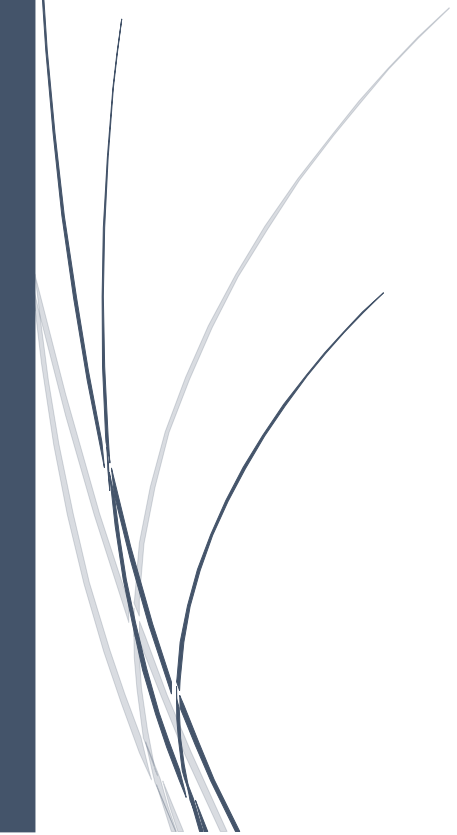


Платформа  
ОММГ 3.0.  
Инструкция  
по установке  
на ОС РОСА.



Георгий Хазан  
ОММДЖИ Технолоджи

# Требования к аппаратному обеспечению и системному ПО.

Для развёртывания платформы OMMG 3.0 (далее - Платформы) требуются:

- для развёртывания корпоративной Платформы при нагрузке не выше 5000 соединений требуется/рекомендуется
  - компьютер с 16/32 ГБ RAM, 4/8 ядрами серверных процессоров класса Intel E5-2620v4 и 3/5 жёсткими дисками, объединёнными в RAID10.
  - виртуальная машины с 16 ГБ RAM, 6-8 ядрами процессора, не менее 30 ГБ дискового пространства (следует предусмотреть дальнейшее наращивание дисков при интенсивном использовании хранилища файлов)
- для развёртывания Платформы, рассчитанной на большие нагрузки, применяются отдельные сервера трёх видов:
  - фронтенд-сервер (www/voice) - сервер с 24 или 32 ядрами и достаточным объёмом памяти (не менее 16 ГБ). Такие сервера нетребовательны к жёстким дискам.
  - сервер баз данных - с развитой дисковой подсистемой и большим объёмом памяти. Для поддержки 100 тысяч пользователей в онлайн понадобятся минимум два таких сервера, для миллиона - четыре.
  - файловый сервер - соединяет высокую вычислительную мощь для конвертации закачанных звуковых и видео-сообщений в общий формат с большой дисковой ёмкостью. Обычно для конфигурации до 100 тысяч в онлайн хватает одного такого сервера, для миллиона в онлайн - в зависимости от объёма закачанных файлов. Может предусматриваться возможность стирания старых файлов (например, спустя месяц), если ёмкости дисков перестанет хватать.
- все компьютеры работают под управлением ОС РОСА (версия 2021.1 или старше).

## Развёртывание базы данных.

Для того, чтобы повысить нагрузочную способность Платформы, её база данных может быть разбита на 16 или 256 частей (шардов), каждая из которых может исполняться на отдельном компьютере. При возрастании нагрузки можно наращивать число физических компьютеров, обслуживающих базу данных, без остановки сервиса.

Платформа OMMG разворачивается поверх СУБД Postgres или Postgres Pro. Эта операция производится на каждом сервере БД.

- От имени root необходимо исполнить команду:  

```
#dnf install -y postgresql17-server
#su - postgres
```
- При необходимости настроить сервер (например, изменить путь для хранения файлов БД) отредактируйте файл настроек:  

```
$nano ~/.profile
```
- Создайте каталог с данными  

```
$/usr/libexec/postgresql17/initdb
```
- Запустите сервер Postgres:  

```
#systemctl enable postgresql17.service
#systemctl start postgresql17.service
```
- Войдите в командную строку Postgres:  

```
#su - postgres
$psql
postgres=#
```
- Создайте пользователя для доступа к БД:  

```
postgres=# CREATE USER db WITH PASSWORD 'xxxxxxxxxxxxx' CREATEDB;
```
- Залейте структуру базу данных из скриптов, входящих в дистрибутив системы:  

```
$cd /var/www/ommg3_api/pgsql
$cat db.sql | psql
$cat dbglobal.sql | psql db
$cat dbstats.sql | psql dbstats
$for i in `seq 1 16`; do cat dbshard.sql | psql -v SCHEMA=db$i dbshard; done
$for i in `seq 1 16`; do cat sprocs.sql | sed 's/SCHEMA/db'$i'/g' | psql \
dbshard; done
```

## Развёртывание веб-фронтенда

Данный набор операций производится на каждой из машин, предназначенных для функционирования в качестве www-сервера.

### Установка php-fpm

- установите серверную часть php:  

```
#dnf install -y php-fpm
```

- настройте его и перезапустите сервер

```
#nano /etc/php-fpm.d/default.conf
```

```
[www]
```

```
user = nobody
```

```
group = nobody
```

```
listen = 127.0.0.1:9001
```

```
listen.owner = nobody
```

```
listen.group = nobody
```

```
pm = dynamic
```

```
pm.max_children = 5
```

```
pm.start_servers = 2
```

```
pm.min_spare_servers = 1
```

```
pm.max_spare_servers = 3
```

- перезапустите сервер

```
#systemctl restart php-fpm
```

## Установка php

В настоящий момент Платформа использует php версии 7.4 или совместимый с ним:

- установите сам php:

```
#dnf install -y php-cli
```

- установите расширения php:

```
#dnf install -y php-curl php-json php-pdo php-pdo_pgsql php-pgsql \  
php-openssl php-imagick php-fileinfo php-pcntl
```

- отконфигурируйте файлы `/var/www/ommg3_api/conf/default.php` и `/var/www/ommg3_api/conf/sconfig.php` таким образом, чтобы там были верные названия доменов, имена пользователей, пароли на базу данных и прочее.

- создайте каталог для логов:

```
#mkdir /var/log/php_log
```

```
#chown nobody:nobody /var/log/php_log
```

```
#chmod 775 /var/log/php_log
```

- запустите конфигуратор базы данных:

```
$cd /var/www/ommg3_api/pgsql
```

```
$php autoinc.php
```

- убедитесь, что php имеет нормальный доступ к MySQL, и вышеозначенный скрипт выполнен без ошибок.

## Установка nginx

Для функционирования платформы требуется nginx версии 1.26 или старше. Установка производится из исходного кода.

- скачайте исходный код nginx и распакуйте его:

```
#cd /opt
```

```
#wget https://nginx.org/download/nginx-1.26.3.tar.gz
```

```
#tar -xvzf nginx-1.26.3.tar.gz
```

- скопируйте исходный код модулей:  
#cp -R /var/www/nginx/modules/\* /opt
- установите нужные библиотеки:  
#dnf install -y nginx libssl1 openssl-devel pcre pcre-devel lib64geoip-devel \ lib64gd-devel lib64xslt-devel
- если у вас не установлен компилятор C++, установите его:  
#dnf install -y basesystem-build
- сконфигурируйте nginx:  
#cd /opt/nginx-1.26.3  
#./configure --prefix=/etc/nginx --sbin-path=/usr/sbin/nginx \ --modules-path=/usr/lib/nginx/modules --conf-path=/etc/nginx/nginx.conf \ --error-log-path=/var/log/nginx/error.log \ --http-log-path=/var/log/nginx/access.log \ --pid-path=/var/run/nginx.pid --lock-path=/var/run/nginx.lock \ --http-client-body-temp-path=/var/cache/nginx/client\_temp \ --http-proxy-temp-path=/var/cache/nginx/proxy\_temp \ --http-fastcgi-temp-path=/var/cache/nginx/fastcgi\_temp \ --http-uwsgi-temp-path=/var/cache/nginx/uwsgi\_temp \ --http-scgi-temp-path=/var/cache/nginx/scgi\_temp \ --user=nginx --group=nginx --with-http\_ssl\_module --with-http\_realip\_module \ --with-http\_addition\_module --with-http\_sub\_module --with-http\_dav\_module \ --with-http\_flv\_module --with-http\_mp4\_module --with-http\_gunzip\_module \ --with-http\_gzip\_static\_module --with-http\_random\_index\_module \ --with-http\_secure\_link\_module --with-http\_stub\_status\_module \ --with-http\_auth\_request\_module --with-http\_xslt\_module \ --with-http\_image\_filter\_module --with-http\_geoip\_module \ --with-threads --with-stream --with-stream\_ssl\_module \ --with-http\_slice\_module --with-mail --with-mail\_ssl\_module \ --with-http\_v2\_module \ --add-module=/opt/nginx-headers-more-module \ --add-module=/opt/nginx-upload-module \ --add-module=/opt/nginx-upload-progress-module
- остановите предустановленный nginx:  
#nginx -s stop
- соберите и установите nginx:  
#make  
#make install
- настройте виртуальные хосты:  
#mkdir /etc/nginx/conf.d  
#cp /var/www/nginx/conf.d/\* /etc/nginx/conf.d  
#cp /var/www/nginx/nginx.conf /etc/nginx
- создайте сертификат для ssl  
#mkdir /etc/nginx/ssl  
#cd /etc/nginx/ssl  
#openssl req -x509 -nodes -days 3650 -newkey rsa:2048 -keyout ommg.key \ -out ommg.crt

```
#cp /var/www/nginx/ssl/ssl.conf /etc/nginx/ssl
#vim /etc/nginx/ssl/ssl.conf
```

- создайте каталог для кэша nginx:  
`#mkdir /var/cache/nginx`
- создайте временный каталог для файлов:  
`#mkdir /var/tmp/nginx`  
`#chown nobody:nobody /var/tmp/nginx`
- создайте временный каталог для загрузки файлов по частям:  
`#mkdir /var/tmp/chunks`  
`#chown nobody:nobody /var/tmp/chunks`
- запустите nginx:  
`#nginx`
- убедитесь в том, что после перезапуска nginx отсутствуют ошибки

## Установка redis

Redis (ver\_2.8.4) используется в Платформе для высоконагруженных кэшей прикладного сервера.

- установите собственно сервер:  
`#dnf install -y redis`
- настройте оповещение о событиях:  
`#nano /etc/redis.conf`  
`# By default all notifications are disabled because most users don't need`  
`# this feature and the feature has some overhead. Note that if you don't`  
`# specify at least one of K or E, no events will be delivered.`  
`notify-keyspace-events "Ex"`
- перезапустите сервер  
`#systemctl restart redis`
- для особо нагруженных систем рекомендуется устанавливать по два экземпляра redis на машине, на портах 6301 и 6302.
- для установки платформы на несколько серверов отключите привязку redis к localhost, закомментировав настройку bind в файле `/etc/redis/redis.conf`:  
`# bind 127.0.0.1`

## Установка node.js

В Платформе используется node.js версии 14.x (входит по умолчанию в набор пакетов):

- установите nodejs и packet manager:  
`#dnf install -y zeromq`
- настройте файл `/var/www/ommg3_nodejs/config.json`
- скопируйте файлы сервисов:  
`#cp /var/www/ommg3_nodejs/systemd/* /etc/systemd/system`
- запустите прикладной сервер:  
`#systemctl start ommg3_push.service`

```
#systemctl start ommg3_storage.service
```

```
#systemctl start ommg3.service
```

- если нужно, чтобы сервисы стартовали при запуске операционной системы, используйте команду enable:

```
#systemctl enable ommg3_push.service
```

```
#systemctl enable ommg3_storage.service
```

```
#systemctl enable ommg3.service
```

## Установка файлового сервера

Если файловый сервер ставится на отдельный сервер, то установите сначала php и nginx, как описано в предыдущем разделе. Настройте файл `/var/www/ommg3_files/sconfig.php` так же, как и для `ommg3_api`.

- редактирование настроек и заливка базы данных:

```
$cd /var/www/ommg3_files/sql
```

```
$nano global.sql
```

```
>замените домен после слов "storage #1" на нужный
```

```
#su - postgres
```

```
$cat global.sql | psql db
```

- установка необходимых утилит

```
#dnf install -y imagemagick ffmpeg gawk
```

- запуск конвертора медиа-файлов

```
$cd /var/www/ommg3_files/scripts
```

```
$php convert_multi.php &
```

- для загрузки больших файлов возможно потребуется внести изменения в настройки nginx и php:

```
#nano /etc/nginx/nginx.conf
```

```
client_max_body_size 100M;
```

```
#nginx -s reload
```

```
#nano /etc/php.ini
```

```
; Maximum allowed size for uploaded files.
```

```
; http://php.net/upload-max-filesize
```

```
upload_max_filesize = 20M
```

```
#nano /etc/php-cgi-fcgi.ini
```

```
; Maximum allowed size for uploaded files.
```

```
; http://php.net/upload-max-filesize
```

```
upload_max_filesize = 20M
```

```
#systemctl restart php-fpm
```

- создание системного пользователя (0001@), от имени которого Платформа будет рассылать свои сообщения:

```
$cd /var/www/ommg3_api/pmi
```

```
$php 0001.php
```

# Развёртывание сервиса звонков

Данный набор операций производится на каждой из машин, предназначенных для функционирования в качестве сервиса звонков.

## Установка coturn

- установите пакет coturn:  
`#dnf install -y coturn`
- скопируйте конфигурационный файл из дистрибутива:  
`#cp /var/www/ommg3_voice/coturn/turnserver.conf /etc/coturn`
- укажите внешний адрес вашего сервера:  
`#nano /etc/coturn/turnserver.conf`  
`#external-ip=xxx.xxx.xxx.xxx`
- запишите файл конфигурации и перезапустите сервис:  
`#systemctl restart coturn`

## Установка redis

Если на данном сервере не устанавливался redis в рамках предыдущих операций, установите и настройте redis.

- установите пакет:  
`#dnf install -y redis`
- внесите следующие изменения в конфигурацию:  
`#nano /etc/redis.conf`  
`timeout 0`  
`tcp-keepalive 60`
- перезапустите сервис:  
`#systemctl restart redis`

## Установка janus

- установите git и компилятор C, если потребуется:  
`#dnf install -y git-core basesystem-build meson`
- установите зависимости:  
`dnf install -y lib64microhttpd-devel lib64jansson-devel openssl-devel \`  
`lib64sofia-sip-devel lib64glib2.0-devel lib64opus-devel \`  
`lib64ogg-devel lib64curl-devel liblua5.3-devel lib64config-devel \`  
`lib64websockets-devel pkgconf libtool automake`
- скачайте и установите libnice:  
`#cd /opt`  
`#git clone https://gitlab.freedesktop.org/libnice/libnice`  
`#cd libnice`  
`#meson --prefix=/usr build && ninja -C build && sudo ninja -C build install`

- скачайте и установите libsrtp:
 

```
#cd /opt
#wget https://github.com/cisco/libsrtp/archive/v2.2.0.tar.gz
#tar xfv libsrtp-2.2.0.tar.gz
#cd libsrtp-2.2.0/
#./configure --prefix=/usr --enable-openssl
#make shared_library && sudo make install
```
- скачайте исходный код и соберите janus:
 

```
#cd /opt
#git clone https://github.com/meetecho/janus-gateway.git
#cd janus-gateway/
#git checkout v1.2.3
#sh autogen.sh
#PKG_CONFIG_PATH=/usr/lib/pkgconfig ./configure --prefix=/opt/janus
#make -j
#make install
```
- скопируйте файлы сервисов:
 

```
#cp /var/www/ommg3_voice/systemd/* /etc/systemd/system
```
- сконфигурируйте и запустите janus:
 

```
#cp /var/www/ommg3_voice/janus/config/* /opt/janus/etc/janus
#nano /opt/janus/etc/janus/janus.jcfg
# turn_server = "xxx.xxx.xxx.xxx"
```
- включите автозапуск сервиса janus:
 

```
#systemctl enable ommg3_janus
```
- запустите сервис janus:
 

```
#systemctl start ommg3_janus
```

## Установка сервиса звонков

- установите библиотеки:
 

```
#apt install -y lib64qt5websockets5
```
- настройте файл конфигурации, если нужно
 

```
#nano /var/www/ommg3_voice/calls_service/callservice.conf
```
- включите автозапуск сервиса звонков:
 

```
#systemctl enable ommg3_call
```
- запустите сервис звонков:
 

```
#systemctl start ommg3_call
```