

Цель работы - установить сервисы и настроить резервное копирование и мониторинг серверов в отказоустойчивой масштабируемой инфраструктуре на основе виртуальных машин.

В модуле Б1 необходимо на виртуальной машине разместить **SQL сервер баз данных для хранения данных мониторинга и данных приложения в модуле В** (если это необходимо приложению). Настроить необходимые конфигурационные файлы, а также создать базу данных и пользователя базы данных.

ВЫПОЛНЕНИЕ:

ЧАСТЬ 1: УСТАНОВКА И КОНФИГУРИРОВАНИЕ ДВУХ СУБД В РЕЖИМЕ РЕПЛИКАЦИИ

Клонировем или создаем 2 виртуальные машины, где установим СУБД. Обе ВМ (ноды) будут работать в режиме **MASTER**. Использование режима MASTER-SLAVE обычно применяется для горизонтального масштабирования, где в одну базу пишутся запросы, а для снижения нагрузки, реплицированные данные «читаются» уже с реплик СУБД, тем самым снижая нагрузку на БД головного сервера. Но, если MASTER «отъедет», то программно ненастроенная работа кластера будет прекращена. Работа кластера в режиме MASTER-MASTER позволяет работать кластеру в «горячем» режиме, и если один сервер «отъедет», то работу на себя возьмет оставшаяся нода в кластере без «перестроения» настроек сервисов или приложений.

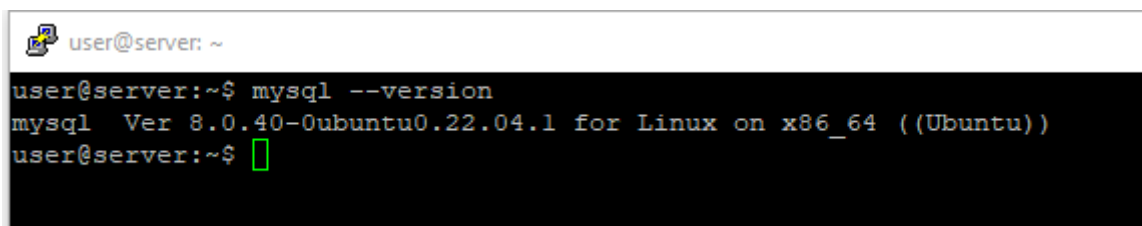
Далее часть действий одинакова для двух ВМ, за исключением одного значения в конфиг файле СУБД mysql и настройки репликации мастеров в консоле СУБД.

Устанавливаем на ВМ **mysql-server 1** и ВМ **mysql-server 2** СУБД MySQL

```
sudo apt update && sudo apt install mysql-server -y
```

Проверяем версию установленного СУБД

```
mysql --version
```



```
user@server: ~  
user@server:~$ mysql --version  
mysql Ver 8.0.40-0ubuntu0.22.04.1 for Linux on x86_64 ((Ubuntu))  
user@server:~$
```

Теперь нам необходимо создать пользователя для репликации на двух серверах.

Так как у нас не созданы управляющие пользователи, то для начала работы нам необходимо перейти в режим суперпользователя:

```
sudo su
```

И теперь войдем в консоль СУБД на каждой из двух ВМ:



Создадим на обоих серверах одинакового пользователя для репликации с именем **replication** и назначим ему пароль **RepliPass!!!** на все базы:

```
CREATE USER 'replication'@'%' IDENTIFIED WITH mysql_native_password BY 'RepliPass!!!';
```

Далее назначим ему все права репликации (*.*) на все БД в нашей СУБД(%):

```
GRANT REPLICATION SLAVE ON *.* TO 'replication'@'%';
```

И применим изменения:

```
FLUSH PRIVILEGES;
```

Выглядит это так (скрин без назначения пароля):

```
user@server:~$ sudo su
root@server:/home/user# mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 11
Server version: 8.0.40-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE USER 'replication'@'%';
Query OK, 0 rows affected (0.02 sec)

mysql> GRANT REPLICATION SLAVE ON *.* TO 'replication'@'%';
Query OK, 0 rows affected (0.01 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.01 sec)

mysql> █
```

Далее нам необходимо создать БД для CMS Wordpress, пользователя и пароль, для подключения к нашим СУБД:

```
CREATE DATABASE cms;
CREATE USER 'userwp'@'%' IDENTIFIED WITH mysql_native_password BY 'passwp';
GRANT ALL PRIVILEGES ON cms.* TO 'userwp'@'%';
FLUSH PRIVILEGES;
```

Запомним имя СУБД (в данном случае это **cms**).

Пользователя (в данном случае это **userwp**) и пароль (в данном случае это **passwp**)

После чего выйдем из консоли СУБД:

exit

Далее нам необходимо отредактировать конфигурационный файл СУБД:

```
nano /etc/mysql/mysql.conf.d/mysqld.cnf
```

Нам необходимо раскомментировать следующие строки в секции [mysqld]:

Это port = 3306

И bind-address сменить с 127.0.0.1 (loopback интерфейс) на прослушивание по всем ип-адресам 0.0.0.0

```
[mysqld]
#
# * Basic Settings
#
user                = mysql
# pid-file           = /var/run/mysqld/mysqld.pid
# socket              = /var/run/mysqld/mysqld.sock
port                = 3306
# datadir             = /var/lib/mysql

# If MySQL is running as a replication slave, this should be
# changed. Ref https://dev.mysql.com/doc/refman/8.0/en/server-syst
# tmpdir              = /tmp
#
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address         = 0.0.0.0
mysqlx-bind-address  = 127.0.0.1
#
```

Далее пролистаем вниз и раскомментируем значения:

ВНИМАНИЕ!!!

server-id = 1 (обратите внимание, что ИД должен быть индивидуальным для каждого сервера, для первого сервера ставим ИД 1, а для второго - 2)

и

log_bin = /var/log/mysql/mysql-bin.log

```
# log-queries-not-using-indexes
#
# The following can be used as easy to replay backup logs or for replication.
# note: if you are setting up a replication slave, see README.Debian about
# other settings you may need to change.
server-id            = 1
log_bin              = /var/log/mysql/mysql-bin.log
# binlog_expire_logs_seconds = 2592000
max_binlog_size      = 100M
# binlog_do_db         = include_database_name
# binlog_ignore_db     = include_database_name
```

Сохраняем файл. После изменений конфиг файла перезагрузим ВМ`ы:

```
systemctl restart mysql
```

Можете проверить статус сервера СУБД командой:

```
systemctl status mysql
```

Войдем снова в консоль СУБД:

```
mysql
```

Далее находясь в консоле СУБД **BM 1** необходимо проверить статус MASTER`а :

```
SHOW MASTER STATUS;
```

```
mysql> SHOW MASTER STATUS;
+-----+-----+-----+-----+-----+
| File           | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
+-----+-----+-----+-----+-----+
| mysql-bin.000001 |      157 |              |                  |                  |
+-----+-----+-----+-----+-----+
1 row in set (0.00 sec)

mysql> 
```

Здесь мы видим начало лога, необходимые для синхронизации реплик. Запоминаем параметры в **File** и **Position**:

На данном скрине приведённым выше это файл mysql-bin.000001

И позиция 157

Эти данные необходимо указывать в следующих запросах в консоле СУБД для настройки репликации.

Теперь войдем в консоль Mysql и применим настройки репликации по типу MASTER-MASTER, т.е. каждый из серверов будет копировать происходящие изменения друг в друга.

Для начала мы останавливаем записи в БД командой stop replica. Далее сменяем мастера на наш сервер репликации, где указываем ип-адрес второй машины, пользователя которого мы создали выше и его пароль, а также лог файл с которого СУБД стоит начать синхронизацию и позицию записи с лог файле, после чего стартуем реплику.

На **первом мастере** указываем мастером для репликации **ип-адрес второго**

```
STOP REPLICATION;
CHANGE MASTER TO MASTER_HOST = '192.168.0.43', master_port=3306, MASTER_USER
= 'replication', MASTER_PASSWORD = 'RepliPass!!!', MASTER_LOG_FILE = 'mysql-
bin.000003', MASTER_LOG_POS = 157;
START REPLICATION;
```

На **втором мастере** указываем **ип-адрес** мастера репликации **первого**

```
STOP REPLICA;  
CHANGE MASTER TO MASTER_HOST = '192.168.0.42', master_port=3306, MASTER_USER  
= 'replication', MASTER_PASSWORD = 'RepliPass!!!', MASTER_LOG_FILE = 'mysql-  
bin.000003', MASTER_LOG_POS = 157;  
START REPLICA;
```

Команда SLAVE STOP в MySQL завершает поток подчинённого сервера. 1

С версии 8.0.22 команда STOP SLAVE устарела, вместо неё следует использовать псевдоним STOP REPLICA. 5

Репликация настроена, теперь проверим статус репликации находясь в консоле mysql:

```
SHOW SLAVE STATUS\G
```

```
mysql> SHOW SLAVE STATUS\G  
***** 1. row *****  
      slave_IO_State: Waiting for source to send event  
      Master_Host: 192.168.56.108  
      Master_User: replication  
      Master_Port: 3306  
      Connect_Retry: 60  
      Master_Log_File: mysql-bin.000001  
      Read_Master_Log_Pos: 1040  
      Relay_Log_File: server-relay-bin.000002  
      Relay_Log_Pos: 615  
      Relay_Master_Log_File: mysql-bin.000001  
      Slave_IO_Running: Yes  
      Slave_SQL_Running: Yes  
      Replicate_Do_DB:  
      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
```

На скрине выше мы видим состояние репликации, статус waiting for source to send event, говорит о том, что сервер ожидает события для отправки данных. Если возникла какая либо ошибка, например статус Connecting, просмотрите позицию Last Error, в нем может быть отображение ошибки. Ошибки могут быть в следствие неправильно назначенного пароля и пользователя субд для репликации, закрытый порт 3306 или неправильный маршрут, закрытие портов файрволлом ufw.

Для траблшуттинга, вы можете со второго ВМ подключится по tenlnet к первому, проверив порт подключения к СУБД:

```
telnet 192.168.56.107 3306
```

```

root@server2:/home/user# telnet 192.168.56.107 3306
Trying 192.168.56.107...
Connected to 192.168.56.107.
Escape character is '^]'.
[
8.0.40-0ubuntu0.22.04.1dq
a
R'xu<);f%<'caching_sha2_password

```

И если порт открыт и маршрут доступен, то произойдёт подключение. В противном случае может быть показана ошибка no route to host, при том, что команда ping видит другой хост, ошибка говорит о том, что порт 3306 не отвечает. Либо он другой, либо не назначен, а скорее всего не прослушивается на внешних интерфейсах (поэтому мы настраивали bind-address 0.0.0.0), либо mysql «упал».

Далее мы проверим, как работает репликация. На первой нодe зайдя в консоль mysql выполните команду:

```
CREATE DATABASE example;
```

После чего на второй нодe кластера проверьте реплицировалась ли она:

```
SHOW DATABASES;
```

<pre> mysql> show databases; +-----+ Database +-----+ example information_schema mysql performance_schema sys +-----+ 5 rows in set (0.00 sec) mysql> show databases; +-----+ Database +-----+ example example2 information_schema mysql performance_schema sys +-----+ 6 rows in set (0.00 sec) mysql> </pre>	<pre> 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: Master_public_key_path: Get_master_public_key: 0 Network_Namespace: 1 row in set, 1 warning (0.00 sec) mysql> create table example -> ^C mysql> create table example; ERROR 1046 (3D000): No database selected mysql> create database example; Query OK, 1 row affected (0.02 sec) mysql> create database example2; Query OK, 1 row affected (0.01 sec) mysql> </pre>
---	---

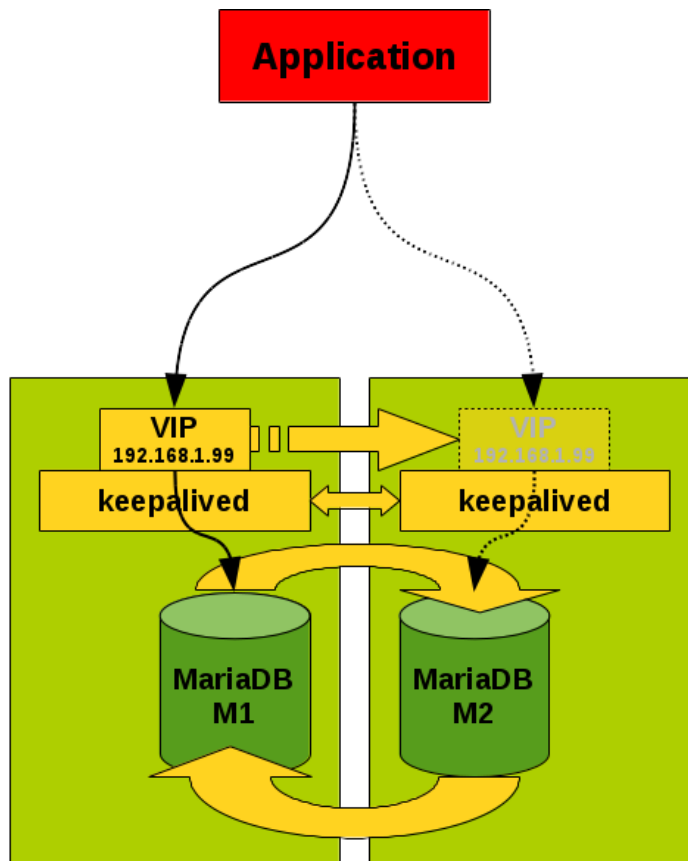
Слева на скрине мы создаем базы данных в СУБД, а в правой части, на второй нодe, видим как появляются реплицированные БД.

Настройка репликации в режиме MASTER-MASTER закончена.

ЧАСТЬ 2. НАСТРОЙКА ВИРТУАЛЬНОГО ИП-АДРЕСА СЕРВЕРОВ СУБД

Выше, мы настроили СУБД в режиме MASTER-репликации и если один сервер СУБД «упадет», то второй будет работать с такими же данными, как и первая. Нам необходимо, чтобы наши веб-сервера «писали и читали» обращаясь к «одному рабочему» СУБД, а те в свою очередь уже реплицировались. Для этого мы будем также использовать keepalived, как в случае и с балансировщиками нагрузки, но уже с использованием BASH скрипта в конфиг файле keepalived для проверки статуса СУБД и переключения на статус MASTER, где СУБД находится в рабочем состоянии. В кластере у нас будет 1 VM СУБД в режиме MASTER, другой в режиме BACKUP.

Наша модель сети выглядит так:



Более подробно можно почитать по ссылке:

<https://www.fromdual.com/mariadb-master-master-gtid-based-replication-with-keepalived-vip>

А) На двух серверах СУБД устанавливаем keepalived и переходим в рабочий каталог:

```
sudo apt update  
sudo apt install keepalived -y  
cd /etc/keepalived/
```

Где с правами суперпользователя создаем файл keepalived.conf:

```
sudo nano /etc/keepalived/keepalived.conf
```

Содержание конфиг файла /etc/keepalived/keepalived.conf **СУБД ВМ 1 (MASTER)**:

```
# primary mysql server
global_defs {
    router_id msq1-01
}
# Health checks
vrrp_script chk_mysql {
    script "/etc/keepalived/checkactive.sh"
#    weight 2      # Is relevant for the diff in priority
    interval 1     # every ... seconds
    timeout 3      # script considered failed after ... seconds
    fall 3         # number of failures for K.O.
    rise 1         # number of success for OK
}
vrrp_instance VI-MM-VIP {
    state MASTER
    interface enp0s3
    virtual_router_id 123
    priority 100
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass pwd123
    }
    virtual_ipaddress {
        192.168.0.55/24 dev enp0s3 # публичный ип-адрес VIP
    }
}
track_script {
    chk_mysql
}
}
```

Обратите внимание на листинг выше, что в части vrrp_script chk_mysql у нас указан уже путь до скрипта "/etc/keepalived/checkactive.sh", который мы создадим позже.

Содержание конфиг файла /etc/keepalived/keepalived.conf на **СУБД ВМ 2 (BACKUP)**:

```
# secondary mysql server
global_defs {
    router_id msq1-02
}
# Health checks
vrrp_script chk_mysql {
#    script "/usr/bin/systemctl is-active --quiet mysqld"
    script "/etc/keepalived/checkactive.sh"
#    weight 2      # Is relevant for the diff in priority
    interval 1     # every ... seconds
    timeout 3      # script considered failed after ... seconds
    fall 3         # number of failures for K.O.
    rise 2         # number of success for OK
}
vrrp_instance VI-MM-VIP {
    state BACKUP
    interface enp0s3
    virtual_router_id 123
    priority 90
    advert_int 1
    authentication {
        auth_type PASS
        auth_pass pwd123
    }
    virtual_ipaddress {
        192.168.0.55/24 dev enp0s3 # публичный ип-адрес VIP
    }
}
track_script {
    chk_mysql
}
}
```


Директива `track_script` запускает скрипт с параметрами, определенными в блоке `vrrp_script`, который имеет следующий формат:

```
vrrp_script <название> {
script <"путь к исполняемому файлу">
interval <число, секунд> - периодичность запуска скрипта, по умолчанию 1 секунда
fall <число> - количество раз, которое скрипт вернул не нулевое значение, при котором перейти в состояние FAULT
rise <число> - количество раз, которое скрипт вернул нулевое значение, при котором выйти из состояния FAULT
timeout <число> - время ожидания, пока скрипт вернет результат, после которого вернуть ненулевое значение.
weight <число> - значение, на которое будет уменьшен приоритет сервера, в случае перехода в состояние FAULT.
По умолчанию 0, что означает, что сервер перейдет в состояние FAULT, после неудачного выполнения скрипта за количество раз, определенное параметром fall.
}
```

Б) На обоих серверах создаем одинаковый скрипт, который будет проверять статус службы СУБД.

```
sudo nano /etc/keepalived/checkactive.sh
```

Содержание скрипта `checkactive.sh` проверки статуса службы `mysql` расположенному по пути `/etc/keepalived/checkactive.sh`:

```
#!/bin/bash
# Название службы MySQL, обычно это mysql или mysqld
# Проверка статуса службы
STATUS=$(systemctl is-active mysql)
if [ "$STATUS" == "inactive" ];
then
# Возвращаем код выхода 1, если служба неактивна
exit 1
else
# Возвращаем код выхода 0 (нет ошибки), если служба активна
echo 0
fi
```

Сохраняем файл и даем права на исполнение скрипта:

```
sudo chmod +x /etc/keepalived/checkactive.sh
```

После чего в режиме суперпользователя выполняем команды создания пользователя `keepalived_script`, далее вносим поток с команды `echo` в файл `sysctl.conf` (в режиме обычного пользователя команда `echo` не выполнится), применяем настройки и запускаем службу `keepalived`:

```
sudo useradd -s /usr/sbin/nologin keepalived_script
echo "net.ipv4.ip_nonlocal_bind=1" >> /etc/sysctl.conf
sudo sysctl -p
sudo systemctl start keepalived
```

После чего проверим статус `keepalived` мастера на **СУБД ВМ 1**

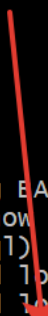
```
systemctl status keepalived
```

```

root@server:/etc/keepalived# systemctl status keepalived
keepalived.service - Keepalive Daemon (LVS and VRRP)
   Loaded: loaded (/lib/systemd/system/keepalived.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2025-01-08 13:52:08 UTC; 4min 22s ago
     Main PID: 6635 (keepalived)
        Tasks: 2 (limit: 3403)
       Memory: 1.9M
          CPU: 8.528s
      CGroup: /system.slice/keepalived.service
              └─6635 /usr/sbin/keepalived --dont-fork
                 └─6636 /usr/sbin/keepalived --dont-fork

Jan 08 13:52:08 server Keepalived[6635]: Startup complete
Jan 08 13:52:08 server Keepalived_vrrp[6636]: (VI-MM-VIP) Entering BACKUP STATE (
Jan 08 13:52:08 server Keepalived_vrrp[6636]: Script `chk_mysql` now returning 2
Jan 08 13:52:08 server Keepalived_vrrp[6636]: VRRP_Script(chk_mysql) failed (exit
Jan 08 13:52:08 server Keepalived_vrrp[6636]: (VI-MM-VIP) received lower priority
Jan 08 13:52:09 server Keepalived_vrrp[6636]: (VI-MM-VIP) received lower priority
Jan 08 13:52:10 server Keepalived_vrrp[6636]: (VI-MM-VIP) received lower priority
Jan 08 13:52:11 server Keepalived_vrrp[6636]: (VI-MM-VIP) Entering MASTER STATE
Jan 08 13:52:11 server Keepalived_vrrp[6636]: A thread timer expired 1 135034 s

```

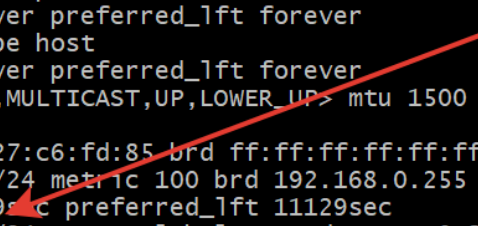


Также видим, что наш сетевой интерфейс, в данном случае enp0s3 получил дополнительный виртуальный ип-адрес:

```

user@server:~$ ip a
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:c6:fd:85 brd ff:ff:ff:ff:ff:ff
   inet 192.168.0.44/24 metric 100 brd 192.168.0.255 scope global dynamic enp0s3
       valid_lft 11129sec preferred_lft 11129sec
   inet 192.168.0.55/24 scope global secondary enp0s3
       valid_lft forever preferred_lft forever
   inet6 fe80::a00:27ff:fec6:fd85/64 scope link
       valid_lft forever preferred_lft forever

```



Проверим статус keepalived мастера на **СУБД ВМ 2**, он должен быть в статусе BACKUP:

```

systemctl status keepalived
● keepalived.service - Keepalive Daemon (LVS and VRRP)
   Loaded: loaded (/lib/systemd/system/keepalived.service; enabled; vendor preset: en
   Active: active (running) since Wed 2025-01-08 14:30:06 UTC; 12min ago
     Main PID: 6518 (keepalived)
        Tasks: 2 (limit: 3403)
       Memory: 1.9M
          CPU: 6.672s
      CGroup: /system.slice/keepalived.service
              └─6518 /usr/sbin/keepalived --dont-fork
                 └─6519 /usr/sbin/keepalived --dont-fork

Jan 08 14:39:26 server Keepalived_vrrp[6519]: (VI-MM-VIP) Entering FAULT STATE
Jan 08 14:40:35 server Keepalived_vrrp[6519]: Script 'chk_mysql' now returning 0
Jan 08 14:40:36 server Keepalived_vrrp[6519]: VRRP_Script(chk_mysql) succeeded
Jan 08 14:40:36 server Keepalived_vrrp[6519]: (VI-MM-VIP) Entering BACKUP STATE
Jan 08 14:41:07 server Keepalived_vrrp[6519]: (VI-MM-VIP) Entering MASTER STATE
Jan 08 14:41:39 server Keepalived_vrrp[6519]: (VI-MM-VIP) Master received advert fr
Jan 08 14:41:39 server Keepalived_vrrp[6519]: (VI-MM-VIP) Entering BACKUP STATE
Jan 08 14:41:44 server Keepalived_vrrp[6519]: (VI-MM-VIP) Entering MASTER STATE
Jan 08 14:41:44 server Keepalived_vrrp[6519]: (VI-MM-VIP) Master received advert fr
Jan 08 14:41:44 server Keepalived_vrrp[6519]: (VI-MM-VIP) Entering BACKUP STATE

```

Теперь проверим на отказоустойчивость:

На **СУБД ВМ 1** останавливаем службу СУБД:

```
systemctl stop mysql
```

Мы видим, что проверка скрипта вызвала ошибку со статусом 1– FAULT STATE и пошло переназначение виртуального ip-адреса на ВМ СУБД 2.

```

root@server:/home/user# systemctl status keepalived
● keepalived.service - Keepalive Daemon (LVS and VRRP)
   Loaded: loaded (/lib/systemd/system/keepalived.service; enabled; vendor preset: en
   Active: active (running) since Wed 2025-01-08 14:36:18 UTC; 1h 0min ago
     Main PID: 681 (keepalived)
        Tasks: 2 (limit: 3403)
       Memory: 5.1M
          CPU: 3min 6.203s
      CGroup: /system.slice/keepalived.service
              └─681 /usr/sbin/keepalived --dont-fork
                 └─719 /usr/sbin/keepalived --dont-fork

Jan 08 15:26:04 server Keepalived_vrrp[719]: A thread timer expired 2.650407 seconds a
Jan 08 15:26:29 server Keepalived_vrrp[719]: (VI-MM-VIP) Received advert from 192.168.
Jan 08 15:26:36 server Keepalived_vrrp[719]: A thread timer expired 5.705065 seconds a
Jan 08 15:26:44 server Keepalived_vrrp[719]: A thread timer expired 1.566876 seconds a
Jan 08 15:31:46 server Keepalived_vrrp[719]: A thread timer expired 1.001083 seconds a
Jan 08 15:35:29 server Keepalived_vrrp[719]: A thread timer expired 1.039423 seconds a
Jan 08 15:35:32 server Keepalived_vrrp[719]: A thread timer expired 1.275365 seconds a
Jan 08 15:37:01 server Keepalived_vrrp[719]: Script 'chk_mysql' now returning 1
Jan 08 15:37:03 server Keepalived_vrrp[719]: VRRP_Script(chk_mysql) failed (exited wit
Jan 08 15:37:03 server Keepalived_vrrp[719]: (VI-MM-VIP) Entering FAULT STATE

```

Проверим статус keepalived на ВМ 2, мы увидим, что она приняла статус MASTER:

```

root@server:/etc/keepalived# systemctl status keepalived
keepalived.service - Keepalive Daemon (LVS and VRRP)
  Loaded: loaded (/lib/systemd/system/keepalived.service; enabled; vendor preset: enabled)
  Active: active (running) since Wed 2025-01-08 14:30:06 UTC; 1h 8min ago
    Main PID: 6518 (keepalived)
      Tasks: 2 (limit: 3403)
     Memory: 1.9M
        CPU: 25.592s
    CGroup: /system.slice/keepalived.service
            └─6518 /usr/sbin/keepalived --dont-fork
              └─6519 /usr/sbin/keepalived --dont-fork

Jan 08 15:26:33 server keepalived_vrrp[6519]: (VI-MM-VIP) Entering MASTER STATE
Jan 08 15:26:33 server keepalived_vrrp[6519]: (VI-MM-VIP) Master received advert fr
Jan 08 15:26:33 server keepalived_vrrp[6519]: (VI-MM-VIP) Entering BACKUP STATE
Jan 08 15:34:00 server keepalived_vrrp[6519]: Script `chk_mysql` now returning 2
Jan 08 15:34:02 server keepalived_vrrp[6519]: VRRP_Script(chk_mysql) failed (exited
Jan 08 15:34:02 server keepalived_vrrp[6519]: (VI-MM-VIP) Entering FAULT STATE
Jan 08 15:36:44 server keepalived_vrrp[6519]: Script `chk_mysql` now returning 0
Jan 08 15:36:45 server keepalived_vrrp[6519]: VRRP_Script(chk_mysql) succeeded
Jan 08 15:36:45 server keepalived_vrrp[6519]: (VI-MM-VIP) Entering BACKUP STATE
Jan 08 15:37:04 server keepalived_vrrp[6519]: (VI-MM-VIP) Entering MASTER STATE

```

А также приняла виртуальный ip-адрес на сетевой интерфейс.

```

root@server:/etc/keepalived# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    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
    link/ether 08:00:27:9b:4e:06 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.43/24 metric 100 brd 192.168.0.255 scope global dynamic
        valid_lft 10982sec preferred_lft 10982sec
    inet 192.168.0.55/24 scope global secondary enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe9b:4e06/64 scope link
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state DOWN
    link/ether 08:00:27:9b:4e:06 brd ff:ff:ff:ff:ff:ff
    inet 192.168.0.43/24 metric 100 brd 192.168.0.255 scope global dynamic
        valid_lft 10982sec preferred_lft 10982sec
    inet 192.168.0.55/24 scope global secondary enp0s3
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe9b:4e06/64 scope link
        valid_lft forever preferred_lft forever

```

И если мы на ВМ СУБД 1 запустим вновь службу mysql, то спустя несколько секунд она должна вернуть себе статус мастер, из-за приоритета в конфиг файле (100):

```
systemctl start mysql
```

И здесь мы видим, что скрипт вернул статус 0, что означает работоспособность MYSQL сервера и ВМ СУБД 1 вернула себе статус MASTER:

```
root@server1:/home/user# systemctl status keepalived
● keepalived.service - Keepalive Daemon (LVS and VRRP)
   Loaded: loaded (/lib/systemd/system/keepalived.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2025-01-08 14:36:18 UTC; 1h 6min ago
     Main PID: 681 (keepalived)
        Tasks: 2 (limit: 3403)
       Memory: 5.1M
          CPU: 3min 29.429s
      CGroup: /system.slice/keepalived.service
              └─681 /usr/sbin/keepalived --dont-fork
                └─719 /usr/sbin/keepalived --dont-fork

Jan 08 15:38:32 server Keepalived_vrrp[719]: A thread timer expired 2.159358 seconds ago
Jan 08 15:39:28 server Keepalived_vrrp[719]: A thread timer expired 1.366233 seconds ago
Jan 08 15:41:28 server Keepalived_vrrp[719]: A thread timer expired 3.024626 seconds ago
Jan 08 15:41:28 server Keepalived_vrrp[719]: Script 'chk_mysql' now returning 0
Jan 08 15:41:28 server Keepalived_vrrp[719]: VRRP_Script(chk_mysql) succeeded
Jan 08 15:41:28 server Keepalived_vrrp[719]: (VI-MM-VIP) Entering BACKUP STATE
Jan 08 15:41:29 server Keepalived_vrrp[719]: (VI-MM-VIP) received lower priority (90) advert
Jan 08 15:41:30 server Keepalived_vrrp[719]: (VI-MM-VIP) received lower priority (90) advert
Jan 08 15:41:31 server Keepalived_vrrp[719]: (VI-MM-VIP) received lower priority (90) advert
Jan 08 15:41:32 server Keepalived_vrrp[719]: (VI-MM-VIP) Entering MASTER STATE
```

НАСТРОЙКА ОТКАЗОУСТОЙЧИВОГО КЛАСТЕРА ИЗ ДВУХ СУБД С РЕПЛИКАЦИЕЙ ДАННЫХ В РЕЖИМЕ MASTER- MASTER И «ОДНИМ» ВИРТУАЛЬНЫМ.

КОНЕЦ ЧАСТИ VM СУБД