КРАММЕРТИ

ИНСТРУКЦИЯ ПО УСТАНОВКЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Программа для ЭВМ

«Вектор города»

1. Введение

Программа для ЭВМ «Вектор города» предназначена для централизованного мониторинга городской инфраструктуры, отображения объектов на карте, просмотра уведомлений, аналитических показателей, камер видеонаблюдения и данных от контроллеров.

Программное обеспечение устанавливается на сервере Заказчика и предоставляет веб-интерфейс для пользователей через браузер. В составе решения используются веб-приложение, база данных PostgreSQL, реверс-прокси Nginx, сервер потокового видео WebRTC/RTSP и сервис чтения показаний датчиков.

2. Требования к аппаратному и системному программному обеспечению

Минимальные требования к серверу для обзорного развертывания:

Компонент	Минимальное значение
Процессор	4 ядра
Оперативная память	8 Гб
Диск	SSD от 128 Гб
Сеть	100 Мбит/с и выше
ОС	Debian 11/12 или Ubuntu Server 22.04/24.04

Рекомендуемые компоненты:

– PostgreSQL 15 или выше;
– Node.js 22 LTS и npm;
– Docker и Docker Compose plugin;
– Nginx;
– Python 3.10 или выше;
– systemd для управления сервисами.

Пользовательское рабочее место должно иметь современный браузер: Yandex Browser, Chromium, Google Chrome, Microsoft Edge или Firefox.

3. Состав программного обеспечения

В состав установки входят следующие части:

1. Веб-приложение «Вектор города» на Next.js - основной интерфейс системы.
2. PostgreSQL - хранение пользователей, объектов, событий, измерений и тревог.
3. Nginx - прием HTTP/HTTPS-запросов и проксирование к приложению.
4. WebRTC/RTSP-сервер - прием RTSP-потоков камер и выдача видео в веб-интерфейс.
5. Сервис чтения датчиков - опрос контроллера и запись показаний в PostgreSQL.

Для базового развертывания все компоненты могут быть установлены на один сервер. При росте нагрузки допускается перенос PostgreSQL, WebRTC-сервера или сервисов интеграции на отдельные узлы.

4. Развертывание программного обеспечения на сервере Заказчика

Пример ниже приведен для установки на одном сервере от имени пользователя с административными правами. Значения в угловых скобках необходимо заменить на параметры конкретной установки.

4.1. Подготовка сервера

Установите базовые пакеты:

```
apt-get update
apt-get install -y curl ca-certificates git nginx postgresql postgresql-contrib python3 python3-venv docker.io
  docker-compose-plugin
```

Создайте каталог установки:

```
mkdir -p /opt/vector-city
```

4.2. Настройка PostgreSQL

Создайте рабочую базу данных и пользователя:

```
sudo -u postgres psql
CREATE DATABASE vector_city;
CREATE USER vector_city_user WITH PASSWORD '<DB_PASSWORD>';
GRANT ALL PRIVILEGES ON DATABASE vector_city TO vector_city_user;
\q
```

Минимально для приема показаний датчиков в базе должны быть подготовлены таблицы измерений и тревог:

```
CREATE TABLE IF NOT EXISTS measurements (
  id BIGSERIAL PRIMARY KEY,
  created_at TIMESTAMPTZ NOT NULL DEFAULT now(),
  element INTEGER NOT NULL,
  address INTEGER NOT NULL,
  category TEXT NOT NULL,
  value DOUBLE PRECISION,
  alarm TEXT
);

CREATE TABLE IF NOT EXISTS controller_alerts (
  id BIGSERIAL PRIMARY KEY,
  created_at TIMESTAMPTZ NOT NULL DEFAULT now(),
  module_name TEXT NOT NULL,
  element INTEGER NOT NULL,
  address INTEGER NOT NULL,
  category TEXT NOT NULL,
  value DOUBLE PRECISION,
  alarm TEXT,
  prev_alarm TEXT,
  message TEXT
);
```

4.3. Установка веб-приложения «Вектор города»

Распакуйте или склонируйте дистрибутив приложения в каталог:

```
cd /opt/vector-city
git clone <REPOSITORY_URL> app
cd /opt/vector-city/app
npm ci
```

Создайте файл окружения:

```
vi /opt/vector-city/app/.env
```

Пример основных параметров:

```
NODE_ENV=production
PORT=3000
HOSTNAME=127.0.0.1
DATABASE_URL=postgresql://vector_city_user:<DB_PASSWORD>@127.0.0.1:5432/vector_city
NEXT_PUBLIC_APP_NAME=Вектор города
```

Соберите приложение:

```
npm run build
```

Создайте systemd-сервис:

```
[Unit]
Description=Vector City web application
After=network.target postgresql.service

[Service]
Type=simple
WorkingDirectory=/opt/vector-city/app
EnvironmentFile=/opt/vector-city/app/.env
ExecStart=/usr/bin/npm run start -- --hostname 127.0.0.1 --port 3000
Restart=always
RestartSec=5

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

Сохраните файл как /etc/systemd/system/vector-city.service и включите сервис:

```
systemctl daemon-reload
systemctl enable vector-city
systemctl start vector-city
```

4.4. Настройка Nginx

Создайте конфигурацию /etc/nginx/sites-available/vector-city.conf:

```
server {
  listen 80;
  server_name <SERVER_DOMAIN>;

  location = /healthz {
    add_header Content-Type text/plain;
    return 200 "ok\n";
  }

  location / {
    proxy_http_version 1.1;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
    proxy_pass http://127.0.0.1:3000;
  }
}
```

Активируйте конфигурацию:

```
ln -sfn /etc/nginx/sites-available/vector-city.conf /etc/nginx/sites-enabled/vector-city.conf
nginx -t
systemctl restart nginx
```

4.5. Установка WebRTC/RTSP-сервера

Распакуйте архив с WebRTC-сервером:

```
mkdir -p /opt/vector-city/webrtc
unzip rtsp-webrtc-master.zip -d /opt/vector-city/webrtc
cd /opt/vector-city/webrtc/rtsp-webrtc-master
```

В файле mediamtx.yml укажите публичный IP-адрес или доменное имя сервера:

```
webrtc: yes
webrtcAddress: :8889
webrtcEncryption: no
webrtcAllowOrigins: ['*']
webrtcAdditionalHosts:
  - <SERVER_PUBLIC_IP_OR_DOMAIN>

paths:
  cam1:
    source: <RTSP_URL>
    sourceOnDemand: no
```

Запустите контейнер:

```
docker compose up -d --build
docker compose ps
```

Для работы видеопотоков должны быть доступны необходимые порты: 8554, 8888, 8889, 8890, 9997 и 8189/udp.

4.6. Установка сервиса чтения датчиков

Создайте каталог сервиса:

```
mkdir -p /opt/vector-city/sensor-reader
cd /opt/vector-city/sensor-reader
python3 -m venv .venv
. .venv/bin/activate
pip install requests psycopg2-binary
```

Сервис опрашивает контроллер по HTTP, получает измерения, сохраняет новые значения в таблицу measurements и при изменении статуса тревоги пишет событие в controller_alerts.

Создайте файл окружения /opt/vector-city/sensor-reader/.env:

```
DEVICE_BASE_URL=http://<DEVICE_IP>
DEVICE_USER=<DEVICE_USER>
DEVICE_PASSWORD=<DEVICE_PASSWORD>
POLL_SEC=10
DATABASE_URL=postgresql://vector_city_user:<DB_PASSWORD>@127.0.0.1:5432/vector_city
```

Создайте systemd-сервис /etc/systemd/system/vector-city-sensor-reader.service:

```
[Unit]
Description=Vector City sensor reader
After=network.target postgresql.service

[Service]
Type=simple
WorkingDirectory=/opt/vector-city/sensor-reader
EnvironmentFile=/opt/vector-city/sensor-reader/.env
ExecStart=/opt/vector-city/sensor-reader/.venv/bin/python /opt/vector-city/sensor-reader/sensor_reader.py
Restart=always
RestartSec=10

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

Включите сервис:

```
systemctl daemon-reload
systemctl enable vector-city-sensor-reader
systemctl start vector-city-sensor-reader
```

5. Проверка запуска

После установки проверьте состояние сервисов:

```
systemctl status postgresql
systemctl status nginx
systemctl status vector-city
systemctl status vector-city-sensor-reader
docker compose -f /opt/vector-city/webrtc/rtsp-webrtc-master/docker-compose.yml ps
```

Проверьте доступность веб-приложения:

```
curl -I http://<SERVER_DOMAIN>/healthz
curl -I http://<SERVER_DOMAIN>/
```

Проверьте поступление измерений:

```
sudo -u postgres psql -d vector_city -c "SELECT created_at, element, address, category, value, alarm FROM
  measurements ORDER BY created_at DESC LIMIT 10;"
```

Проверьте журнал сервиса датчиков:

```
journalctl -u vector-city-sensor-reader -n 50 --no-pager
```

После успешной проверки пользователи могут начать работу с системой через браузер по адресу:

```
http://<SERVER_DOMAIN>
```