Порядок действий по установке серверной части Vitro-CAD (Linux)

Сервер БД:

1. Установка необходимых пакетов ПО из репозиториев:

```
sudo apt-get update && sudo apt-get upgrade
sudo apt-get install unzip curl postgresql postgresql-client postgresql-contrib
```

2. Отредактировать конфигурационные файлы postgresql:

Проверяем версию postgresql

```
postgres --version
```

Открываем редактор с учетом установленной версии

```
nano /etc/postgresql/14/main/postgresql.conf
```

- изменить в секции # - Connection Settings - значение параметра listen_addresses с localhost на *
должно получиться:

```
listen_addresses = '*'
```

```
nano /etc/postgresql/14/main/pg_hba.conf
```

- изменить в секции # IPv4 Local connections: значение параметра ADDRESS с 127.0.0.1/32 на 0.0.0.0/0

должно получиться:

```
# IPv4 local connections:

host      all      all      0.0.0.0/0      scram-sha-256
```

3. Перезапустить службу:

```
sudo systemctl restart postgresql
```

4. Проверить выполняется ли прослушивание на порту 5432 для всех активных адресов:

```
netstat -ltnp

tcp      0      0 0.0.0.0:5432      0.0.0.0:*      LISTEN      14569/postgres
```

5. Добавить пользователя postgres в группу владельца директории, из которой выполняются действия:

Проверка владельца текущей папки (обозначена ".")

```
ls –la
```

Добавление пользователя postgres в группу владельца

```
sudo usermod -aG dir_owner_group postgres
```

6. Изменить права группе владельца директории (дать права на изменение):

```
chmod -R 775 dir_owner_group
```

7. Войти в cli postgresql от имени пользователя postgres:

```
sudo -u postgres psql
```

8. Создать пользователя с ролью суперпользователя:

```
postgres-#CREATE USER vitrodbuser WITH PASSWORD '@pwd4vitro!' SUPERUSER;
```

9. Создать новую базу данных с именем vitrodb:

```
postgres-#CREATE DATABASE vitrodb with owner = vitrodbuser encoding = 'UTF8' connection limit = -1;
```

10. Отключиться от cli postgresql:

```
postgres-# \q
```

11. Скачать и распаковать дистрибутив:

```
curl "https://linkTo/core.Web.Linux.zip" --output 'Core.Web.Linux.zip'

sudo unzip Core.Web.Linux.zip
```

12. Скопировать в доступное для пользователей расположение (к примеру tmp) файл резервной копии базы данных:

```
sudo cp ~/Core.Web/Linux/DB/vitro.bak /tmp
```

13. Запустить восстановление БД из бэкапа:

```
sudo -u postgres pg_restore -v -U vitrodbuser -h localhost -d vitrodb /tmp/vitro.bak
```

13.1 Запустить восстановление БД из бекапа (для файлов с расширением sql):

```
sudo -u postgres psql -U vitrodbuser -h localhost -d vitrodb < /tmp/vitro.bak.sql
```

Сервер приложений/фронт энд:

14. Установка необходимых пакетов ПО из репозиториев:

```
sudo apt-get update && sudo apt-get upgrade
sudo apt-get install unzip curl nginx nginx-extras
```

15. Скачать и распаковать дистрибутив:

```
curl "https://linkTo/Core.Web.Linux.zip" --output 'Core.Web.Linux.zip'

sudo unzip Core.Web.Linux.zip
```

16. Скопировать папку файлового хранилища в расположение, в которое смонтирован раздел для хранения:

```
sudo cp -r ~/Core.Web.Linux/DB/VitroFileStorage /mnt/storage
```

17. Сменить владельца для папки файлового хранилища:

```
sudo chown -R www-data:www-data /mnt/storage/VitroFileStorage
```

18. Скопировать папку приложения в желаемое расположение:

```
sudo cp -r ~/Core.Web.Linux /usr/local/Vitro.Server.Core.Web
```

19. Сменить владельца для папки приложения:

```
sudo chown -R www-data:www-data /usr/local/Vitro.Server.Core.Web
```

20. Установить атрибут "Исполняемый" для файла Vitro.Server.Core.Web:

```
sudo chmod +x /usr/local/Vitro.Server.Core.Web/Vitro.Server.Core.Web
```

21. Внести изменения в конфигурационный файл db.json:

```
sudo nano /usr/local/Vitro.Server.Core.Web/Vitro/Server/Conf/db.json
```

- изменить строку соединения, указав в качестве значения для атрибута "server" имя или IP сервера БД, а в качестве значения для атрибута "database" название созданной в п.7 базы данных:

Json

```
 "ConnectionString": "server=vitrodbserver;database=vitrodb;user id=vitrodbuser;password=@pwd4vitro!;Include Error Detail=true",
```

- изменить параметр "FileStoragePath", указав корректный путь к папке файлового хранилища:

Json

```
"FileStoragePath": "/mnt/storage/VitroFileStorage",
```

22. Создать и отредактировать файл модуля systemd для запуска службы:

```
sudo nano /lib/systemd/system/vitro-server.service
```

- содержимое файла модуля:

```
[Unit]
Description=Vitro .NET Web Application

[Service]
WorkingDirectory=/usr/local/Vitro.Server.Core.Web
ExecStart=/usr/local/Vitro.Server.Core.Web/Vitro.Server.Core.Web --urls http://localhost:4615
Restart=always
# Restart service after 10 seconds if the dotnet service crashes:
RestartSec=10
KillSignal=SIGINT
SyslogIdentifier=dotnet-example
User=www-data
SyslogIdentifier=Vitro-server

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

23. Применить изменения и запустить службу:

```
sudo systemctl enable vitro-server

sudo systemctl start vitro-server
```

24. Проверить выполняется ли прослушивание на порту, заданном в файле модуля (здесь 4615) для адреса Lo интерфейса:

```
netstat -ltnp

Proto Recv-Q Send-Q Local Address      Foreign Address      State      PID/Program name

...

tcp             0           0 127.0.0.1:4615       0.0.0.0:*                   LISTEN   -

...
```

Конвертация комплекта сертификата '*'.pfx в crt/pem средствами OpenSSL

25. Скачать сертификат в формате *.pfx

```
curl "https://linkTo/example.ru.pfx" --output 'example.ru.pfx'
```

26. для получения из *.pfx сертификата сервера в формате pem выполнить:

```
openssl pkcs12 -in example.ru.pfx -clcerts -nokeys -out example.ru.pem
```

в процессе будет запрошен пароль

27. для получения из *.pfx приватного ключа в формате pem выполнить:

```
openssl pkcs12 -in example.ru.pfx -nocerts -out example.ru.key
```

в процессе будет запрошен пароль. Далее будет запрошен новый пароль для нового файла ключа (в этом случае в качестве пароля вводится НЕ пустое значение, любой пароль)

28. для получения незащищенного паролем приватного ключа (из защищенного паролем файла ключа полученного на предыдущем шаге) выполнить:

```
openssl rsa -in example.ru.key -out example.ru.pem
```

29. Скопировать полученные файлы сертификата и незащищенного паролем приватного ключа в /etc/ssl/private (либо иное удобное расположение) `cp example.ru.* /etc/ssl/private`

30. Настроить обратное проксирование при помощи nginx, отредактировав конфигурационный файл:

```
sudo nano /etc/nginx/sites-available/default
```

должно получиться:

/etc/nginx/sites-available/default

```
# Default server configuration
#

server {
        listen 80 default_server;
        listen [::]:80 default_server;

        # Only allow access if the host is correct
        if ( $host != "vitro.example.ru" ){
            return 444; #CONNECTION CLOSED WITHOUT RESPONSE
        }

        #permanent redirekt from http to https
        return 301 https://$host$request_uri;
}

server {
        # SSL configuration
        #
        listen 443 ssl default_server;
        listen [::]:443 ssl default_server;

        server_name vitro.example.ru;

        ssl_certificate     /etc/ssl/private/example.ru.pem;
        ssl_certificate_key /etc/ssl/private/example.ru.key;

        # Only allow access if the host is correct
        if ( $host != "vitro.example.ru" ){
            return 444; #CONNECTION CLOSED WITHOUT RESPONSE
        }

        proxy_buffer_size   1M;
        proxy_buffers   4 1M;

        location / {
                #proxying traffic to upstream application (Vitro.Server.Core.Web)
                proxy_pass http://127.0.0.1:4615;
        }

}
```

31. Для обеспечения загрузки больших файлов добавить в конфигурационный файл сайта Nginx в явном виде значение параметра `client_max_body_size`

/etc/nginx/sites-available/default

```
client_max_body_size 100000M;
```

32. Также для исключения переполнения корневого раздела диска, необходимо включить в Nginx кеширование для обратного проксирования.
ключи для помещения в секцию server (или location):

/etc/nginx/sites-available/default

```
                        proxy_buffering        on;
```

/etc/nginx/nginx.conf

```
        proxy_cache_path  /var/cache/nginx  levels=1:2    keys_zone=STATIC:10m inactive=24h  max_size=1g;
```

итоговые конфиги:

/etc/nginx/nginx.conf

```
user www-data;
worker_processes auto;
pid /run/nginx.pid;
include /etc/nginx/modules-enabled/*.conf;
events {
        worker_connections 768;
        # multi_accept on;
}
http {
        ##
        # Basic Settings
        ##
        sendfile on;
        tcp_nopush on;
        types_hash_max_size 2048;
        # server_tokens off;
        # server_names_hash_bucket_size 64;
        # server_name_in_redirect off;
        include /etc/nginx/mime.types;
        default_type application/octet-stream;
        ##
        # SSL Settings
        ##
        ssl_protocols TLSv1 TLSv1.1 TLSv1.2 TLSv1.3; # Dropping SSLv3, ref: POODLE
        ssl_prefer_server_ciphers on;
        ##
        # Logging Settings
        ##
        access_log /var/log/nginx/access.log;
        error_log /var/log/nginx/error.log;
        ##
        # Gzip Settings
        ##
        gzip on;

        proxy_cache_path  /var/cache/nginx  levels=1:2    keys_zone=STATIC:10m inactive=24h  max_size=1g;

        # gzip_vary on;
        # gzip_proxied any;
        # gzip_comp_level 6;
        # gzip_buffers 16 8k;
        # gzip_http_version 1.1;
        # gzip_types text/plain text/css application/json application/javascript text/xml application/xml
application/xml+rss text/javascript;
        ##
        # Virtual Host Configs
        ##
        include /etc/nginx/conf.d/*.conf;
        include /etc/nginx/sites-enabled/*;
}
```

/etc/nginx/sites-available/default

```
# Default server configuration
#

server {
        listen 80 default_server;
        listen [::]:80 default_server;

        # Only allow access if the host is correct
        if ( $host != "vitro.example.ru" ){
            return 444; #CONNECTION CLOSED WITHOUT RESPONSE
        }

        #permanent redirekt from http to https
        return 301 https://$host$request_uri;
}

server {
        # SSL configuration
        #
        listen 443 ssl default_server;
        listen [::]:443 ssl default_server;

        server_name vitro.example.ru;

                client_max_body_size 100000M;

        ssl_certificate     /etc/ssl/private/example.ru.pem;
        ssl_certificate_key /etc/ssl/private/example.ru.key;

        # Only allow access if the host is correct
        if ( $host != "vitro.example.ru" ){
            return 444; #CONNECTION CLOSED WITHOUT RESPONSE
        }

        proxy_buffer_size   1M;
        proxy_buffers   4 1M;

        location / {
                #proxying traffic to upstream application (Vitro.Server.Core.Web)
                proxy_pass http://127.0.0.1:4615;
                                proxy_buffering        on;
        }

}
```

33. Выполнить перезагрузку службы nginx:

```
sudo systemctl reload nginx
```