**ИНСТРУКЦИЯ ПО СКАЧИВАНИЮ И УСТАНОВКЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ КОМПЛЕКСА ДЛЯ ИНСПЕКЦИОННОГО КОНТРОЛЯ (ICS)**

**НА 12 ЛИСТАХ**

[Подготовка сервера 2](#_Toc65244909)

[Установка Docker CE и вспомогательных утилит 2](#_Toc65244910)

[Установка и настройка БД PostgreSQL 2](#_Toc65244911)

[Установка и настройка БД Redis 3](#_Toc65244912)

[Установка и настройка Elasticsearch 3](#_Toc65244913)

[Подготовка конфигурации 5](#_Toc65244914)

[Настройка конфигурации http сервера nginx 5](#_Toc65244915)

[Установка ssl сертификата 7](#_Toc65244916)

[Настройка компонент системы 7](#_Toc65244917)

[Запуск системы 10](#_Toc65244918)

# Подготовка сервера

## Установка Docker CE и вспомогательных утилит

ПО портала работает с использованием технологий контейнеризации Docker. На сервер необходимо установить Docker версии не ниже 19.03.6 по инструкции с сайта <https://docs.docker.com/engine/install>

Для использования команд docker без sudo добавьте вашего пользователя в группу docker:

$ sudo usermod -aG docker $USER

Завершите ssh сессию на сервере для применения изменений. При повторном входе на сервер проверьте возможность использования команд docker без sudo.

$ docker ps

Вывод команды должен быть следующим:

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

Для настройки конфигурации всех контейнеров используется единый конфигурационный YAML файл и утилита docker compose для запуска служб из этой конфигурации.

Необходимо установить Docker compose версии не ниже 1.17.1 по инструкции с сайта https://docs.docker.com/compose/install

## Установка и настройка БД PostgreSQL

На сервере необходимо установить PostgreSQL версии 12.

Инструкции по установке можно найти на сайте <https://www.postgresql.org/docs/12/tutorial-install.html>

После установки необходимо произвести первичную настройку системы.

В консоли Postgresql с ролью postgres создайте базу данных и пользователя с назначением всех прав на эту базу.

$ sudo -u postgres psql

postgres=# create database revizor\_production;

postgres=# create user revizor\_user with encrypted password ‘password’;

postgres=# grant all privileges on database revizor\_production to revizor\_user;

## Установка и настройка БД Redis

На сервере необходимо установить Redis версии 5.0.7

Инструкции по установке можно найти на сайте <https://redis.io/topics/quickstart>

## Установка и настройка Elasticsearch

ПО Elasticsearch запускается внутри Docker контейнера. Для этого используется конфигурационный файл docker-compose.yml, в котором описаны необходимые для запуска и работы параметры.

Для запуска elasticsearch необходимо создать специальную директорию и установить права на доступ к ней

$ sudo mkdir /var/elastic

$ sudo chown 1000:1000 /var/elastic

Создайте файл docker-compose.yml в директории **/elasticsearch** и поместите приложенный ниже текст.

$ sudo vim /elasticsearch/docker-compose.yml

*docker-compose.yml для elasticsearch*

version: "3.4"

services:

es:

container\_name: es

hostname: es

image: elasticsearch:7.5.2

environment:

- discovery.type=single-node

- xpack.security.enabled=false

- ES\_JAVA\_OPTS=-Xms512m -Xmx512m

ports:

- "9200:9200"

volumes:

- '/var/elastic:/usr/share/elasticsearch/data'

Запуск elasticsearch:

$ cd /elastcsearch

$ docker-compose up -d es

Для того, чтобы убедиться, что контейнер запустился необходимо выполнить команду:

$ docker ps -a

В случае успешного запуска контейнера вывод команды должен быть следующим:

<container\_id> elasticsearch:7.5.2 "/usr/local/bin/dock…" <n> months ago Up <n> months 0.0.0.0:9200->9200/tcp, 9300/tcp es

Для просмотра логов:

$ docker logs -ft es

Для остановки процесса:

$ docker stop es

# Подготовка конфигурации

## Настройка конфигурации http сервера nginx

ПО Nginx запускается внутри Docker контейнера. На этапе настройки необходимо создать следующие директории:

**/var/nginx** – директория для конфигурационного файла nginx

**/etc/letsencrypt** – директория для SSL сертификатов

$ sudo mkdir /var/nginx /etc/letsencrypt

Создайте конфигурационный файл revizor\_app.conf в специальной директории и поместите в него приложенный ниже текст.

$ sudo vim /var/nginx/revizor\_app.conf

*revizor\_app.conf для nginx*

server {

server\_name app.revizor.tanto-s.ru;

location / {

proxy\_pass http://front\_app:3000;

}

location ~ ^/api {

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_set\_header X-Forwarded-Proto https;

proxy\_set\_header X-Forwarded-Port 443;

proxy\_set\_header Host $http\_host;

proxy\_redirect off;

proxy\_pass http://api:3000;

}

listen 443 ssl;

ssl\_certificate /etc/letsencrypt/private.cer;

ssl\_certificate\_key /etc/letsencrypt/private.pem;

}

server {

listen 80;

server\_name app.revizor.tanto-s.ru;

include conf.d/acme;

if ($host = app.revizor.tanto-s.ru) {

return 301 https://$host$request\_uri;

}

return 404;

}

Создайте конфигурационный файл revizor\_admin.conf в специальной директории и поместите в него приложенный ниже текст.

$ sudo vim /var/nginx/revizor\_admin.conf

*revizor\_admin.conf для nginx*

server {

server\_name revizor.tanto-s.ru;

location / {

proxy\_pass http://front\_admin:3000;

}

location ~ ^/api {

proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;

proxy\_set\_header X-Forwarded-Proto https;

proxy\_set\_header X-Forwarded-Port 443;

proxy\_set\_header Host $http\_host;

proxy\_redirect off;

proxy\_pass http://api:3000;

}

location /cable {

proxy\_pass http://api:3000;

proxy\_http\_version 1.1;

proxy\_set\_header Upgrade $http\_upgrade;

proxy\_set\_header Connection "upgrade";

proxy\_set\_header Host $host;

}

listen 443 ssl;

ssl\_certificate /etc/letsencrypt/private.cer;

ssl\_certificate\_key /etc/letsencrypt/private.pem;

}

server {

listen 80;

server\_name revizor.tanto-s.ru;

include conf.d/acme;

if ($host = revizor.tanto-s.ru) {

return 301 https://$host$request\_uri;

}

return 404;

}

## Установка ssl сертификата

В директории **/etc/letsencrypt** поместите файлы private.cer и private.pem для работы https соединения с порталом.

## Настройка компонент системы

ПО портала запускается внутри Docker контейнеров. Для его работы используется 5 контейнеров:

1. api – контейнер бэкенд сервера
2. front\_admin – контейнер фронтенд веб приложения администрирования
3. revizor-app – контейнер фронтенд мобильного веб приложения
4. nginx – контейнер http сервера
5. sidekiq – контейнер сервера отложенных задач
6. s3server — контейнер сервера хранения файлов

Docker образы хранятся в архиве <https://tanto-s.ru/icsinspectionfiles/docker_ics.zip>. Для инициализации образов на ВМ необходимо выполнить команду:

$ docker load --input <image\_name>.tar

Полученный образ можно увидеть с помощью команды:

$ docker image ls

Создайте файл docker-compose.yml в директории **/docker\_stack** и поместите приложенный ниже текст.

$ sudo mkdir /docker\_stack

$ sudo vim /docker\_stack/docker-compose.yml

*docker-compose.yml*

version: "3.4"

services:

nginx:

container\_name: nginx

image: nginx:1.17.9

volumes:

- '/var/nginx:/etc/nginx/conf.d'

- '/var/www/html:/var/www/html'

- '/etc/letsencrypt:/etc/letsencrypt'

depends\_on:

- api

- front\_app

- front\_admin

ports:

- 80:80

- 443:443

api:

container\_name: api

image: revizor\_api:latest

environment:

- RAILS\_ENV=

- RAILS\_HOSTNAME=

- CORS\_ORIGIN\_FRONT=

- RAILS\_MASTER\_KEY=1a79c13329be27f5d8dd3619c78854f4

- SIDEKIQ\_USERNAME=

- SIDEKIQ\_PASSWORD=

- REVIZOR\_API\_DATABASE\_HOST=

- REVIZOR\_API\_DATABASE\_PASSWORD=

- REVIZOR\_API\_DATABASE\_USER=

- REDIS\_URL=

- ELASTICSEARCH\_URL=

- FRONTEND\_URL=

- SENDGRID\_API\_KEY=

- AWS\_BUCKET\_NAME=

- AWS\_ACCESS\_KEY\_ID=

- AWS\_SECRET\_ACCESS\_KEY=

expose:

- '3000'

sidekiq:

container\_name: sidekiq

image: revizor\_api:latest

environment:

- RAILS\_ENV=

- RAILS\_HOSTNAME=

- CORS\_ORIGIN\_FRONT=

- RAILS\_MASTER\_KEY=1a79c13329be27f5d8dd3619c78854f4

- SIDEKIQ\_USERNAME=

- SIDEKIQ\_PASSWORD=

- REVIZOR\_API\_DATABASE\_HOST=

- REVIZOR\_API\_DATABASE\_PASSWORD=

- REVIZOR\_API\_DATABASE\_USER=

- REDIS\_URL=

- ELASTICSEARCH\_URL=

- FRONTEND\_URL=

- SENDGRID\_API\_KEY=

- AWS\_BUCKET\_NAME=

- AWS\_ACCESS\_KEY\_ID=

- AWS\_SECRET\_ACCESS\_KEY=

entrypoint: bundle exec sidekiq -e production

front\_app:

container\_name: front\_app

hostname: front\_app

image: revizor\_app:latest

expose:

- '3000'

front\_admin:

container\_name: front\_admin

hostname: front\_admin

image: revizor\_admin:latest

expose:

- '3000'

В случае, если имя docker образа отличается от того, что показано в файле, значение следует поменять на актуальное.

Для контейнеров api и sidekiq необходимо заполнить значения переменных, указанных в разделе environment файла docker-compose.yml. Для этого необходимо ознакомиться со значениями каждой из них. Значения переменных у сервисов api и sidekiq идентичны.

|  |  |
| --- | --- |
| RAILS\_ENV | Системная переменная Rails проекта. Для работы в режиме продакшн значение production |
| RAILS\_HOSTNAME | Адрес, на котором доступен API сервис. Формат: host |
| CORS\_ORIGIN\_FRONT | Доменное имя фронт приложение портала. Формат example1.com, example2.com |
| RAILS\_MASTER\_KEY | Ключ для расшифровки зашифрованных параметров конфигурации |
| SIDEKIQ\_USERNAME | Имя пользователя для интерфейса администратора сервиса Sidekiq |
| SIDEKIQ\_PASSWORD | Пароль для интерфейса администратора сервиса Sidekiq |
| REVIZOR\_API\_DATABASE\_HOST | Адрес сервера базы данных Postgres. Формат: host или IP |
| REVIZOR\_API\_DATABASE\_PASSWORD | Пароль базы данных Postgres |
| REVIZOR\_API\_DATABASE\_USER | Имя пользователя базы данных Postgres |
| REDIS\_URL | Адрес сервера базы данных Redis. Формат redis://host:port |
| ELASTICSEARCH\_URL | Адрес сервера Elasticsearch. Формат: http(s)://host:port |
| FRONTEND\_URL | Доменное имя фронт приложение администрирования. Формат: http(s)://example.com |
| SENDGRID\_API\_KEY | Токен доступа сервиса отправки почты Sengrid |
| AWS\_BUCKET\_NAME | Имя бакета S3 |
| AWS\_ACCESS\_KEY\_ID | ID ключа доступа к бакету S3 |
| AWS\_SECRET\_ACCESS\_KEY | Ключ доступа к бакету S3 |

Переменные SIDEKIQ\_USERNAME и SIDEKIQ\_PASSWORD задаете самостоятельно, какие удобно вам. С помощью них можно будет зайти на страницу мониторинга по адресу /api/sidekiq.

# Запуск системы

Для запуска контейнеров необходимо в директории **/docker\_stack** и выполнить команду:

$ cd /docker\_stack

$ docker-compose up -d

Для проверки работоспособности запущенных сервисов используется следующая команда:

$ docker logs -ft <service\_name>

В случае успешного запуска в консоль будут выведены логи.

Для контейнера api:

Current default time zone: 'Etc/UTC'

Local time is now: Mon Nov 16 17:42:36 UTC 2020.

Universal Time is now: Mon Nov 16 17:42:36 UTC 2020.

Container timezone set to: Europe/Moscow

Puma starting in single mode...

\* Version 4.3.5 (ruby 2.6.3-p62), codename: Mysterious Traveller

\* Min threads: 5, max threads: 5

\* Environment: production

\* Listening on tcp://0.0.0.0:3000

Use Ctrl-C to stop

Для контейнера sidekiq:

2020-11-05T12:01:40.183Z pid=1 tid=1lbx INFO: Booting Sidekiq 6.0.7 with redis options {:url=>"redis://redis:6379"}

pid=1 tid=1lbx INFO: Booted Rails 6.0.3.2 application in production environment

2020-11-05T12:01:41.677Z pid=1 tid=1lbx INFO: Running in ruby 2.7.1p83 (2020-03-31 revision a0c7c23c9c) [x86\_64-linux]

2020-11-05T12:01:41.677Z pid=1 tid=1lbx INFO: See LICENSE and the LGPL-3.0 for licensing details.

Для контейнера front\_admin:

> revizor\_admin@1.0.0 start /usr/front

> nuxt-ts start

---------------CONFIG---------------

API\_URL: https://revizor.tanto-s.ru/api

HOST: 0.0.0.0

PORT: 3000

WS\_URL: wss://revizor.tanto-s.ru/cable

ℹ Listening on: http://172.18.0.3:3000/

Для контейнера revizor-app:

> revizor\_front@1.0.0 start /usr/front

> NUXT\_HOST=0.0.0.0 NUXT\_PORT=3000 nuxt-ts start

---------------CONFIG---------------

API\_URL: https://revizor.tanto-s.ru/api

IMG\_URL: https://revizor.tanto-s.ru

HOST: 0.0.0.0

PORT: 3000

YMAPS: 09737f4e-ca92-46df-91d0-f358dea41b0f

ℹ Listening on: http://172.18.0.2:3000/

В случае неуспешного запуска в логе будут перечислены причины неуспешного запуска контейнера.