# Развёртывание кластера

## Master

Устанавливаем на сервер БД



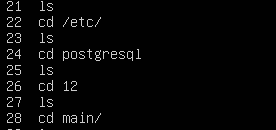
Создаём пользователя



Создаём БД

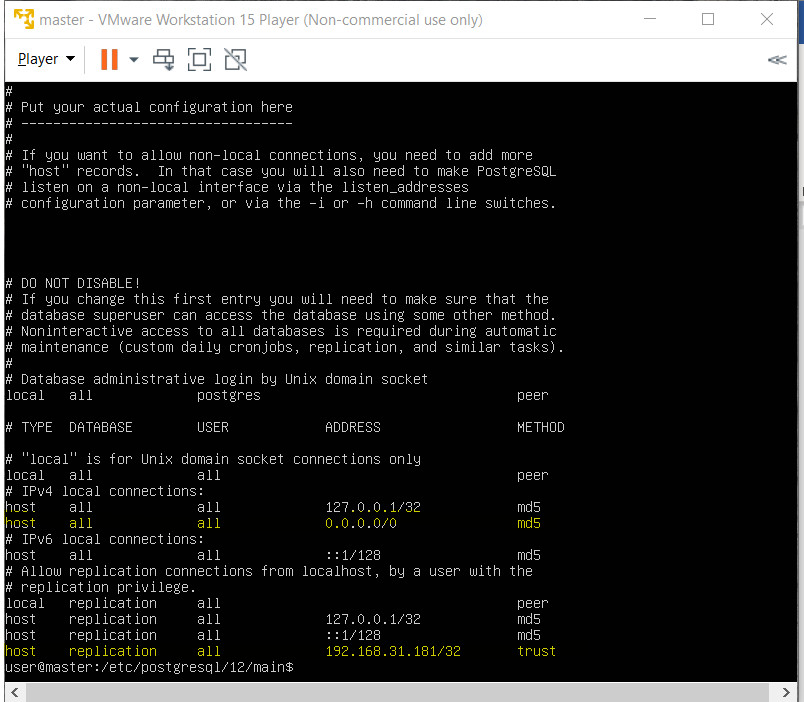


Идём в папку конфигураций БД

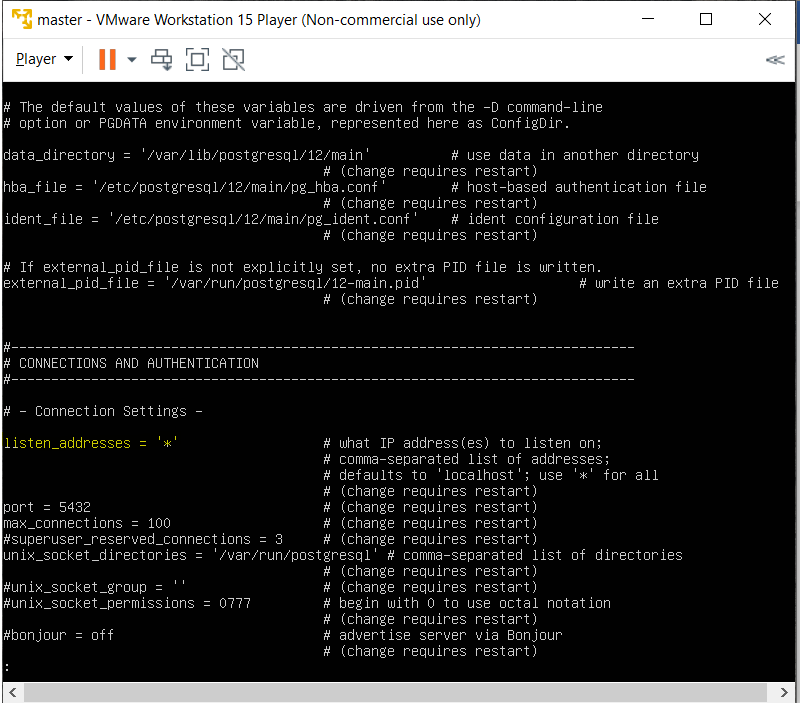


И редактируем файл pg\_hba.conf и postgresql.conf

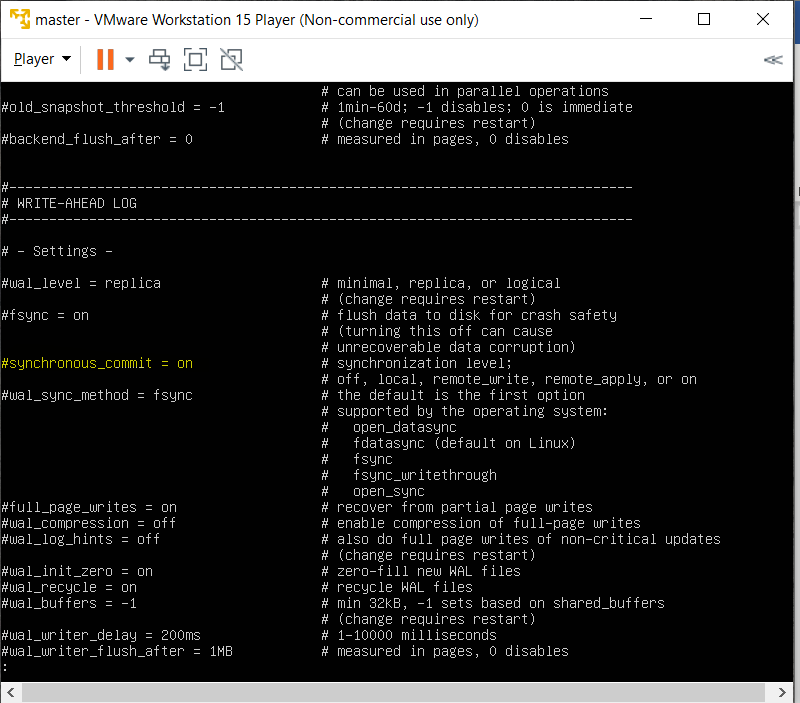
**pg\_hba.conf**: разрешили подключиться к БД из интернета по паролю и для задачи репликации разрешили пускать без пароля с сервера с ip 192.168.31.181 (slave)



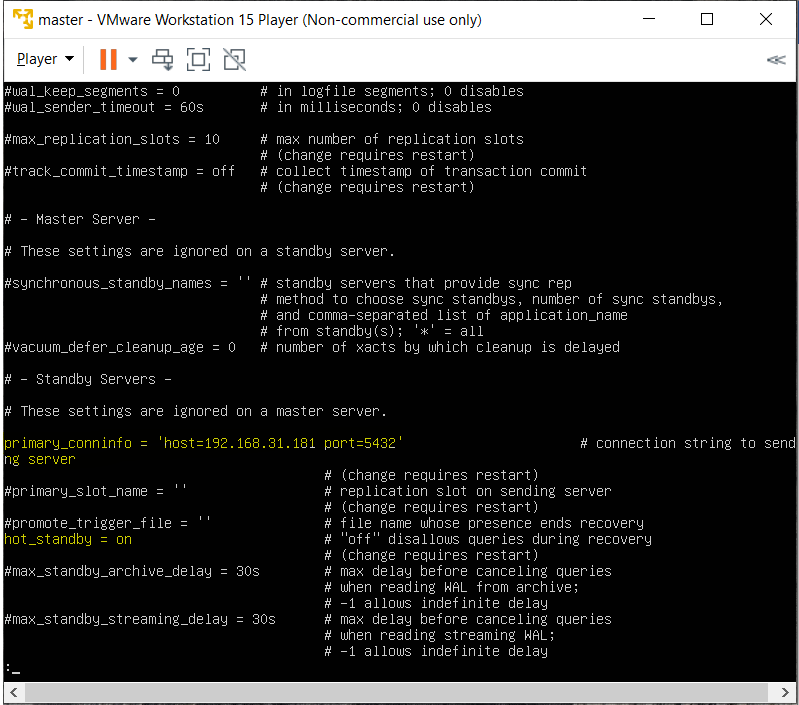
**postgresql.conf**: указываем БД принимать соединение с любого адреса



Убеждаемся, что включена синхронная репликация



Указываем данные другого сервера БД и включаем возможность принимать запросы во время синхронизации



После редактирования файлов перезапускаем БД



## Slave

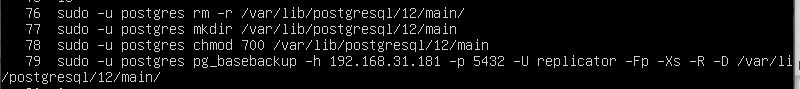
Выполняем все такие же действия, как и для мастера + включаем репликацию:

1)Удаляем существующую БД

2)Создаём папку для БД

3) Назначаем права на папку

4) Синхронизируем мастера и slave и активируем репликацию



Затем перезапускаем БД

## Проверка работоспособность программы-агента

Устанавливаем питон и зависимости для нашей программы-агента

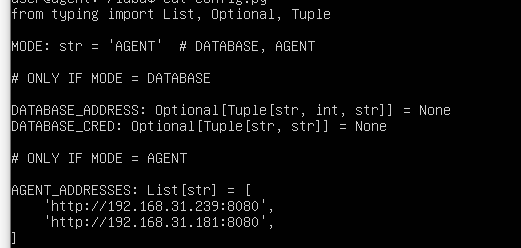


Редактируем файл конфигурации программы агента

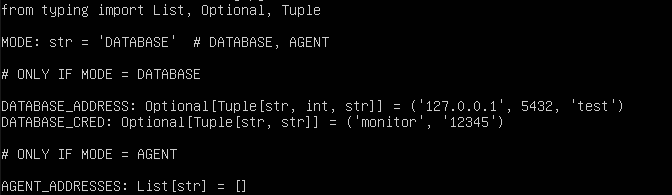


Для арбитра файл выглядит следующим образом:

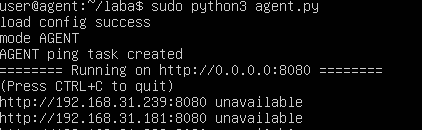
1. Указываем режим работы
2. Прописываем адреса других серверов с БД

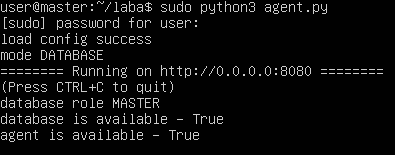


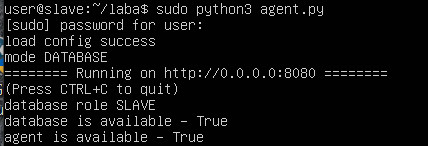
Для БД файл конфигурации выглядит следующим образом:



Запускаем на всех серверах программу агента от имени суперпользователя для того, чтобы программа могла управлять состоянием сервера

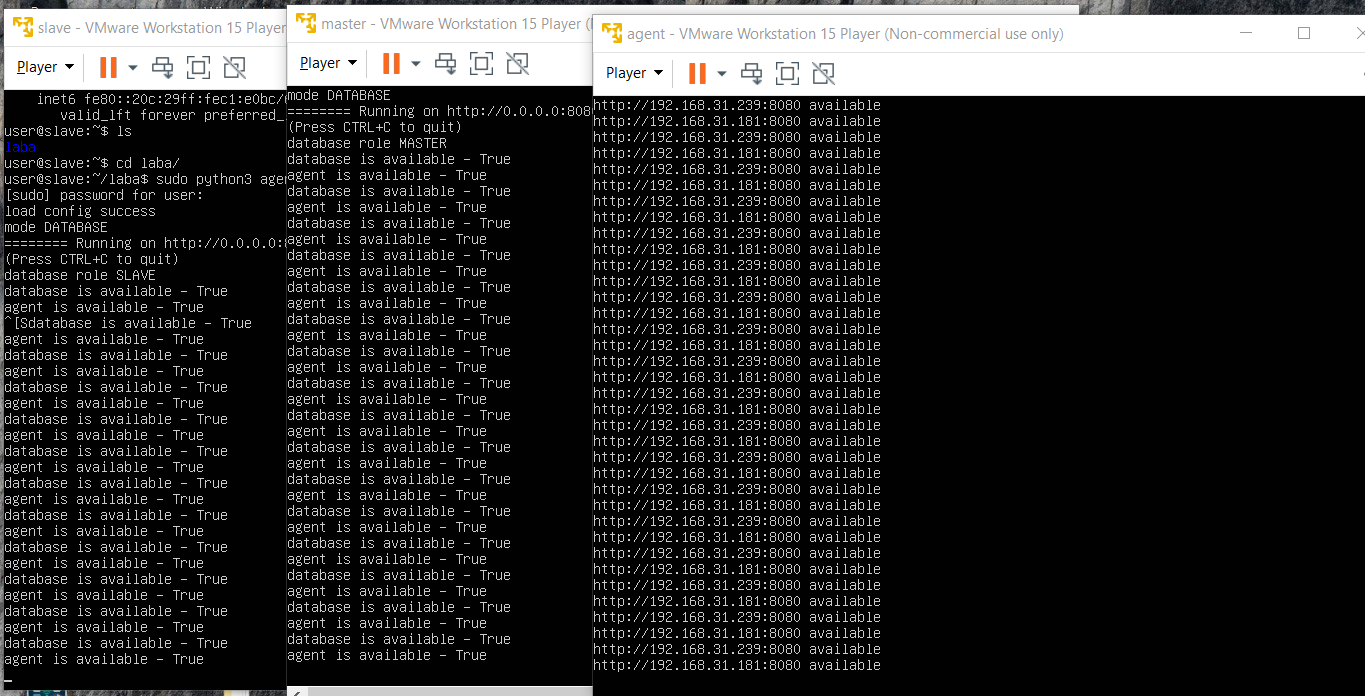






Заметим, что программа агент сама определила кто мастер, а кто slave.

Всё запущено и система работает стабильно:



# Проверки

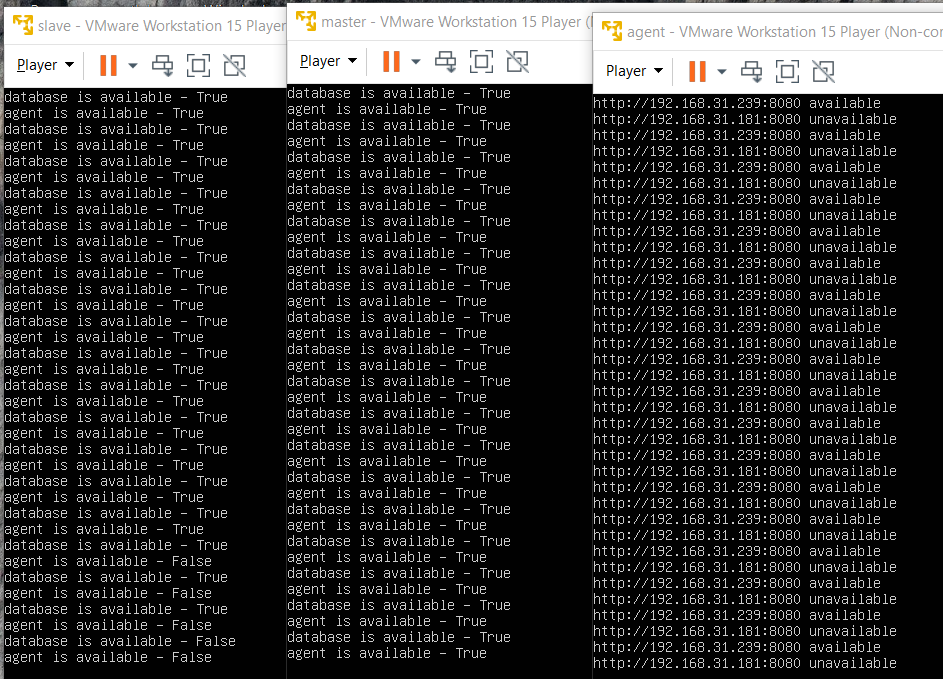
Проверки выполняются путём 100% потери пакетов, т.е. все программы остаются живы

1. **Проверим, что в случае отказа slave мастер остаётся жив, slave не повышается до мастера**

Агент наблюдает отсутствие связи с сервером по адресу 192.168.31.181, который является slave.

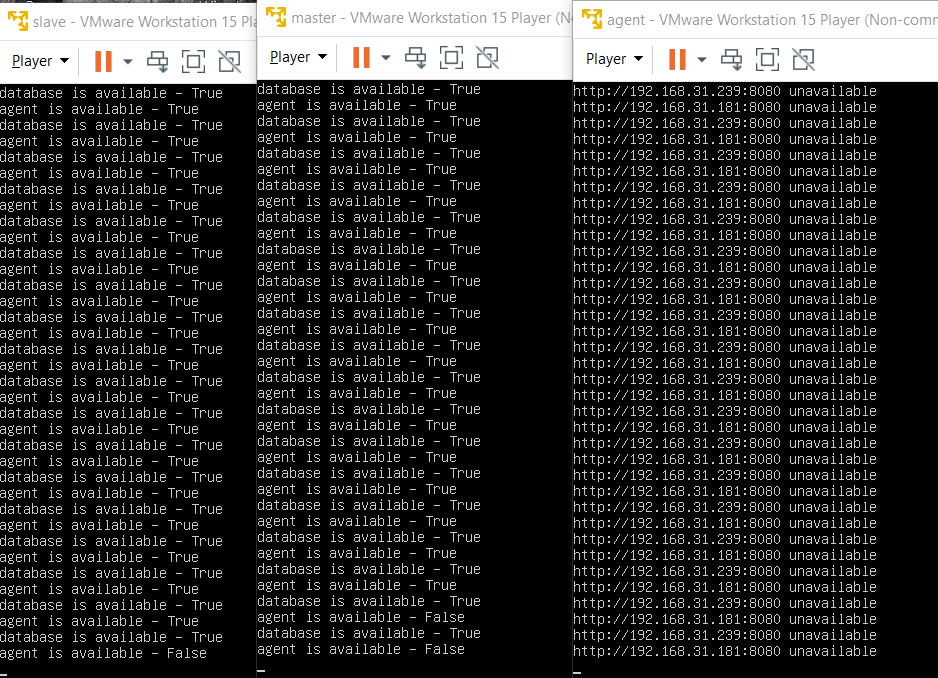
На мастере всё хорошо.

Slave в первую очередь обнаруживает, что отпал агент, а затем и синхронизация баз. Из полученных данных делает вывод, что проблемы на его стороне и не повышает себя до мастера.



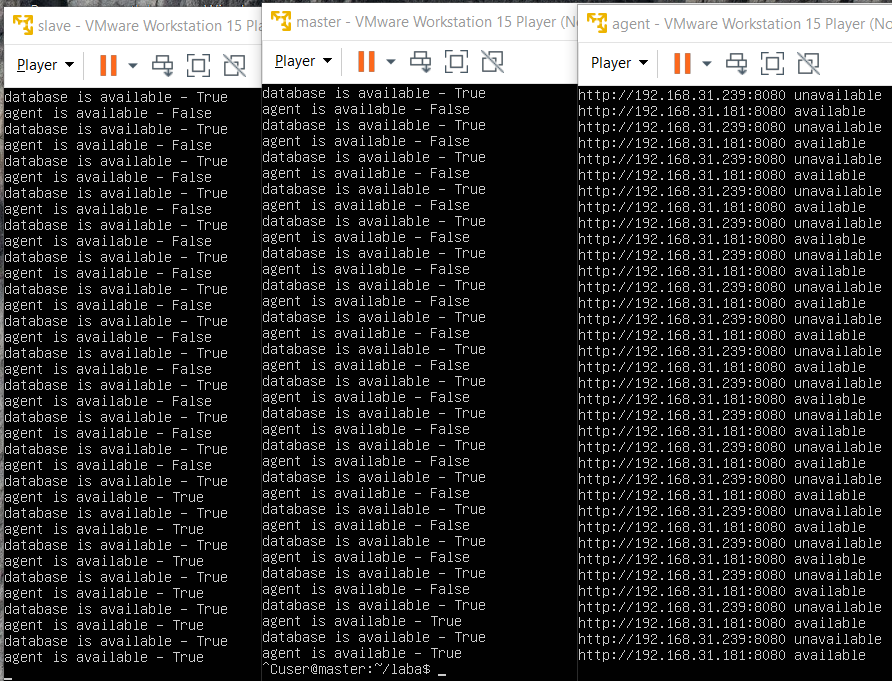
1. **Потеря арбитра**

Агент выявили недоступность арбитра, но синхронизация баз сохраняется, поэтому мастер и slave ничего не предпринимают.



1. **Падение программы агента на мастере**

Арбитр наблюдает отсутствие мастера, slave видит, что синхронизация баз работает, поэтому ничего не предпринимает. Соответственно кластер переживает перезагрузку программ агентов.



1. **Падение мастера**

Арбитр заметил пропажу мастера, но slave остаётся доступным. Slave заметил отсутствие синхронизации с мастером, но наличие связи с арбитром. Поэтому принимает решение повысить себя до мастера. Мастер понял, что пропала связь с арбитром и нет синхронизации со slave, поэтому принимает решение отключиться.

