server.conf
pandorafms# sed -i -e
"s/\$config\[\"dbhost\"\]=\".*\";/\$config[\"dbhost\"]=\"$VIRT_IP\";/" \
/var/www/html/pandora_console/include/config.php
```

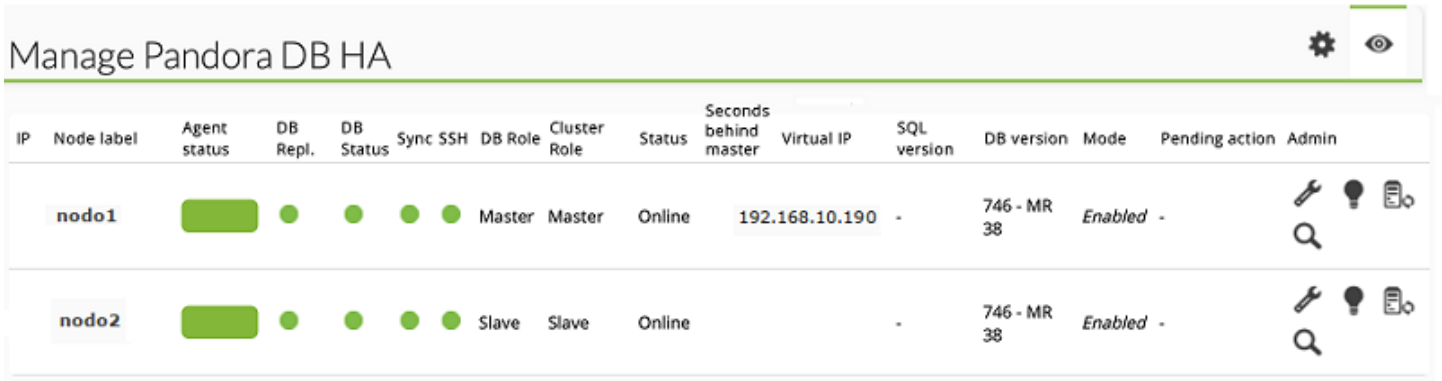
Запускаем сессию в консоли Pandora FMS и переходим в *Servers* → *Manage database HA*:



Нажмите *Add new node* и создайте запись для первого узла:



Затем нажмите *Create slave* и добавьте новую запись во второй узел. Это должно выглядеть примерно так:

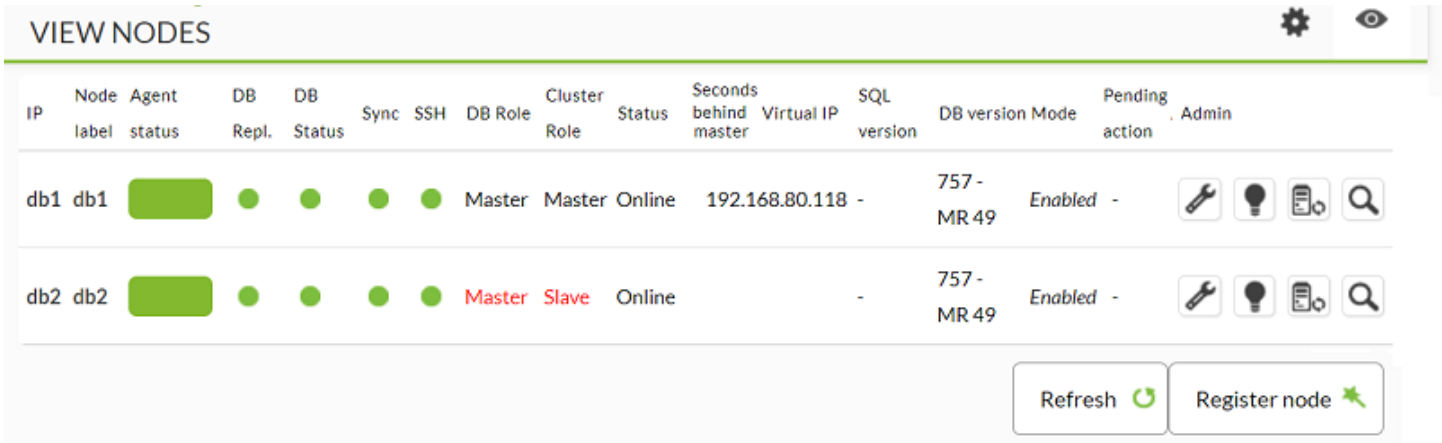


Seconds behind master должен быть около 0. Если он продолжает расти, репликация не увеличивается.

Автоматическое восстановление узлов в Splitbrain

Сценарий.

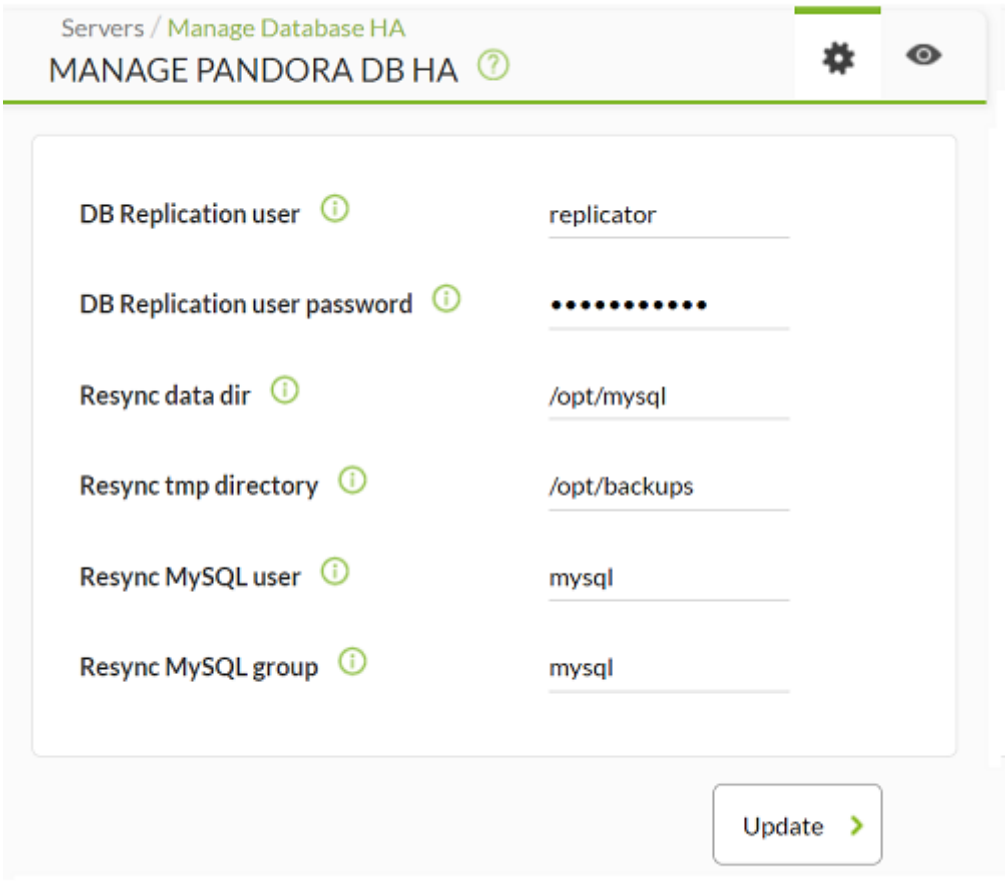
Оба сервера являются основными, или, в режиме просмотра консоли HA, оба выглядят основными, но виртуальный IP находится только на одном узле (том, который фактически выступает в качестве основного).



В этот момент, если параметр *token* splitbrain_autofix установлен в 1, начнется процесс восстановления узла в *splitbrain*.

Для правильной работы этой функциональности необходимо правильно настроить следующие компоненты:

- Ключи корневого пользователя SSH, общие для сервера `pandora_ha master` и всех серверов баз данных.
- Пользователь Replicator, настроенный в настройке с правами от сервера, на котором размещен сервер `pandora_ha master`.



- Свободное место для резервного копирования базы данных на обоих серверах, где размещены 2 базы данных (первичный и вторичный Master/Slave).

В случае, если каталог данных и путь (`datadir`, `path`),на котором будет создан раздел, находятся в одном разделе, необходимо, чтобы свободное пространство составляло не менее 50%.

Если все вышеперечисленные пункты настроены правильно, процесс восстановления происходит следующим образом:

1. Удалите предыдущие резервные копии.
2. Создайте резервную копию `datadir` вторичного узла.
3. Резервное копирование основного узла.
4. Отправляет резервную копию основного узла на вторичный узел (*Master → Slave)*.
5. Запускает ресурс «вторичного» узла с соответствующими параметрами ресинхронизации на момент резервного копирования.
6. Проверяет, что ресурс активен и корректен. Для этого он должен использовать конфигурацию, указанную в параметрах ha_max_resync_wait_retries и ha_resync_sleep.

В случае сбоя в любой точке процесса, он должен повторить процесс столько раз, сколько указано параметром ha_max_splitbrain_retries.

После завершения процесса в представлении событий появится событие, указывающее на успешное завершение процесса.

Если среда по-прежнему не восстанавливается автоматически, то вторичный узел переходит в режим ожидания, а в представлении событий появляется событие, указывающее, что восстановление должно быть выполнено вручную.

Добавление нового узла в кластер

Повторите описанные выше шаги для установки node1 и node2, в зависимости от версии, которая будет использоваться на новом узле:

RHEL 8 o Rocky Linux 8 (PFMS Версия 760 или более поздняя).

RedHat 7 o CentOS 7 (PFMS Версия 759 или более ранняя).

Уберите баннер с надписью Open SSH:

```
node3# [ -f /etc/cron.hourly/motd_rebuild ] && rm -f
/etc/cron.hourly/motd_rebuild
node3# sed -i -e 's/^Banner.*//g' /etc/ssh/sshd_config
node3# systemctl restart sshd
```

Создадим новые ключи аутентификации SSH для нового узла и копируем открытый ключ для каждого узла:

Ключи могут быть сгенерированы для пользователя, не являющегося пользователем root, для установки кластера с пользователем «no root».

```
node1# ssh-copy-id -p22 root@node3
node1# ssh node3
```

```
node2# ssh-copy-id -p22 root@node3
node2# ssh node3
```

```
pandorafms# ssh-copy-id -p22 root@node3
pandorafms# ssh node3
```

```
node3# echo -e "\n\n\n" | ssh-keygen -t rsa
node3# ssh-copy-id -p22 root@node1
node3# ssh-copy-id -p22 root@node2
node3# ssh node1
```

```
node3# ssh node2
```

В узле Pandora FMS скопируйте файл «known_hosts» для пользователя apache:

```
pandorafms# yes | cp /root/.ssh/known_hosts /usr/share/httpd/.ssh/
```

Следующие шаги необходимы только в том случае, если узлы используют SSH на нестандартном порту. Замените 22 на правильный номер порта:

```
all# echo -e "Host node1\n    Port 22">> /root/.ssh/config
all# echo -e "Host node2\n    Port 22">> /root/.ssh/config
all# echo -e "Host node3\n    Port 22">> /root/.ssh/config
```

Нужно установить необходимые пакеты:

```
node3# yum install -y
http://www.percona.com/downloads/percona-release/redhat/0.1-4/percona-release-0.
1-4.noarch.rpm
node3# yum install -y Percona-Server-server-57 percona-xtrabackup-24
```

Мы убедимся, что служба Percona деактивирована, поскольку она управляется кластером:

```
node3# systemctl stop mysqld
node3# systemctl disable mysqld
```

Если системная служба не отключена, менеджер
ресурсов кластера будет работать неправильно.

Мы настраиваем Percona, заменяя <ID> на номер, который должен быть уникальным для каждого узла кластера:

Репликация базы данных не будет работать, если два
узла имеют одинаковый SERVER_ID.

После установки *server* вы найдете в пути генератор конфигурации для репликации базы данных:

```
/usr/share/pandora_server/util/myconfig_ha_gen.sh
```

```
Example: ./myconfig_ha_gen.sh -i serverid [-l file_location] [-d database] [-b
binlog] [-u dbuser] [-p dbpass] [-s poolsize] [-h help]
Mandatory parameters:
    -i --serverid Set the server id for the database (Mandatory)
```

```
Optional parameters:
    -l --location Set my.cnf custom location including filename. [ default
value: /etc/my.cnf ] (optional)
    -d --database Set the database to be replicated. [ default value: pandora ]
(optional)
    -b --binlog Set binlog file. [ default value: mysql-bin ] (optional)
    -u --dbuser Set dbuser for mysql connection and backups. [ default value:
root ] (optional)
    -p --dbpass Set dbpassword for mysql connection and backups. [ default
value: pandora ] (optional)
    -s --poolsize Set innodb_buffer_pool_size static size in M (Megabytes) or G
(Gigabytes). [ default value: autocalculated ] (optional)
    -h --help Print help.
```

В текущем случае, когда базы данных находятся не на том же сервере, что и приложение, необходимо скопировать *script* на узлы для локального выполнения.

```
pandorafms#
scp /usr/share/pandora_server/util/myconfig_ha_gen.sh root@node3:/root/
```

Необходимо будет только передать параметр serverid (обязательный) в стандартных средах некоторые необязательные параметры для пользовательских сред.

Если заданный по умолчанию или заданный пользователь не подключается к базе данных, сценарий завершится с ошибкой подключения.

У вас также есть возможность передать имя базы данных, пользователя и пароль в качестве аргументов. В противном случае будут использоваться настройки по умолчанию.

В этом случае он выполнит сценарий на обоих узлах, передав `server id` только в том случае, если он имеет учетные данные по умолчанию, в противном случае он должен определить необходимые параметры.

```
node3#
/root/myconfig_ha_gen.sh -i 3
```

Запустим сервер Percona:

node3# systemctl start mysqld

Будет сгенерирован новый временный пароль, подключенный к /var/log/mysqld.log. Мы подключаемся к серверу Percona и меняем пароль root:

```
node3# mysql -uroot -p$(grep "temporary password" /var/log/mysqld.log | \
rev | cut -d' ' -f1 | rev)
mysql> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('Pandor4!');
mysql> UNINSTALL PLUGIN validate_password;
mysql> SET PASSWORD FOR 'root'@'localhost' = PASSWORD('pandora');
```

```
mysql> quit
```

Мы делаем резервную копию базы данных главного узла (в данном примере - node1) и записываем имя и позицию главного файла журнала (в примере - mysql-bin.000001 и 785):

```
node1# [ -e /root/pandoradb.bak ] && rm -rf /root/pandoradb.bak
node1# innobackupex --no-timestamp /root/pandoradb.bak/
node1# innobackupex --apply-log /root/pandoradb.bak/
node1# cat /root/pandoradb.bak/xtrabackup_binlog_info
mysql-bin.000001        785
```

Загрузим базу данных в новый узел, который назовем узел3, и настроим его на репликацию с узла1 (мы установим MASTER_LOG_FILE и MASTER_LOG_POS на значения из предыдущего шага):

```
node3# systemctl stop mysqld

node1# rsync -avpP -e ssh /root/pandoradb.bak/ node3:/var/lib/mysql/

node3# chown -R mysql:mysql /var/lib/mysql
node3# chcon -R system_u:object_r:mysqld_db_t:s0 /var/lib/mysql
node3# systemctl start mysqld
node3# mysql -uroot -ppandora
mysql> CHANGE MASTER TO MASTER_HOST='node1',
MASTER_USER='root', MASTER_PASSWORD='pandora',
MASTER_LOG_FILE='mysql-bin.000001', MASTER_LOG_POS=785;
mysql> START SLAVE;
mysql> SHOW SLAVE STATUS \G
   *************************** 1. row ***************************
                Slave_IO_State: Waiting for master to send event
                   Master_Host: node1
                   Master_User: root
                   Master_Port: 3306
                 Connect_Retry: 60
               Master_Log_File: mysql-bin.000002
           Read_Master_Log_Pos: 785
                Relay_Log_File: node3-relay-bin.000003
                 Relay_Log_Pos: 998
         Relay_Master_Log_File: mysql-bin.000002
              Slave_IO_Running: Yes
             Slave_SQL_Running: Yes
               Replicate_Do_DB: pandora
           Replicate_Ignore_DB:
            Replicate_Do_Table:
        Replicate_Ignore_Table:
       Replicate_Wild_Do_Table:
   Replicate_Wild_Ignore_Table:
                    Last_Errno: 0
                    Last_Error:
                  Skip_Counter: 0
```

```
             Exec_Master_Log_Pos: 785
                 Relay_Log_Space: 1252
                 Until_Condition: None
                  Until_Log_File:
                   Until_Log_Pos: 0
              Master_SSL_Allowed: No
              Master_SSL_CA_File:
              Master_SSL_CA_Path:
                 Master_SSL_Cert:
               Master_SSL_Cipher:
                  Master_SSL_Key:
           Seconds_Behind_Master: 0
   Master_SSL_Verify_Server_Cert: No
                   Last_IO_Errno: 0
                   Last_IO_Error:
                  Last_SQL_Errno: 0
                  Last_SQL_Error:
     Replicate_Ignore_Server_Ids:
                Master_Server_Id: 1
                     Master_UUID: 580d8bb0-6991-11e8-9a22-16efadb2f150
                Master_Info_File: mysql.slave_master_info
                       SQL_Delay: 0
             SQL_Remaining_Delay: NULL
         Slave_SQL_Running_State: Slave has read all relay log; waiting for more
updates
               Master_Retry_Count: 86400
                      Master_Bind:
          Last_IO_Error_Timestamp:
         Last_SQL_Error_Timestamp:
                   Master_SSL_Crl:
               Master_SSL_Crlpath:
               Retrieved_Gtid_Set:
                Executed_Gtid_Set:
                    Auto_Position: 0
            Replicate_Rewrite_DB:
                    Channel_Name:
              Master_TLS_Version:
   1 row in set (0.00 sec)
mysql> quit

node3# systemctl stop mysqld
```

Важно обеспечить, чтобы Slave_IO_Running и Slave_SQL_Running показывали Yes. Другие значения могут отличаться от приведенного примера.

Мы устанавливаем необходимые пакеты для кластера:

```
 node3# yum install -y epel-release corosync ntp pacemaker pcs
```

```
 node3# systemctl enable ntpd
 node3# systemctl enable corosync
 node3# systemctl enable pcsd

 node3# systemctl start ntpd
```

Мы добавляем новый узел в кластер:

```
node3# echo -n hapass | passwd hacluster --stdin
node3# cd /usr/lib/ocf/resource.d/
node3# mkdir percona
node3# cd percona
node3# curl -L -o pacemaker_mysql_replication.zip
https://pandorafms.com/library/wp-content/uploads/2019/12/pacemaker_mysql_replic
ation.zip
node3# unzip pacemaker_mysql_replication.zip
node3# rm -f pacemaker_mysql_replication.zip
node3# chmod u+x mysql
```

```
 node1# pcs cluster auth -u hacluster -p hapass --force node3
 node1# pcs cluster node add --enable --start node3
```

Установим clone-max на количество узлов в нашем кластере (3 в данном примере):

```
node3# pcs resource update master_pandoradb meta master-max="1" \
master-node-max="1" clone-max="3" clone-node-max="1" notify="true" \
globally-unique="false" target-role="Master" is-managed="true"
```

Мы проверяем состояние узла:

```
node3# pcs status
Cluster name: pandoraha
Stack: corosync
Current DC: node1 (version 1.1.18-11.el7_5.2-2b07d5c5a9) - partition with quorum
Last updated: Fri Jun  1 10:55:47 2018
Last change: Fri Jun  1 10:55:09 2018 by root via crm_attribute on node3

3 nodes configured
3 resources configured

Online: [ node1 node2 node3 ]

Full list of resources:

pandoraip      (ocf::heartbeat:IPaddr2):       Started node1
Master/Slave Set: master_pandoradb [pandoradb]
    Masters: [ node1 ]
    Slaves: [ node2 node3 ]

Daemon Status:
```

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

Все узлы должны быть в режиме онлайн.

(`Online: [ node1 node2 node3 ]`).

Другие значения могут отличаться от приведенного
примера.

Регистрируем узел кластера в консоли Pandora из меню «Servers → Manage database HA».

Восстановление сломанного узла

В качестве примера мы будем использовать node2. Мы переводим node2 в режим ожидания:

```
node2# pcs node standby node2
node2# pcs status
    Cluster name: pandoraha
    Stack: corosync
    Current DC: node1 (version 1.1.18-11.el7_5.2-2b07d5c5a9) - partition with
quorum
    Last updated: Tue Jun 12 08:20:49 2018
    Last change: Tue Jun 12 08:20:34 2018 by root via cibadmin on node2

    2 nodes configured
    3 resources configured

    Node node2: standby
    Online: [ node1 ]

    Full list of resources:

     Master/Slave Set: master_pandoradb [pandoradb]
         Masters: [ node1 ]
         Stopped: [ node2 ]
     pandoraip      (ocf::heartbeat:IPaddr2):       Started node1

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

node2 должен находиться в режиме ожидания

(Node node2: standby).

Другие значения могут отличаться от приведенного
примера.

Мы создаем резервную копию каталога данных в Percona:

```
node2# systemctl stop mysqld
node2# [ -e /var/lib/mysql.bak ] && rm -rf /var/lib/mysql.bak
node2# mv /var/lib/mysql /var/lib/mysql.bak
```

Создаем резервную копию базы данных главного узла (в данном примере node1) и обновляем имя главного узла, а также имя и положение главного файла журнала в кластере (в данном примере node1, mysql-bin.000001 и 785):

```
node1# [ -e /root/pandoradb.bak ] && rm -rf /root/pandoradb.bak
node1# innobackupex --no-timestamp /root/pandoradb.bak/
node1# innobackupex --apply-log /root/pandoradb.bak/
node1# binlog_info=$(cat /root/pandoradb.bak/xtrabackup_binlog_info)
node1# crm_attribute --type crm_config --name pandoradb_REPL_INFO -s
mysql_replication \
-v "node1|$(echo $binlog_info | awk '{print $1}')|$(echo $binlog_info | awk
'{print $2}')"
```

Мы загружаем базу данных сломанного узла:

```
 node1# rsync -avpP -e ssh /root/pandoradb.bak/ node2:/var/lib/mysql/

 node2# chown -R mysql:mysql /var/lib/mysql
 node2# chcon -R system_u:object_r:mysqld_db_t:s0 /var/lib/mysql
```

Мы деактивируем режим ожидания узла2:

```
 node2# pcs node unstandby node2
 node2# pcs resource cleanup --node node2
```

Мы проверяем состояние кластера:

```
node2# pcs status
Cluster name: pandoraha
Stack: corosync
Current DC: node1 (version 1.1.18-11.el7_5.2-2b07d5c5a9) - partition with quorum
Last updated: Fri Jun  1 10:55:47 2018
Last change: Fri Jun  1 10:55:09 2018 by root via crm_attribute on node3

2 nodes configured
3 resources configured
```

```
Online: [ node1 node2 ]

Full list of resources:

pandoraip      (ocf::heartbeat:IPaddr2):       Started node1
Master/Slave Set: master_pandoradb [pandoradb]
    Masters: [ node1 ]
    Slaves: [ node2 ]

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

Оба узла должны быть в режиме онлайн.

(`Online: [ node1 node2 ]`).

Другие значения могут отличаться от приведенного примера.

Мы проверяем состояние репликации базы данных:

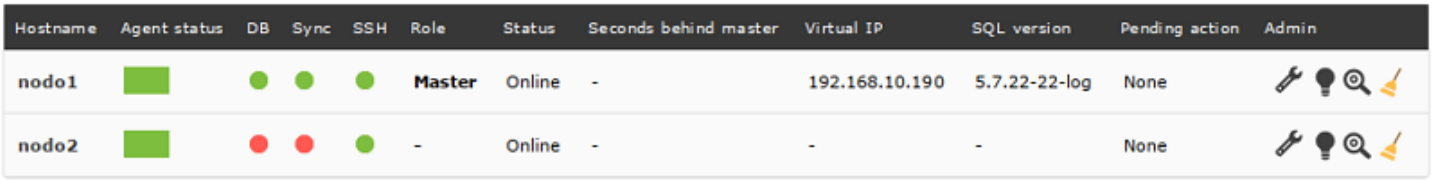
```
node2# mysql -uroot -ppandora
 mysql> SHOW SLAVE STATUS \G
   *************************** 1. row ***************************
                Slave_IO_State: Waiting for master to send event
                   Master_Host: node1
                   Master_User: root
                   Master_Port: 3306
                  Connect_Retry: 60
                Master_Log_File: mysql-bin.000001
            Read_Master_Log_Pos: 785
                 Relay_Log_File: node2-relay-bin.000003
                  Relay_Log_Pos: 998
          Relay_Master_Log_File: mysql-bin.000001
               Slave_IO_Running: Yes
              Slave_SQL_Running: Yes
                Replicate_Do_DB: pandora
            Replicate_Ignore_DB:
             Replicate_Do_Table:
         Replicate_Ignore_Table:
        Replicate_Wild_Do_Table:
    Replicate_Wild_Ignore_Table:
                     Last_Errno: 0
                     Last_Error:
                   Skip_Counter: 0
            Exec_Master_Log_Pos: 785
                Relay_Log_Space: 1252
```

```
                Until_Condition: None
                 Until_Log_File:
                  Until_Log_Pos: 0
             Master_SSL_Allowed: No
             Master_SSL_CA_File:
             Master_SSL_CA_Path:
                Master_SSL_Cert:
              Master_SSL_Cipher:
                 Master_SSL_Key:
          Seconds_Behind_Master: 0
   Master_SSL_Verify_Server_Cert: No
                 Last_IO_Errno: 0
                 Last_IO_Error:
                Last_SQL_Errno: 0
                Last_SQL_Error:
     Replicate_Ignore_Server_Ids:
               Master_Server_Id: 1
                    Master_UUID: 580d8bb0-6991-11e8-9a22-16efadb2f150
               Master_Info_File: mysql.slave_master_info
                      SQL_Delay: 0
            SQL_Remaining_Delay: NULL
        Slave_SQL_Running_State: Slave has read all relay log; waiting for more
updates
             Master_Retry_Count: 86400
                    Master_Bind:
        Last_IO_Error_Timestamp:
       Last_SQL_Error_Timestamp:
                 Master_SSL_Crl:
             Master_SSL_Crlpath:
             Retrieved_Gtid_Set:
              Executed_Gtid_Set:
                  Auto_Position: 0
           Replicate_Rewrite_DB:
                   Channel_Name:
             Master_TLS_Version:
   1 row in set (0.00 sec)
```

Важно обеспечить, чтобы Slave_IO_Running и Slave_SQL_Running показывали Yes. Другие значения могут отличаться от приведенного примера.

Решение проблем

Что делать, если один из узлов кластера не работает?



Hostname	Agent status	DB	Sync	SSH	Role	Status	Seconds behind master	Virtual IP	SQL version	Pending action	Admin
nodo1					Master	Online	-	192.168.10.190	5.7.22-22-log	None	
nodo2					-	Online	-	-	-	None	

Служба не будет затронута до тех пор, пока главный узел работает. Если главный узел падает, вторичный узел автоматически становится главным. Смотрите: Восстановление сломанного узла.

Вернуться в оглавление Документации Pandora FMS

1) https://aws.amazon.com/ru/ec2/instance-types/t2/?nc1=h_ls