

# Устанавливаем и настраиваем базу Redis на Ubuntu 20

19 июля 2021

```
700 appendonly yes
701
702 # The name of the append only file (default: "appendonly.aof")
703
704 appendfilename "appendonly.aof"
705
706 # The fsync() call tells the Operating System to actually write data on disk
707 # instead of waiting for more data in the output buffer. Some OS will really f
708 # data on disk, some other OS will just try to do it ASAP.
709 #
710 # Redis supports three different modes:
711 #
712 # no: don't fsync, just let the OS flush the data when it wants. Faster.
713 # always: fsync after every write to the append only log. Slow. Safest.
```

Redis - это NoSQL база данных, которая хранит данные в виде ключ-значение. Кроме удобного и понятного формата хранения есть еще одно преимущество - это быстрая база данных поддержка которой реализована на множестве языков. В этой статье рассмотрим как установить такую базу на Ubuntu создавать и восстанавливать бэкапы.

## Навигация по посту

- [Для чего нужен Redis](#)
- [Установка из репозитория](#)
  - [Проверка работы сервиса](#)
- [Установка последней версии с компиляцией](#)
  - [Создание пользователя и выдача прав](#)
  - [Создание сервиса](#)
- [Настройка удаленного подключения](#)
  - [Аутентификация и создание пароля](#)
- [Бэкап и восстановление](#)
  - [Настройка RDB](#)
  - [Настройка AOF](#)
  - [Какую модель бэкапа использовать](#)

## Для чего нужен Redis

В любом языке программирования существуют массивы имеющие следующий вид:

```
{'key': {'key2': 'value', 'key3': ['value', 'value3']}}
```

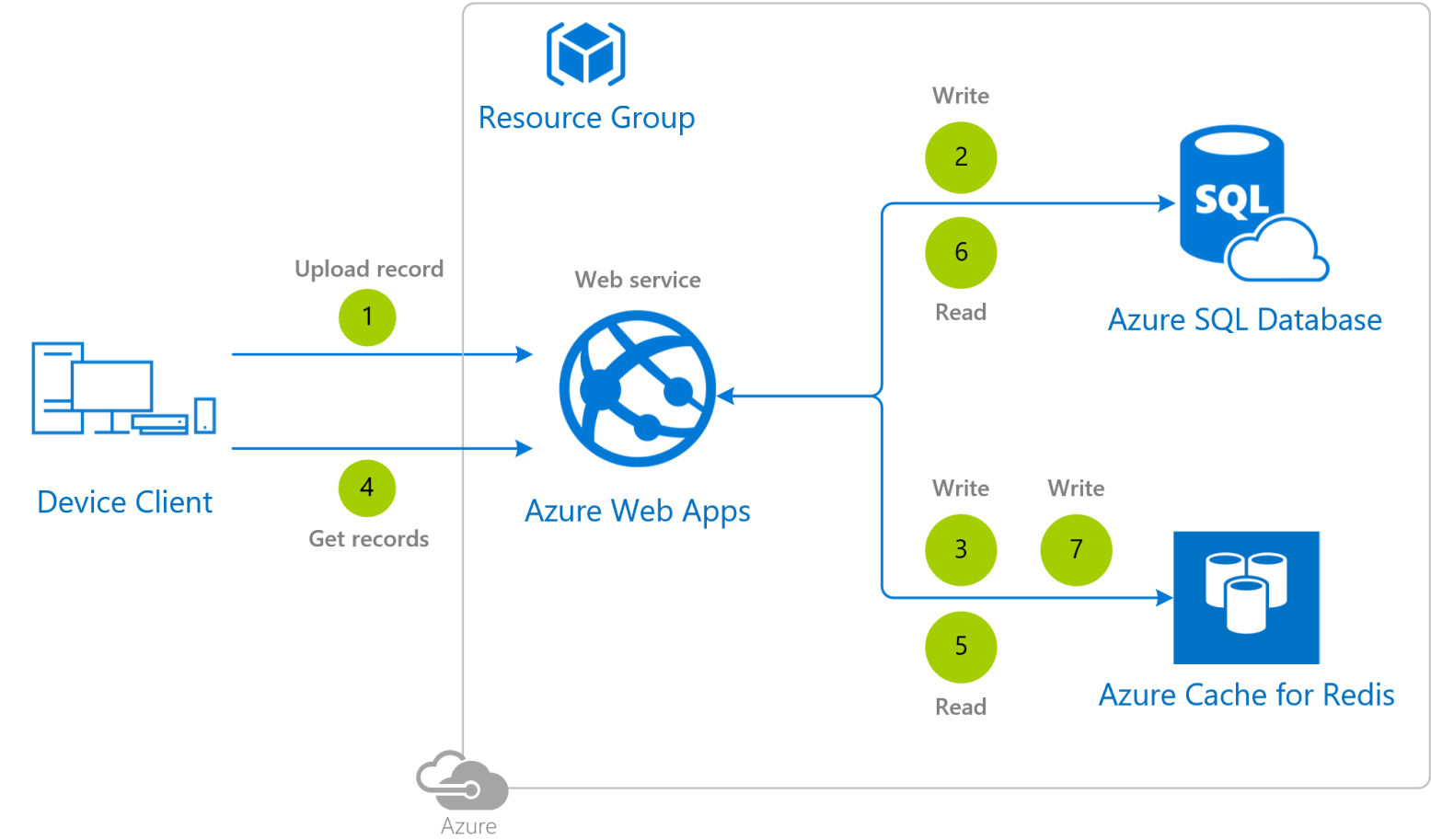
Подобный вид данных сложно сохранить в SQL базы. Что бы это сделать нам нужно будет создать несколько таблиц (по принципу нормализации), а затем выстраивать связь. Можно так же сохранить эти данные как JSON, но это не лучший способ для работы с SQL.

Структура данных, которую мы храним Redis, очень похожа на тот тип, что представлен выше. Т.е. разработчик интуитивно понимает как работать с базой так как она похожа на уже знакомый тип данных.

Кроме этого, Redis может хранить данные ограниченное количество времени, а затем автоматически удалять. Для примера - пользователи часто запрашивают историю покупок. Так как это операция может быть трудоемкой для SQL - ее можно выполнить 1 раз, а результат сохранить как временный в Redis. Пользователь, повторно просматривая историю, будет запрашивать данные из Redis, а не из SQL. Этот процесс может быть и обратным, т.е. сначала данные заносятся в Redis, а потом в SQL.

В основном данные хранятся в памяти сервера, но могут быть выгружены на диск. Если это какие-то важные данные, например сессии пользователей, то можно настроить репликацию и создать кластер.

Схема, которая показывает одну из реализаций при работе с пользователями:



## Установка из репозитория

Перед установкой Redis нужно обновить список доступных пакетов в репозитории Ubuntu:

```
sudo apt update
```

Сам пакет находится в стандартном репозитории Ubuntu, и мы можем его установить оттуда. Отмечу, что чаще всего это не самая последняя версия:

```
sudo apt install redis-server
```

Перед запуском сервиса нам нужно изменить значение параметра 'supervised'. Этот параметр указывает на систему обслуживания сервисов. В современных версиях Ubuntu, а так же CentOS, все сервисы обслуживаются 'systemd'. Мы должны открыть следующий файл:

```
sudo vim /etc/redis/redis.conf
```

И изменить следующее значение:

```
# было supervised no
supervised systemd
```

```
138 # If you run Redis from upstart or systemd, Redis can interact with your
139 # supervision tree. Options:
140 #   supervised no      - no supervision interaction
141 #   supervised upstart - signal upstart by putting Redis into SIGSTOP mode
142 #   supervised systemd - signal systemd by writing READY=1 to $NOTIFY_SOCKET
143 #   supervised auto    - detect upstart or systemd method based on
144 #                       UPSTART_JOB or NOTIFY_SOCKET environment variables
145 # Note: these supervision methods only signal "process is ready."
146 #       They do not enable continuous liveness pings back to your supervisor.
147 # было supervised no
148 supervised systemd
149
150 # If a pid file is specified, Redis writes it where specified at startup
151 # and removes it at exit.
```

В этом же файле вы можете изменить место хранения данных. Значение по умолчанию '/var/lib/redis':

```
258 # The DB will be written inside this directory, with the filename specified
259 # above using the 'dbfilename' configuration directive.
260 #
261 # The Append Only File will also be created inside this directory.
262 #
263 # Note that you must specify a directory here, not a file name.
264 dir /var/lib/redis
265
266 ##### REPLICATION #####
```

Далее включим сервер:

```
sudo systemctl enable --now redis-server
```

## Проверка работы сервиса

Проверим, что запуск сервиса был успешен:

```
sudo systemctl status redis-server
```

```
alex@alex:~$ systemctl status redis.service
● redis-server.service - Advanced key-value store
   Loaded: loaded (/lib/systemd/system/redis-server.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2021-07-18 01:55:28 UTC; 2min 11s ago
     Docs: http://redis.io/documentation,
           man:redis-server(1)
  Main PID: 2897 (redis-server)
    Tasks: 4 (limit: 2175)
   Memory: 1.8M
   CGroup: /system.slice/redis-server.service
           └─2897 /usr/bin/redis-server 127.0.0.1:6379

Jul 18 01:55:28 alex systemd[1]: Starting Advanced key-value store:
```

Мы так же можем проверить работу сервера через внутренние команды. Для этого нужно использовать redis-cli:

```
redis-cli

ping
```

```
alex@ubuntu:~$ redis-cli
127.0.0.1:6379> ping
PONG
```

Для выхода из консоли redis выполните:

```
exit
```

## Установка последней версии с компиляцией

На момент написания статьи, в репозитории Ubuntu, доступна 5-ая версия Redis, а на официальном сайте проекта пишется о вышедшей 6+ версии. Мы можем скачать последнюю версию с сайта проекта, затем скомпилировать и установить ее.

Минус такого подхода в том, что программы типа apt не будут знать об этой установке и зависимых программах. С другой стороны, как указано на официальном сайте, единственные зависимости - это компилятор gcc и libc.

Что бы установить последнюю версию, первое что нужно сделать, это скачать и распаковать архив. По ссылке ниже всегда находится последняя версия программы:

```
wget https://download.redis.io/redis-stable.tar.gz
```

Скорее всего вам понадобится ряд пакетов, которые могут выполнять компиляцию программы (они могут быть уже установлены):

```
sudo apt install build-essential tcl pkg-config
```



```
alex@alex:~$ wget https://download.redis.io/redis-stable.tar.gz
--2021-07-17 18:03:26-- https://download.redis.io/redis-stable.tar.gz
Resolving download.redis.io (download.redis.io)... 45.60.129.1
Connecting to download.redis.io (download.redis.io)|45.60.129.1|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2510889 (2.4M) [application/octet-stream]
Saving to: 'redis-stable.tar.gz'

redis-stable.tar.gz      100%[=====>]      2.39M  10.1MB/s

2021-07-17 18:03:27 (10.1 MB/s) - 'redis-stable.tar.gz' saved [2510889/2510889]

alex@alex:~$ ls
redis-stable.tar.gz
```

Далее распакуем архив и скомпилируем ПО из его исходников:

```
tar xzf redis-stable.tar.gz
cd redis-stable
make MALLOC=libc
```

После компиляции, опционально, можно запустить проверку, которая будет длиться в 3-4 раза дольше предыдущего шага:

```
make test
```

```
119 seconds - unit/hyperloglog
196 seconds - north
226 seconds - unit/maxmemory
330 seconds - integration/replication
0 seconds - defrag

\o/ All tests passed without errors!

Cleanup: may take some time... OK
make[1]: Leaving directory '/home/alex/redis-stable/src'
```

Выполним установку Redis из скомпилированных исходников:

```
sudo make install
```

```
alex@alex:~/redis-stable$ sudo make install
cd src && make install
make[1]: Entering directory '/home/alex/redis-stable/src'

Hint: It's a good idea to run 'make test' ;)

INSTALL redis-server
INSTALL redis-benchmark
INSTALL redis-cli
make[1]: Leaving directory '/home/alex/redis-stable/src'
```

Создадим директорию в которой будем хранить файл конфигурации. Файл конфигурации с настройками по умолчанию находится в текущей папке - его мы должны скопировать:

```
sudo mkdir /etc/redis
sudo cp redis.conf /etc/redis
```

В файле конфигурации, в параметре 'supervised' мы так же должны указать систему инициализации. В ubuntu это systemd:

```
sudo vim /etc/redis/redis.conf
```

```
269 # Note: these supervision methods only signal "process is ready."
270 #       They do not enable continuous pings back to your supervisor.
271 #
272 # The default is "no". To run under upstart/systemd, you can simply uncomment
273 # the line below:
274 #
275 supervised systemd
276
277 # If a pid file is specified, Redis writes it where specified at startup
278 # and removes it at exit.
279 #
```

Не закрывая этот файл так же нужно указать директорию, в которой будет храниться база Redis. Этой директорией, обычно, является '/var/lib/redis/', а устанавливается она в параметре 'dir':

```
446 # The working directory.
447 #
448 # The DB will be written inside this directory, with the filename specified
449 # above using the 'dbfilename' configuration directive.
450 #
451 # The Append Only File will also be created inside this directory.
452 #
453 # Note that you must specify a directory here, not a file name.
454 dir /var/lib/redis/
455
456 ##### REPLICATION #####
457
```

## Создание пользователя и выдача прав

Доступ к папке, в которой будет храниться база, должны иметь только пользователи root и redis. Системного пользователя redis нам нужно создать самостоятельно. Это можно сделать через следующие команды:

```
sudo adduser --system --group --no-create-home redis
```

```
alex@alex:~/redis-stable$ sudo adduser --system --group --no-create-home redis
Adding system user `redis' (UID 112) ...
Adding new group `redis' (GID 117) ...
Adding new user `redis' (UID 112) with group `redis' ...
Not creating home directory `/home/redis'.
alex@alex:~/redis-stable$
```

Теперь создадим папку, в которой будет храниться база. Выдадим нужные права и сменим владельца:

```
sudo mkdir /var/lib/redis
sudo chown redis:redis /var/lib/redis
sudo chmod 770 /var/lib/redis
```

```
alex@alex:~/redis-stable$ sudo chown redis:redis /var/lib/redis
alex@alex:~/redis-stable$ sudo chmod 770 /var/lib/redis
alex@alex:~/redis-stable$ ls -la /var/lib/redis
ls: cannot open directory '/var/lib/redis': Permission denied
alex@alex:~/redis-stable$ sudo ls -la /var/lib/redis
total 8
drwxrwx---  2 redis redis 4096 Jul 17 18:50 .
drwxr-xr-x 39 root  root  4096 Jul 17 18:50 ..
alex@alex:~/redis-stable$
```

## Создание сервиса

Последним этапом будет создание файла-сервиса, который сможет читать 'systemd'. Этот файл мы должны создать по следующему пути:

```
sudo vim /etc/systemd/system/redis.service
```

В этот файл поместим следующее содержимое:

```
[Unit]
Description=Redis In-Memory Data Store
After=network.target

[Service]
User=redis
Group=redis
ExecStart=/usr/local/bin/redis-server /etc/redis/redis.conf
ExecStop=/usr/local/bin/redis-cli shutdown
Restart=always

[Install]
WantedBy=multi-user.target
```

Запустим сервис и проверим его работу:

```
sudo systemctl enable --now redis.service
sudo systemctl restart redis.service
sudo systemctl status redis.service
```

```
alex@alex:~/redis-stable$ sudo systemctl status redis.service
● redis.service - Redis In-Memory Data Store
   Loaded: loaded (/etc/systemd/system/redis.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2021-07-18 02:15:04 UTC; 1s ago
     Main PID: 38626 (redis-server)
        Tasks: 4 (limit: 2175)
       Memory: 2.1M
      CGroup: /system.slice/redis.service
              └─38626 /usr/local/bin/redis-server 127.0.0.1:6379

Jul 18 02:15:04 alex redis-server[38626]: 38626:C 18 Jul 2021 02:15:04.657 # systemd supervision request received
Jul 18 02:15:04 alex redis-server[38626]: 38626:C 18 Jul 2021 02:15:04.658 # oOoOoOoOoOoOo Redis is starting now
Jul 18 02:15:04 alex redis-server[38626]: 38626:C 18 Jul 2021 02:15:04.658 # Redis version=6.2.4, bits=64
Jul 18 02:15:04 alex redis-server[38626]: 38626:C 18 Jul 2021 02:15:04.658 # Configuration loaded
Jul 18 02:15:04 alex redis-server[38626]: 38626:M 18 Jul 2021 02:15:04.658 * Increased maximum number of open files to 1024
```

Запустим cli и выполним некоторые проверки там:

```
redis-cli

ping

set test1 test2
```

```
alex@alex:~/redis-stable$ redis-cli
127.0.0.1:6379> ping
PONG
127.0.0.1:6379> set 1 2
OK
127.0.0.1:6379> get 1
"2"
```

Если ваш вывод аналогичен данным со скриншота, то установка прошла успешно.

Вам так же будет интересно

[Устанавливаем и настраиваем базу Redis на Ubuntu 20](#)

## Настройка удаленного подключения

Для настройки удаленного подключения нужно изменить 2 параметра. Эти параметры связаны с интерфейсом на который может прийти подобный запрос и паролем.

IP адрес интерфейса, с которого может прийти запрос, устанавливается в параметре 'bind'. По умолчанию в этом параметре установлено значение '127.0.0.1', что дает возможность подключаться только локально.

К существующему параметру мы должны добавить IP адрес интерфейса, который будет принимать подключения к Redis. Узнать адрес интерфейса мы можем следующим способом:

```
ip a
```

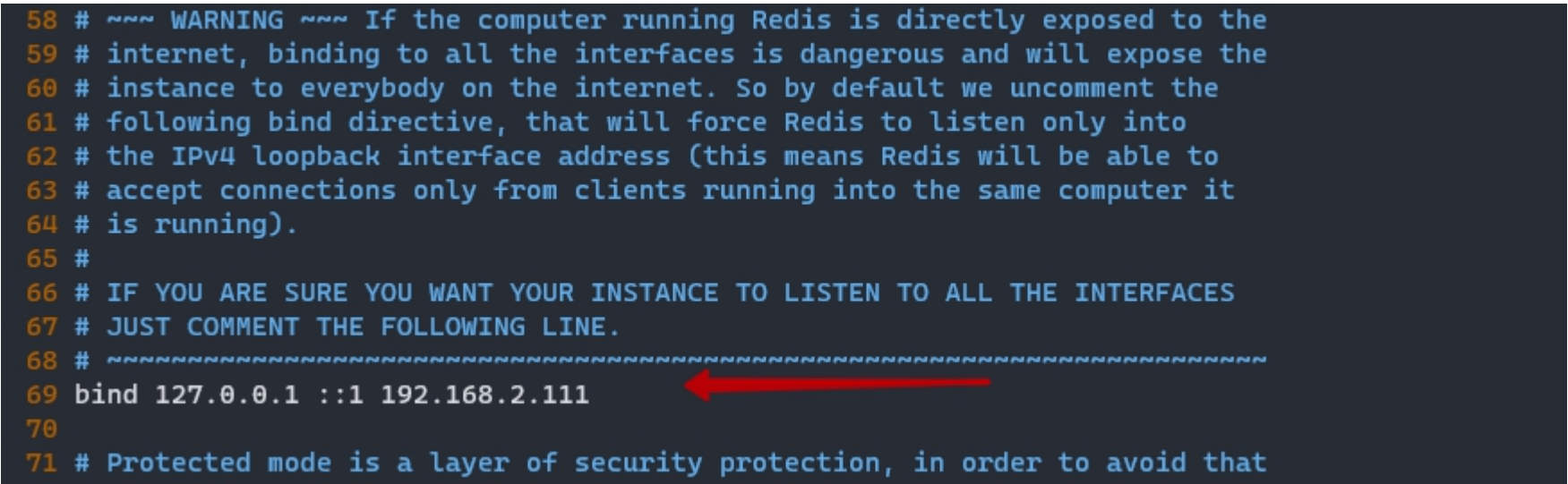
```
alex@ubuntu:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
   inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP group default qlen 1000
   link/ether 00:15:5d:d5:b1:26 brd ff:ff:ff:ff:ff:ff
   inet 192.168.2.111/24 scope global eth0
       valid_lft forever preferred_lft forever
```

Есть альтернативный подход в виде адреса '0.0.0.0', который говорит о возможности подключения с любого интерфейса. Попробуйте его, если столкнетесь с ошибками.



Изменить параметр 'bind' мы можем открыв конфигурационный файл:

```
sudo vim /etc/redis/redis.conf
```



Перезапустите сервис:

```
sudo systemctl restart redis-server
```

Если у вас выключен фаервол - вы сможете подключиться к Redis. Иначе вам нужно прописать правила, которые разрешают подключение к 6379 TCP порту.

Прописать правило, которое разрешить подключение только диапазону адресов в следующей подсети '192.168.0.0/16', можно так:

```
sudo ufw allow from 192.168.0.0/16 to any port 6379 proto tcp
```

Альтернативно вы можете разрешить подключение всем используя следующую команду:

```
sudo ufw allow 6379
```

## Аутентификация и создание пароля

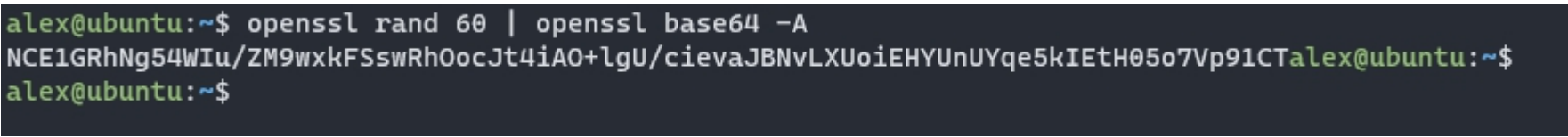
На данный момент любой пользователь может подключиться к Redis, если IP адреса совпадают. Мы можем установить пароль, который будет запрашиваться перед подключением. Для этого вам нужно найти и раскомментировать следующую строчку в файле конфигурации:

```
# requirepass foobared
```

Вместо 'foobared' вы должны ввести свой пароль. Как написано в самом файле конфигурации - Redis это быстрая база и может принимать 150к попыток ввода пароля в секунду. В связи с этим простой перебор паролей (брутфорс) весьма вероятная проблема. Разработчики советуют использовать более длинные пароли, чем вы это делаете обычно.

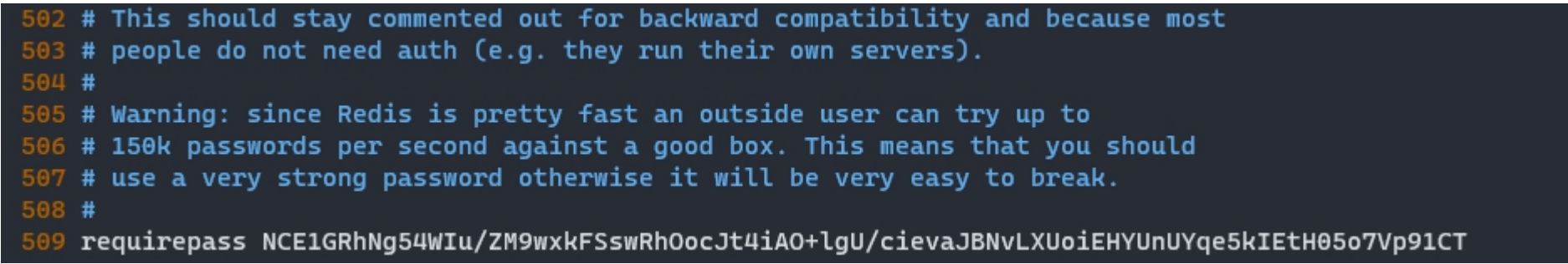
Вы можете сгенерировать длинный и безопасный пароль используя 'openssl':

```
openssl rand 60 | openssl base64 -A
```



Пример установленного пароля:

```
sudo vim /etc/redis/redis.conf
```



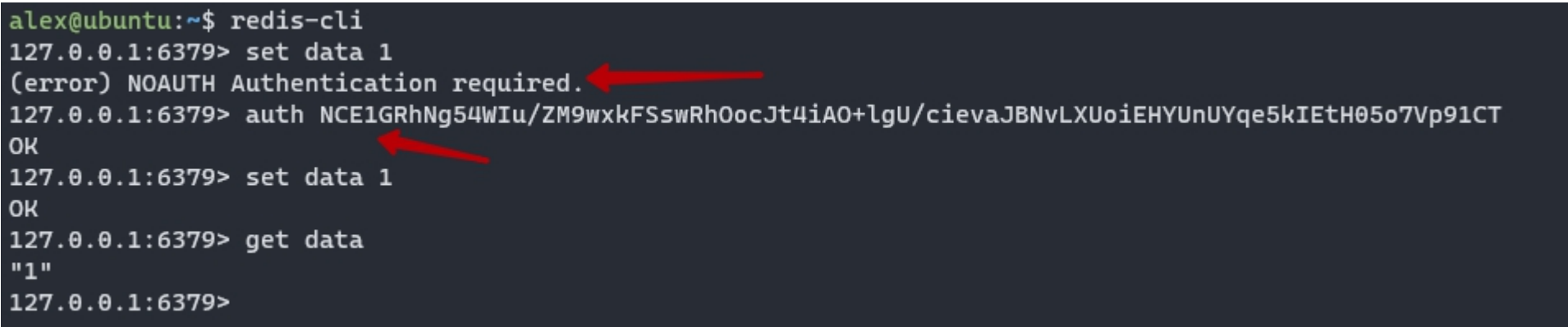
Повторно перезагружаем сервис что бы изменения вступили в силу:

```
sudo systemctl restart redis-server
```

Теперь вы можете подключаться удаленно используя пароль.

Удаленные подключения, чаще всего, выполняются какой-то библиотекой, в которой реализован процесс аутентификации. Если вы планируете подключаться локально, то этот процесс может потребовать ручного ввода данных. Что бы пройти аутентификацию локально мы должны войти в CLI, а затем использовать команду 'auth' с вашим паролем:

```
redis-cli
auth ваш_пароль
```



Как видно на примере выше мы получим ошибку '(error) NOAUTH Authentication required.' если не введем пароль.

## Бэкап и восстановление

В Redis существует 3 модели работы с бэкапом:

- **RDB** - создание бэкапов с определенными интервалами. Состоит из 1-ого файла;
- **AOF (append only file)** - создание бэкапа после каждой записи данных (журналирование команд). Состоит из 2 файлов;
- **RDB+AOF** - совмещает обе предыдущих модели, но по умолчанию будет восстанавливаться AOF т.к. имеет более актуальную информацию.

Создание бэкапов так же можно отключить. По умолчанию включен RDB, а AOF отключен.

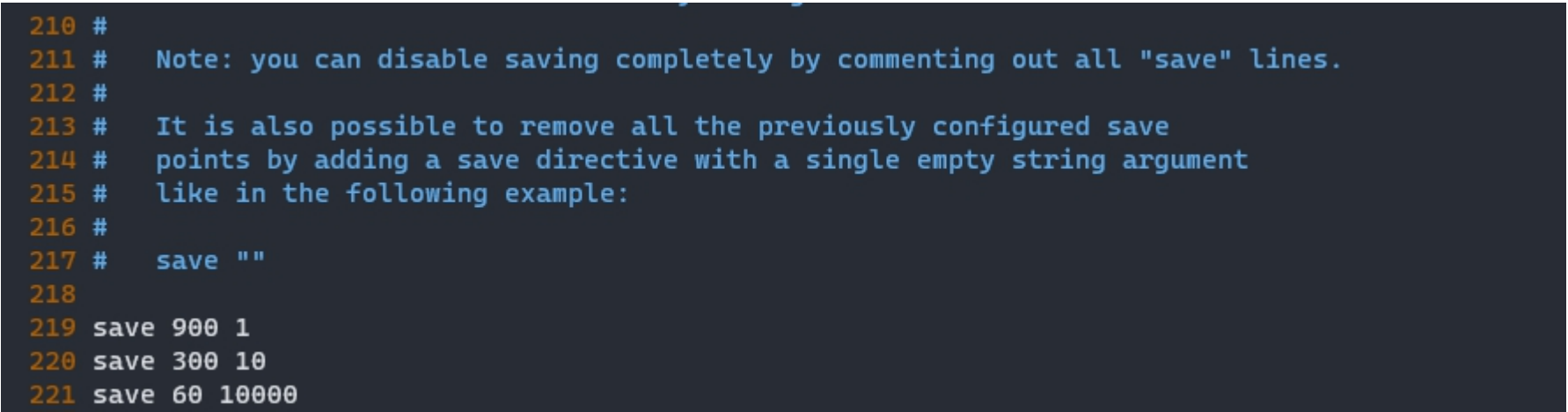
Большая часть изменений связанных с бэкапом делается в конфигурационном файле:

```
sudo vim /etc/redis/redis.conf
```

## Настройка RDB

Открыв конфигурационный файл можно увидеть, параметры 'save', которые имеют следующее значения:

```
save <количество секунд> <количество изменений>
```



Т.е. выгрузка в файл происходит через определенный интервал и при условии какого-то количества изменений в базе. Если убрать эти строки, то вы отключите RDB.

Создать бэкап можно так же через CLI выполнив одну из двух команд:

- **SAVE** - блокирует работу базы до создания бэкапа.
- **BGSAVE** - клонирует (fork()) существующий процесс Redis и выгружает его данные на диск. После этого процесс завершает работу. Таким образом нет блокировок, но возрастает требование к памяти.

Настройки установленные в конфигурационном файле всегда выполняют BGSAVE. Просто SAVE выполняется только через CLI.



Пример выполнения бэкапа через консоль:

```
redis-cli SAVE
```

```
alex@ubuntu:~$ redis-cli
127.0.0.1:6379> SAVE
OK
```

Путь сохранения файла бэкапа и само его название определяется в конфигурационном файле. По умолчанию оно равно: '/var/lib/redis/dump.rdb'.

```
253 # The filename where to dump the DB
254 dbfilename dump.rdb
255
256 # The working directory.
257 #
258 # The DB will be written inside this directory, with the filename specified
259 # above using the 'dbfilename' configuration directive.
260 #
261 # The Append Only File will also be created inside this directory.
262 #
263 # Note that you must specify a directory here, not a file name.
264 dir /var/lib/redis
265
266 ##### REPLICATION #####
```

Что бы восстановить подобный файл, мы должны выполнить следующее:

- 1. Остановить сервер Redis. Иначе файл будет перезаписан;
- 2. Скопировать файл с бэкапом в директорию где был создан бэкап. Названия меняться не должно. Т.е. если бэкап был взят по пути '/var/lib/redis/dump.rdb', то и восстанавливаться он будет отсюда;
- 3. Запустить Redis.

Порядок команд для восстановления:

```
sudo systemctl stop redis-server
sudo cp ./backup.rdb /var/lib/redis/dump.rdb
sudo systemctl start redis-server
```

## Настройка AOF

Работу с базой можно настроить через принцип логов (типа транзакционных логов в SQL базах). Каждая команда записи и изменения записывается в файл с расширением '.aof'. Из-за того что файл '.aof' хранит все выполненные команды (которые могут повторяться 100-ни раз), он может весомо превышать основной файл данных.

Когда файл '.aof' достигает критической отметки, установленной в настройках, он повторно выгружает все данные из памяти в новый файл. Так как выгрузка идет из памяти, а не из файла '.aof', у нас уже нет дубликатов. Если к серверу поступает какая-то новая команда она записывается в старый и новый файл. После полной выгрузки старый файл заменяется новым.

Изменить принцип работы бэкапа можно в конфигурационном файле. Кроме параметров журналирования есть и другие:

- 1. **appendonly no** - говорит, нужно ли выполнять AOF бэкап. По умолчанию отключен. Нужно установить 'yes' что бы включить;
- 2. **appendfilename "appendonly.aof"** - имя файла, которое будет создаваться при бэкапе и читаться при восстановлении;
- 3. **appendfsync everysec** - как часто нужно выполнять выгрузку. По умолчанию - каждую секунду. Можно так же установить параметр 'always' (сразу после изменения данных) и 'no' (раз в 30 секунд);
- 4. **auto-aof-rewrite-min-size 64mb** - минимальный размер AOF файла, после которого произойдет перезапись;
- 5. **auto-aof-rewrite-percentage 100** - на сколько процентов должен увеличиться размер файла после последней перезаписи. Если первая запись сработала при 64mb, то при значении 100%, она повторно сработает на 128mb. Значение 0 - отключение автоматической перезаписи.

```
700 appendonly yes
701
702 # The name of the append only file (default: "appendonly.aof")
703
704 appendfilename "appendonly.aof"
705
706 # The fsync() call tells the Operating System to actually write data on disk
707 # instead of waiting for more data in the output buffer. Some OS will really flush
708 # data on disk, some other OS will just try to do it ASAP.
709 #
710 # Redis supports three different modes:
711 #
712 # no: don't fsync, just let the OS flush the data when it wants. Faster.
713 # always: fsync after every write to the append only log. Slow, Safest.
714 # everysec: fsync only one time every second. Compromise.
715 #
716 # The default is "everysec", as that's usually the right compromise between
717 # speed and data safety. It's up to you to understand if you can relax this to
718 # "no" that will let the operating system flush the output buffer when
719 # it wants, for better performances (but if you can live with the idea of
720 # some data loss consider the default persistence mode that's snapshotting),
721 # or on the contrary, use "always" that's very slow but a bit safer than
722 # everysec.
723 #
724 # More details please check the following article:
725 # http://antirez.com/post/redis-persistence-demystified.html
726 #
727 # If unsure, use "everysec".
728
729 # appendfsync always
730 appendfsync everysec
731 # appendfsync no
732
733 # When the AOF fsync policy is set to always or everysec, and a background
"/etc/redis/redis.conf" 1373L, 61984C written
```

Перед включением AOF важно помнить, что если вы его только включили, то **все данные будут стерты**. После установки параметров и перезапуске сервера, Redis попыбует прочитать файл с расширением '.aof'. Так как его нет - будет создан новый, пустой, файл. Избежать это можно установкой параметра через CLI:

```
sudo redis-cli

CONFIG SET appendonly yes
CONFIG REWRITE
```

```
alex@ubuntu:~$ redis-cli
127.0.0.1:6379> CONFIG SET appendonly yes
OK
127.0.0.1:6379> CONFIG REWRITE
OK
```

Если у вас появится ошибка '(error) ERR Rewriting config file: Permission denied', то попробуйте сменить владельца папки:

```
sudo chown redis:redis /etc/redis/redis.conf
```

Что бы не ждть ситуации, когда файл журнала будет перезаписан, можно выполнить следующую команду:

```
BGREWRITEAOF
```

Пример на скриншоте ниже показывает, что файл состоял из 33 строк. После выполнения команды 'BGREWRITEAOF' размер файла сократился до 1 строки т.к. содержал в себе дублирующие команды:

```
alex@alex:~$ sudo cat /var/lib/redis/appendonly.aof | wc
    33     33    147
alex@alex:~$ redis-cli
127.0.0.1:6379> BGREWRITEAOF
Background append only file rewriting started
127.0.0.1:6379> exit
alex@alex:~$ sudo cat /var/lib/redis/appendonly.aof | wc
     1      4    106
alex@alex:~$
```

Бэкап будет находиться по следующему пути: '/var/lib/redis/appendonly.aof' (папка и название файла по умолчанию).

Что бы выполнить восстановление базы - вы так же должны остановить сервис Redis:

```
sudo systemctl stop redis-server
```

Как уже писалось раньше, при включенном AOF и RDB, всегда будет искаться и восстанавливаться файл '.aof'. Из-за такого поведения, если вам нужно восстановить базу '.rdb', изначально вам нужно отключить AOF. Если вы и планировали восстанавливаться AOF, то вам просто нужно переместить файл по пути '/var/lib/redis/appendonly.aof', и включить сервер:

```
sudo cp ./appendonly.aof /var/lib/redis/  
sudo systemctl start redis-server
```

## Какую модель бэкапа использовать

Выбор модели, которую стоит использовать, зависит от ресурсов выделенных под Redis и важности данных.

Если диски у вас и так сильно нагружены, а потеря данных за последние 5-10 минут не является критической, то RDB лучший вариант. Так же стоит учитывать, что при бэкапе выполняется fork()(клонирование) основного процесса, что повышает требование к оперативной памяти.

Если диски у вас не самое слабое место и вам нужно максимально сохранить данные - лучший вариант AOF. Работу этой модели вы можете настроить с сохранением после каждой команды.


...

Рекомендую





Подписывайтесь на наш Telegram канал

Теги: [#ubuntu](#) [#redis](#)

### Комментарии



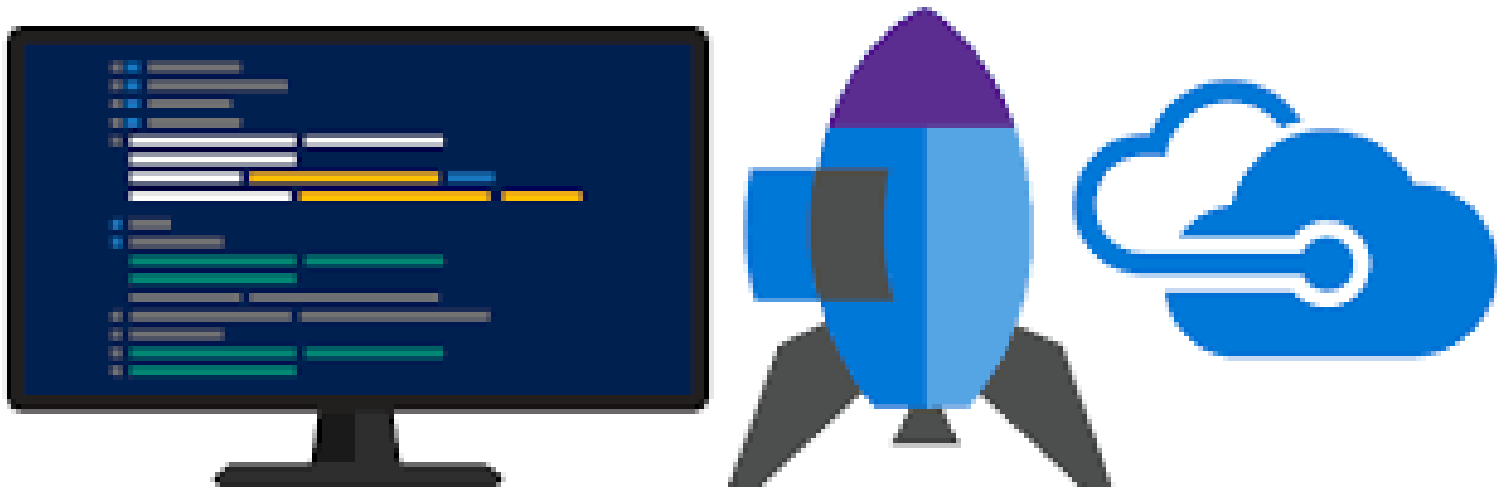
Ваш комментарий...



поделиться с друзьями

Отправить

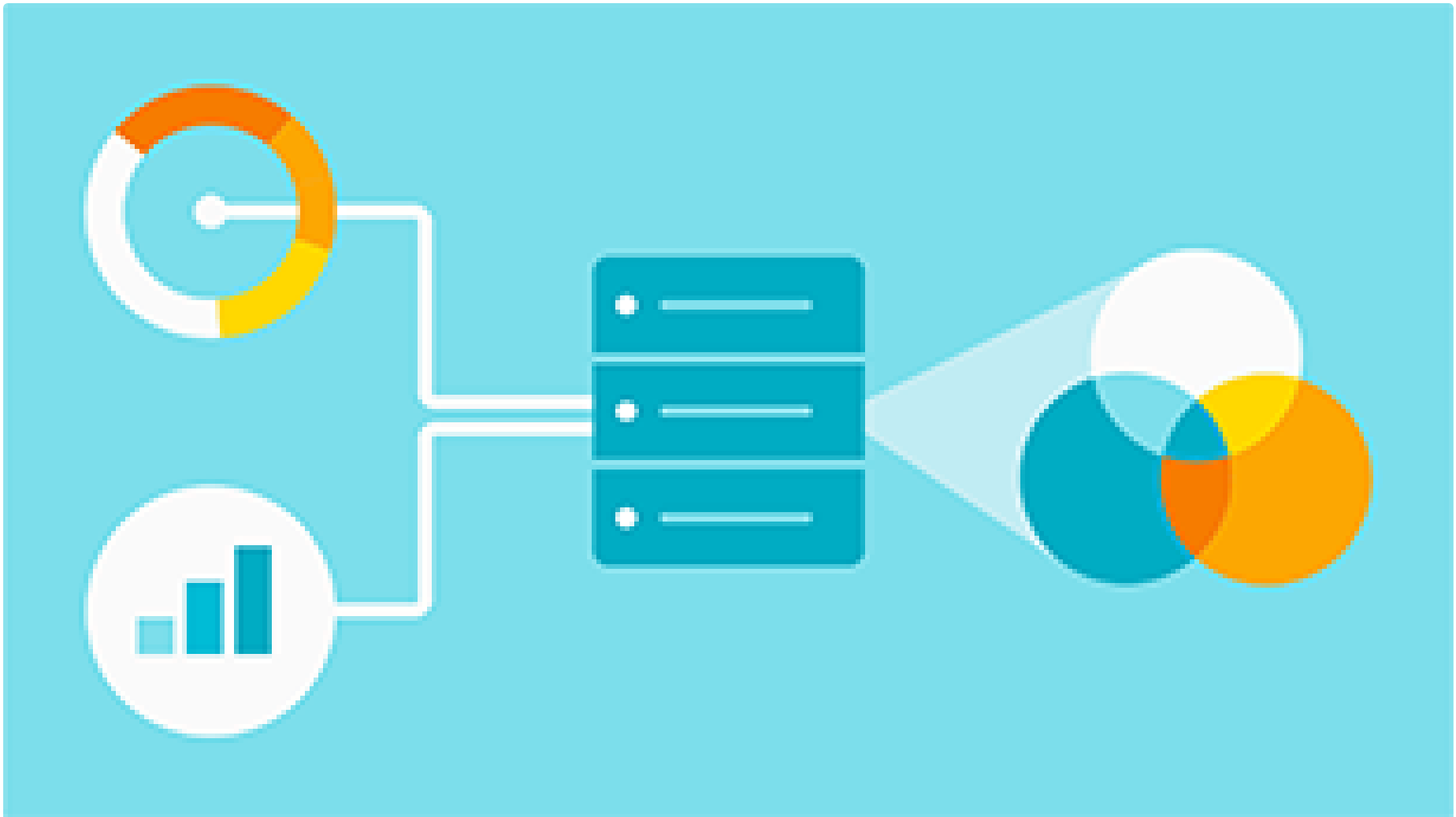
### Последняя статья



### Создаем страницу регистрации пользователей Django

### Популярная статья:





Работа с папками и создание путей с модулем OS в Python

Каналы



Популярные тэги

- [powershell](#)
- [windows](#)
- [ms-sql](#)
- [wmi](#)
- [ad](#)
- [hyper-v](#)

О блоге

Этот блог представляет собой конспекты выученного материала, приобретённого опыта и лучшие практики в системном администрировании и программировании.

[Обратная связь](#)

[Онлайн сервисы](#)

© FixMyPC 2019 - 2023