

Памятка:

	ВЫПОЛНЯЕМ ОДИНАКОВО КОМАНДЫ НА ОБОИХ ВЕБ-СЕРВЕРАХ
	ВЫПОЛНЯЕМ КОМАНДУ ИНДИВИДУАЛЬНО НА СЕРВЕРЕ

## I

Устанавливаем пакеты приложений, которые нам будут необходимы для работы веб-сервера на базе Apache2, части ядра СУБД необходимого, для удаленного подключения и тестирования СУБД, а также интерпретатора PHP и его пакетов для установки и настройки CMS на базе Wordpress:

```
sudo apt update  
  
sudo apt install apache2 nano wget unzip mariadb-server php-mysql apache2 libapache2-mod-php php-gd php-xml php-mbstring php-curl curl -y
```

Далее по заданию необходимо установить CMS Wordpress на 2 веб-сервера.

Так как мы настроили синхронизацию данных с помощью RSYNC и планировщика Cron, наш скаченный настроенный и установленный CMS на одном веб-сервере, должен сначала «скопироваться» и синхронизироваться на втором веб-сервере, в тот момент, когда мы работаем в каталоге /var/www/html на первом веб-сервере.

Выше, мы уже установили компоненты веб-сервера и интерпретатора PHP с необходимыми библиотеками для CMS.

Далее необходимо выполнить следующие команды:

# Скачивание последней версии WordPress

```
wget https://wordpress.org/latest.zip
```

# Распаковка архива

```
unzip latest.zip
```

# Перемещение файлов WordPress в директорию веб-сервера

```
sudo rsync -av wordpress/ /var/www/html/
```

# Установка прав доступа пользователя www-data

```
sudo chown -R www-data:www-data /var/www/html/  
  
sudo find /var/www/html/ -type d -exec chmod 755 {} \  
  
sudo find /var/www/html/ -type f -exec chmod 644 {} \  

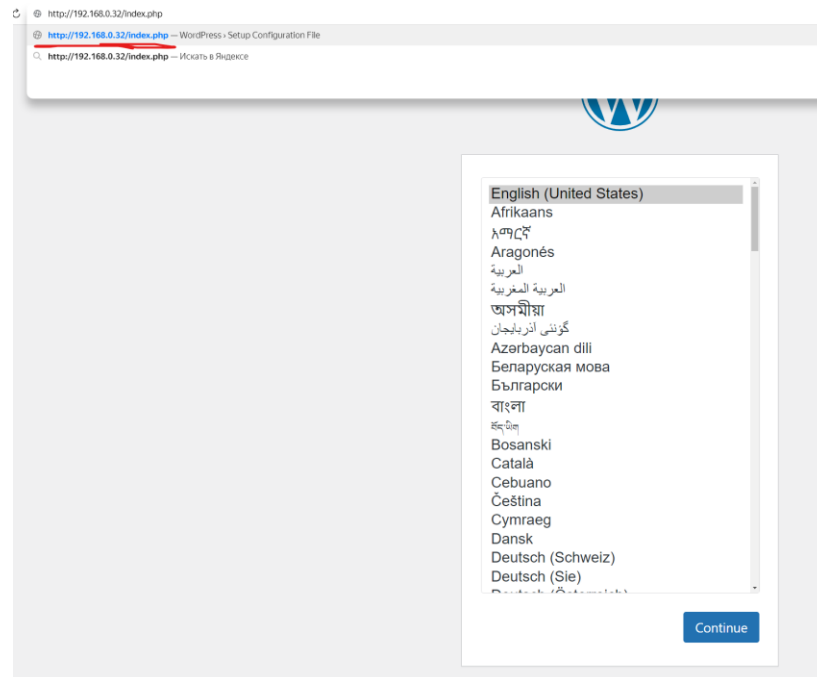
```

При запуске установки CMS в браузере, нам нужно будет указать настройки подключения удаленного сервера СУБД, так как по заданию нам, необходимо, чтобы веб-сервера работали в отказоустойчивом кластере СУБД, которые мы **настроили в первой части и получили публичный ип-адрес с помощью сервиса keepalived.**

Наберите в браузере:


IP-адрес вебсервера1/index.php

Например (если вы наберете просто ип-адрес, то вам выйдет наши индексные index.html):



После настройки отказоустойчивого кластера СУБД (Описано в Модуль Б1), запомните параметры пользователя, имени СУБД, пароля, которые использовали для CMS Wordpress. Напомним, что

после «Сервер базы данных» мы указываем VIP адрес кластера СУБД MYSQL.



Введите здесь информацию о подключении к базе данных. Если вы в ней не уверены, свяжитесь с поддержкой вашего хостинга.

Имя базы данных

cms

Имя базы данных, в которую вы хотите установить WordPress.


Имя пользователя

userwp

Имя пользователя базы данных.

Пароль

.....

 Показать

Пароль пользователя базы данных.

Сервер базы данных

192.168.0.55

Если localhost не работает, нужно узнать правильный адрес в службе поддержки хостинг-провайдера.


Префикс таблиц

wp\_

Если вы хотите запустить несколько копий WordPress в одной базе, измените это значение.

Отправить

192.168.0.32WordPress - Настройка файла конфигурации



Всё в порядке! Вы успешно прошли эту часть установки. WordPress теперь может подключиться к вашей базе данных. Если вы готовы, пришло время...

Запустить установку

## Добро пожаловать

Добро пожаловать в знаменитую пятиминутную установку WordPress! Просто заполните поля — и вперёд, к использованию самой мощной и гибкой персональной платформы для публикаций в мире!

## Требуется информация

Пожалуйста, укажите следующую информацию. Не переживайте, потом вы всегда сможете изменить эти настройки.

Название сайта	<input type="text" value="Харконен"/>
Имя пользователя	<input type="text" value="student"/> <small>Имя пользователя может содержать только латинские буквы, пробелы, подчёркивания, дефисы, точки и символ @.</small>
Пароль	<div><input type="password" value="v\$7j#kJzEQb3&amp;1Rwf!"/><div>Надёжный</div><div>Скрыть</div></div> <p><b>Важно:</b> Этот пароль понадобится вам для входа. Сохраните его в надёжном месте.</p>
Ваш e-mail	<input type="text" value="dsadsa@dadsa.rtg"/> <small>Внимательно проверьте адрес электронной почты, перед тем как продолжить.</small>
Видимость для поисковых систем	<input type="checkbox"/> Попросить поисковые системы не индексировать сайт <small>Будет ли учитываться этот запрос — зависит от поисковых систем.</small>
<input type="button" value="Установить WordPress"/>	

Теперь займемся балансировкой нагрузки наших веб-серверов.

Активируем модули apache2 для настройки проксирования и балансировки нагрузки между двумя веб-серверами:

```
sudo a2enmod proxy
sudo a2enmod proxy_http
sudo a2enmod lbmethod_byrequests
```

Перезагрузить Apache2 для применения настроек:

```
sudo systemctl restart apache2
```

Далее узнаем ип-адреса внешних интерфейсов (публичных) наших ДВУХ веб-серверов:

```
ip a
```

Скорее всего на вашем гипервизоре будут другие названия сетевых интерфейсов, в данном случае наша сетевая карта смотрящая в NAT имеет сетевой интерфейс enp0s3 с ип-адресом 10.0.2.15, у вас же на программном гипервизоре может быть сконфигурировано подключение в режиме моста, чтобы DHCP-сервер головного сетевого устройства присваивал вашим ВМ публичный ип-адрес и давал доступ в сеть Интернет (WAN), который будет виден «из вне» для дальнейшей настройки. enp0s8 это имя второго сетевого интерфейса, которое смотрит в LAN сеть всех ВМ.

## НА ВЕБ-СЕРВЕРАХ В КОНФИГ ФАЙЛАХ АРАСНЕ2 МЫ РАБОТАЕМ С ИП-АДРЕСОМ ВНЕШНЕГО ИНТЕРФЕЙСА (КОТОРЫЙ СМОТРИТ В WAN):

```
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:f4:7a:26 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 metric 100 brd 10.0.2.255 scope global dynamic enp0s3
        valid_lft 81420sec preferred_lft 81420sec
    inet6 fe80::a00:27ff:fef4:7a26/64 scope link
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:c8:b0:d4 brd ff:ff:ff:ff:ff:ff
    inet 192.168.56.101/24 brd 192.168.56.255 scope global enp0s8
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fec8:b0d4/64 scope link
        valid_lft forever preferred_lft forever
```

далее нам необходимо настроить балансировщик нагрузки из стека из наших двух веб-серверов (они же ноды). Создаем новый конфиг файл для нашего первого веб-сервера:

```
sudo nano /etc/apache2/sites-available/balancer.conf
```

В него нужно написать или вставить следующее содержимое:

```
<VirtualHost ИПАДРЕСПЕРВОГОСЕРВЕРА:80>
```

```
<Proxy "balancer://mycluster">
    BalancerMember http://127.0.0.1 route=web1 status=+H
    BalancerMember http://ИПАДРЕСВТОРОГОСЕРВЕРА route=web2 status=+H
    ProxySet stickysession=ROUTEID
</Proxy>
```

```
ProxyPreserveHost On
ProxyPass / balancer://mycluster/ stickysession=ROUTEID
ProxyPassReverse / balancer://mycluster/
```

```
</VirtualHost>
```

В самом первом параметре мы указываем ип-адрес нашего веб-сервера, на котором создан данный конфиг файл. далее разберем одну из строк **BalancerMember <http://127.0.0.1>** – которая указывает машине при первом запросе ссылаться на сетевой интерфейс данной машины. **route=web1** и **ProxySet stickysession=ROUTEID** указывают машине, что необходимо следить за сессиями пользователей и направлять их на те ип-адреса, с которыми у них была открыт обмен, чтобы не терять последующий обмен данными. **status=+H** указывает, что балансировщику нагрузки сначала нужно проверить «жизнеспособность» сервера и если он отвечает, то уже направить на него запрос пользователя. **BalancerMember <http://ИПАДРЕСВТОРОГОСЕРВЕРА> route=web2 status=+H** соответственно переадресует запрос следующего пользователя сессии уже на второй веб-сервер в нашем стеке.

минус такого подхода балансировки заключается в том, что сервер одновременно выполняет и функции балансировщика нагрузки и функции веб-сервера, что дает дополнительную нагрузку и снижает производительность системы. дешево и сердито.

Должно получиться, как пример (на ип-адреса 192\*\*\*\* не обращайте внимание) так:

```
<VirtualHost 192.168.56.101:80>

    <Proxy "balancer://mycluster">
        BalancerMember http://127.0.0.1 route=web1 status=+H
        BalancerMember http://192.168.56.102 route=web2 status=+H
        ProxySet stickysession=ROUTEID
    </Proxy>

    ProxyPreserveHost On
    ProxyPass / balancer://mycluster/ stickysession=ROUTEID
    ProxyPassReverse / balancer://mycluster/
</VirtualHost>
```

Для второго веб-сервера конфигурационный файл должен быть с поменяными местами ип-адресами:

```
sudo nano /etc/apache2/sites-available/balancer.conf
```

```
<VirtualHost ИПАДРЕСвторогоСЕРВЕРА:80>

    <Proxy "balancer://mycluster">
        BalancerMember http://127.0.0.1 route=web1 status=+H
        BalancerMember http://ИПАДРЕСпервогоСЕРВЕРА route=web2 status=+H
        ProxySet stickysession=ROUTEID
    </Proxy>

    ProxyPreserveHost On
    ProxyPass / balancer://mycluster/ stickysession=ROUTEID
    ProxyPassReverse / balancer://mycluster/
</VirtualHost>
```

После чего нам необходимо активировать наш конфигурационный файл:

```
sudo a2ensite balancer.conf
```

и деактивировать дефолтный конфиг файла сервера Apache2:

```
sudo a2dissite 000-default.conf
```

Перезагрузить Apache2 для применения настроек конфигурационного файла:

```
sudo systemctl restart apache2
```

По итогу в каталоге /etc/apache2/sites-enabled/ у нас должен быть файл balancer.conf с симлинком.

```
total 4
-rw-r--r-- 1 root root 1358 Jan 13 12:30 000-default.conf2
lrwxrwxrwx 1 root root   32 Jan 13 14:11 balancer.conf -> ../sites-available/balancer.conf
root@web-server1:/etc/apache2/sites-enabled#
```

Это заключительный шаг в настройке наших балансировщиков нагрузки на базе Apache2.

## II

Мы настроили балансировщик нагрузки и компоненты веб-сервера на базе APACHE2. Теперь необходимо, чтобы файлы наших веб-серверов синхронизировались т.е. при создании или загрузке какого то файла, он появлялся в таких же каталогах и у другого веб-сервера. Для этого мы будем использовать rsync, который работает по SSH протоколу. База данных у нас уже настроена в режиме репликации, по этому каждый из веб-серверов будет писать и читать в «ОДНУ» баз данных, а те работать в свою очередь в режиме репликации и отказоустойчивости.

Для этого у нас должен быть одинаковый пользователь и пароль на всех VM, на веб-серверах в частности (чтобы не запутаться).

Чтобы синхронизация запускалась автоматизировано без ввода пароля и вашего участия, необходимо первично подключиться по SSH веб-серверами друг другу, обменяться «отпечатками», сгенерировать публичный ssh ключ и скопировать с первого веб-сервера на второй. На первом веб-сервере генерируем ключ:

ВНИМАНИЕ!!! Вы генерируете ключ из под того пользователя, которым работаете в терминале. Если вы работаете под root`ом, то вы не сможете дефолтно передать ключ и подключиться по SSH не настроив конфиг файл SSH. Поэтому разогнитесь командой exit в стандартно пользователя с правами вызова sudo, который вы создавали при разворачивании виртуальных машин. Допустим это пользователь **user**. Под пользователем **user** выполняем генерацию ключа:

```
ssh-keygen
```

(на все предложения жамкаем enter)

После чего копируем его на второй веб-сервер:

```
ssh-copy-id имяпользователя@адресВТОРОГОВЕБсервера
```

При сообщении готовы ли вы продолжить подключение напишите обязательно: **yes**

Пример:

```
The key's randomart image is:
+---[RSA 3072]-----+
|
|  . = o  o
|   E S . o . .
|  O B   o = +
| o O @ = . . + o
| . *. @ o * o . .
|  O B o o . . .
+---[SHA256]-----+
user@web-server1:~$ ssh-copy-keygen 192.168.56.102
ssh-copy-keygen: command not found
user@web-server1:~$ ssh-copy-id 192.168.56.102
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/user/.ssh/id_rsa.pub"
The authenticity of host '192.168.56.102 (192.168.56.102)' can't be established.
ED25519 key fingerprint is SHA256:qAaP518AQdUCSN6kh+A5PNTGcLAYf39PvJhfjDUpw1Y.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes|
```

Подтверждаете пароль пользователя, который есть на втором веб-сервере. Все мы обменялись ключами, теперь нет необходимости вводить каждый раз пароль пока сохранены «отпечатки».

Проверим установлен ли у нас rsync (он должен быть установлен на обоих серверах):

```
sudo apt install rsync sshpass
```

Далее создадим BASH-скрипт на **первом сервере**, который будет синхронизировать данные первого веб-сервера со вторым веб-сервером и этот скрипт будет работать в обе стороны, но исполняться только на первом веб-сервере. Ставить их на оба сервера, создавать нагрузку и если один сервер и так «отвалится» толку не будет от исполнения скриптов на двух серверах:

## nano sync.sh

Создаем скрипт, который делает блокировку процесса rsync, если он запущен и выполняется, чтобы планировщиком не было параллельного запуска скрипта со следующим содержанием на **первом веб-сервере**:

```
#!/bin/bash
# генерируем ключ из под sudo с указанием того пользователя из под которого будем
# подключаться из под SSH
WEBSERVER1="192.168.56.101" # ip address webserver 1
WEBSERVER2="192.168.56.108" # ip address webserver 2
SSHUSER="user" # наш пользователь с логином user на подключаемой машине
USERSSHPASS="123456" # пароль подключения к 2 серверу
DESTINATION="/var/www/html/" # указываем куда и откуда
# добавляем исполнение в rsyncs утилиты sshpass чтобы не запрашивал пароль
export RSYNC_RSH="sshpass -p '$USERSSHPASS' ssh"
# при запуске скрипта создается файл и проверяется при следующем запуске крона, на #
# наличие уже запущенного скрипта
touch /tmp/run_once_marker
if [ -f /tmp/run_once_marker ]; then
    exit 0
fi

pid_file="/tmp/rsync.pid"
log_file="/tmp/rsync.log"

while true; do
    if [ ! -f "$pid_file" ]; then
        sleep 5
        # запускаем rsync и сохраняем его PID в файл.
        rsync -avz --update --chown=www-data:www-data $DESTINATION
        $SSHUSER@$WEBSERVER2:$DESTINATION
        rsync -avz --update --chown=www-data:www-data
        $SSHUSER@$WEBSERVER2:$DESTINATION $DESTINATION &
        echo $! > "$pid_file"
    fi

    # Проверяем наличие процесса с указанным PID.
    if pgrep -P $(cat "$pid_file") > /dev/null; then
        # Ожидаем завершения процесса.
        wait $(cat "$pid_file")
    else
        echo "Процесс с PID $(cat "$pid_file") не найден."
        rm "$pid_file"
    fi

    # Удаляем файл PID после завершения процесса.
    rm "$pid_file"
done
```

# Синхронизация с первого сервера на второй, при изменении на серверах сверяем и обновляем (атрибут update)  
# --chown=www-data:www-data - данным атрибутом указываем, что владельцем всех копируемых файлов будет

Назначим права на исполнение (путь файла как пример, в данном случае он находится в домашнем каталоге пользователя, под которым в данный момент работаем):

```
chmod +x ~/sync.sh
```

Теперь необходимо произвести настройку планировщика cron

Добавим задание в cron, чтобы оно выполнялось каждые 5 секунд. Для этого откройте редактор crontab командой (выбрав соответствующий редактор):

```
crontab -e
```

Добавьте следующую строку и сохраните файл:

```
* * * * * /путь_вашего_скрипта/sync.sh
```



Эта строка означает, что например скрипт расположенный в каталоге /home/user/lock.sh будет выполняться каждую минуту, но чтобы Cron не создавал дублированные процессы скрипта синхронизации в скрипте lock.sh создается блокировка файла, которая сообщает планировщику, что файл уже используется.

Получим такое сообщение от Cron'a, если все успешно

```
user@web-server1:~$ chmod +x ~/sync.sh
user@web-server1:~$ crontab -e
no crontab for user - using an empty one
crontab: installing new crontab
user@web-server1:~$
```

#### Примечания

- Убедитесь, что пути к директориям указаны правильно.
  - Проверьте, что rsync установлен на обоих серверах.
  - Если необходимо синхронизировать файлы в обе стороны, используйте опцию --update или добавьте второй вызов rsync в обратном направлении.
- Такой подход обеспечит регулярную синхронизацию файлов между двумя серверами с минимальным временем задержки.

