Описание состава экземпляра и порядок развёртывания ПО ПМСО

Правообладатель: ООО «Платформа», ИНН 5902054490. **Программное обеспечение:** ПО ПМСО (ПО платформа мониторинга систем оповещения).

1. Назначение документа

ПО ПМСО предоставляется как интернет-сервис (SaaS). Для экспертной проверки используется способ «удалённый доступ к инфраструктуре с развёрнутым экземпляром ПО» (разд. 2.1.2.3 методических рекомендаций по работе с ФГИС «Реестры ПО»). Документ содержит:

- описание состава развёрнутого экземпляра: процессы, контейнеры, расположение файлов и компонентов (разд. 2);
- пошаговый порядок развёртывания экземпляра на чистом сервере (разд. 4);
- порядок обновления и резервного копирования (разд. 5);
- контакты технической поддержки (разд. 6).

2. Состав экземпляра

Экземпляр запускается набором из четырёх контейнеров Docker Compose на одном сервере. Описание составлено по файлу `docker-compose.yml` в корне репозитория.

Таблица 1 — Контейнеры экземпляра

Контейнер	Образ или сборка	Назначение	Порты
`db`	`postgres:16`	База данных PostgreSQL (БД `rson_monitoring`, пользователь `rson_app`)	только внутренняя сеть
`backend`	сборка из `backend/Dockerfile` (Python 3.11)	Серверная часть: FastAPI, планировщик опроса Zabbix, миграции БД	8000, только внутренняя сеть
`web`	сборка из `deploy/nginx/Dockerfile` (Node.js 20 для сборки клиента, затем nginx)	Веб-сервер: отдаёт собранный клиент (React), проксирует запросы к серверной части, принимает HTTPS	80, 443 наружу
`certbot`	`certbot/certbot`	Выпуск и продление	не публикуются

Контейнер	Образ или сборка	Назначение	Порты
		сертификата Let's Encrypt (проверка продления каждые 12 часов)	

Таблица 2 — Процессы внутри контейнера backend

Процесс	Что делает
entrypoint.sh	Ждёт доступности БД (до 30 попыток с паузой 1 с), выполняет alembic upgrade head (создание и обновление таблиц), запускает uvicorn app.main:app на порту 8000
Планировщик APScheduler	Внутри процесса uvicorn, раз в SYNC_INTERVAL_MINUTES минут (по умолчанию 5) опрашивает Zabbix по API, обновляет статусы объектов и историю событий

Таблица 3 — Расположение файлов и данных

Что	Где	Хранение
Код на сервере	/opt/rson-monitoring/	каталог на диске сервера
Настройки серверной части	/opt/rson-monitoring/backend/.env.prod	файл на сервере, права 600, в репозиторий не входит
Настройки сборки (пароль БД, ключ карты)	/opt/rson-monitoring/.env	файл на сервере, права 600, в репозиторий не входит
Данные PostgreSQL	том Docker pgdata, внутри контейнера /var/lib/postgresql/data	том Docker
Фото к заявкам	том Docker ticket_uploads, внутри контейнера backend /app/uploads	том Docker
Сертификаты HTTPS	том Docker certbot_conf, внутри /etc/letsencrypt	том Docker
Резервные копии БД	/var/backups/rson/ на сервере и копия во внешнем хранилище	см. разд. 5

Таблица 4 — Параметры серверной части (файл backend/.env.prod)

Параметр	Назначение
DATABASE_URL	Строка подключения к PostgreSQL

Параметр	Назначение
`ZABBIX_URL`, `ZABBIX_API_TOKEN`, `ZABBIX_USER`, `ZABBIX_PASSWORD`	Подключение к внешней системе мониторинга Zabbix (источник данных)
`SECRET_KEY`	Ключ подписи токенов доступа
`SMTP_HOST`, `SMTP_PORT`, `SMTP_USER`, `SMTP_PASSWORD`, `SMTP_FROM`	Отправка писем со ссылками для входа
`FRONTEND_URL`	Адрес сайта, подставляется в письма
`PRODUCT_NAME`, `OPERATOR_LABEL`	Название продукта и подпись оператора в интерфейсе и письмах
`SYNC_INTERVAL_MINUTES`	Интервал опроса Zabbix (по умолчанию 5)

Значения параметров в документ не включаются.

Таблица 5 — Внешние сервисы

Сервис	Назначение	Направление
Zabbix (внешняя система мониторинга)	Источник данных о состоянии объектов	исходящие запросы по API
Почтовый сервер SMTP	Отправка ссылок для входа	исходящие
Let’s Encrypt	Выпуск сертификата HTTPS	исходящие
Тайлы карты	Подложка карты в браузере пользователя	запросы из браузера

3. Требования к серверу

- Операционная система: Ubuntu 24.04 LTS (на ней проверен порядок развёртывания).
- Docker и Docker Compose v2 (пакеты Ubuntu `docker.io` и `docker-compose-v2`).
- Открытые входящие порты: 22 (SSH, по ключу), 80, 443.
- Для сборки клиента на сервере с 4 ГБ оперативной памяти используется swap-файл 2 ГБ (создаётся скриптом первичной настройки).
- Доменное имя с A-записью на IP сервера (в экземпляре: `rscon.platformsw.ru`). Имя домена задано в файлах `deploy/nginx/nginx.conf` и `deploy/vds/first_cert.sh`; при развёртывании под другим именем заменить в этих двух файлах.
- Параметры боевого сервера: 2 vCPU, 4 ГБ оперативной памяти, диск 60 ГБ NVMe, Ubuntu 24.04.

4. Пошаговое развёртывание на чистом сервере

1. **Первичная настройка сервера** (один раз, под root). Скрипт `deploy/vds/setup_vds.sh` создаёт пользователя `deploy`, отключает вход по паролю, ставит Docker и Compose, включает файрвол (22, 80, 443), создаёт swap 2 ГБ, включает автоматические обновления безопасности, создаёт каталог

/opt/rson-monitoring: bash setup_vds.sh authorized_keys.txt, где файл содержит открытые SSH-ключи администратора.

2. **Загрузка кода** в /opt/rson-monitoring/ под пользователем deploy: rsync из репозитория или git clone.
3. **Файлы настроек.** Создать backend/.env.prod (параметры из таблицы 4, DATABASE_URL с хостом db, порт 5432) и корневой .env с переменными DB_PASSWORD и VITE_YANDEX_TILES_API_KEY. Права на оба файла: chmod 600.
4. **Запуск базы данных:** docker compose up -d db. Дождаться состояния healthy: docker compose ps.
5. **Сертификат HTTPS.** После того как A-запись домена указывает на сервер: bash deploy/vds/first_cert.sh <почта>. Скрипт временно занимает порт 80, выпускает сертификат Let's Encrypt и кладёт его в том certbot_conf.
6. **Запуск всех контейнеров:** docker compose up -d --build. При запуске backend автоматически применяет миграции БД и создаёт таблицы.
7. **Первый администратор** (пользователей через интерфейс создаёт только администратор, поэтому первый создаётся командой): docker compose exec backend python scripts/create_admin.py --username admin --email <почта> --full-name "<ФИО>". Скрипт печатает одноразовую ссылку для входа и пытается отправить её на почту.
8. **Проверка:**
   - curl -s https://<домен>/health возвращает {"status":"ok"};
   - страница https://<домен>/ открывается, вход выполняется по ссылке из письма.

5. Обновление и резервное копирование

Обновление. Выполняется при каждом изменении кода в ветке main (автоматически через GitVerse CI/CD, файл .gitverse/workflows/deploy.yaml): копирование файлов на сервер командой rsync --delete (без файлов настроек и каталога .git) и выполнение docker compose up -d --build. Вручную то же самое: загрузить новый код в /opt/rson-monitoring/ и выполнить docker compose up -d --build на сервере. Миграции БД применяются при запуске backend.

Резервное копирование БД. Скрипт deploy/vds/backup_db.sh снимает дамп PostgreSQL (pg_dump -Fc) в /var/backups/rson/, проверяет, что дамп читается, хранит 7 последних копий и при наличии файла /etc/rson-backup.env выгружает копию во внешнее закрытое хранилище. Ежедневный запуск в 03:30 включается командой sudo bash deploy/vds/install_backup_timer.sh (systemd-таймер). Проверка восстановления: deploy/vds/restore_check.sh.

Восстановление. Дамп восстанавливается в пустую БД командой docker compose exec -T db pg_restore -U rson_app -d rson_monitoring --no-owner --no-acl < <файл дампа>. Скрипт restore_check.sh проверяет это на отдельном временном контейнере PostgreSQL 16, не затрагивая рабочую базу.

6. Контакты

Техническая поддержка и консультации по развёртыванию и настройке: info@platformsw.ru, с 9:00 до 18:00 по времени г. Перми.