Руководство администратора «Л1 Первая Линия Техподдержки»

Руководство администратора «Л1 Первая Линия Техподдержки»

Содержание

Документ	Назначение
Системные требования	Состав системы, ресурсы, платформа, диски и сетевые доступы.
Развёртывание	Подготовка сервера, установка в открытом или закрытом контуре и первоначальная проверка.
Конфигурация	Правила работы с `.env` и справочник переменных окружения.
Эксплуатация	Запуск и остановка, журналы, резервное копирование, обновление, откат и диагностика.

Быстрый маршрут

1. Проверьте системные и сетевые требования.
2. Выберите способ получения образов: из Container Registry или из архивов закрытого контура.
3. Выполните установку.
4. Сохраните `.env`, `APP_SECRETS_KEY` и сведения о версиях в защищённом хранилище.
5. Настройте резервное копирование PostgreSQL и MinIO по руководству по эксплуатации.

Критерии успешного развёртывания

Развёртывание завершено, если:

- сервисы `postgres`, `redis`, `minio`, `backend` и `frontend` имеют состояние `Up` и `healthy`;
- одноразовые сервисы `migrate` и `minio-mc` завершились с кодом `0`;
- веб-интерфейс открывается по `FRONTEND_PUBLIC_URL`;
- пользовательский компьютер может загрузить и скачать тестовый файл через `S3_PUBLIC_ENDPOINT`;

Инструкция рассчитана на администратора, знакомого с Linux, Docker, Docker Compose, DNS и reverse proxy.

Системные требования

Требования ниже относятся к односерверному развёртыванию.

Состав развёртывания

Сервис	Назначение	Постоянные данные
`frontend`	Веб-интерфейс и проксирование запросов к backend.	Нет.
`backend`	API, бизнес-логика и фоновые обработчики.	Нет; конфигурация и файловые журналы монтируются с хоста.
`postgres`	Основные данные системы.	Да, Docker volume `postgres_data`.
`redis`	Кэш и временная координация фоновых задач.	Нет, данные считаются временными.
`minio`	Файлы и вложения.	Да, Docker volume `minio_data`.
`migrate`	Применение миграций PostgreSQL перед запуском backend.	Нет, одноразовый сервис.
`minio-mc`	Создание бакета и технического пользователя MinIO.	Нет, одноразовый сервис.

Все сервисы по умолчанию запускаются на одном Docker-сервере в общей bridge-сети.

Аппаратные ресурсы

Контур	CPU	RAM	Системный диск	Диск данных
Демонстрация и функциональное тестирование	2 vCPU	4 ГБ	20–30 ГБ SSD	от 20 ГБ
Начальная промышленная конфигурация	4 vCPU	8 ГБ	40–60 ГБ SSD	от 100 ГБ
Повышенная нагрузка	8 vCPU	16 ГБ	80–100 ГБ SSD/NVMe	от 250 ГБ
Высокая нагрузка или повышенные требования к доступности	По результатам нагрузочного тестирования	от 32 ГБ	NVMe	отдельные PostgreSQL и S3-хранилище

Это предварительные ориентиры, а не гарантированные показатели производительности. Итоговая конфигурация зависит от:

- числа одновременно работающих пользователей;
- количества создаваемых обращений и сообщений;
- интенсивности IMAP/SMTP-обмена;
- количества и размера вложений;
- срока хранения файлов и резервных копий;
- требований к времени отклика и восстановлению после сбоя.

Программная платформа

- 64-разрядный Linux;
- архитектура хоста, совпадающая с архитектурой образов поставки; базовый целевой вариант — `amd64`;
- Docker Engine 24 или новее;
- Docker Compose Plugin v2.20 или новее; Compose v1 не поддерживается;
- включённый автозапуск Docker после перезагрузки сервера;
- `curl` и `openssl` для проверки установки и генерации секретов;
- корректная синхронизация времени через NTP.

Рекомендуемые базовые дистрибутивы: минимальная установка Ubuntu Server 22.04/24.04 LTS или Debian 12 без графической оболочки.

> Применение Astra Linux, ALT Linux, RED OS и других дистрибутивов требует предварительной проверки совместимости конкретной версии Docker Engine и образов поставки.

Проверьте окружение перед установкой

```
uname -m
docker version
docker compose version
systemctl is-enabled docker
timedatectl status
```

Для рабочих мест рекомендуется актуальная версия Chromium-based браузера или Firefox ESR с включёнными JavaScript, cookies и доступом к обоим публичным адресам системы.

Входящие соединения

В промышленной среде снаружи публикуются только HTTPS-адреса веб-интерфейса и S3 API.

Откуда	Куда	Порт	Назначение
Рабочие места или reverse proxy	frontend	`443/TCP` снаружи → `9090/TCP` на хосте	Веб-интерфейс.

Откуда	Куда	Порт	Назначение
Рабочие места или reverse proxy	MinIO S3 API	443/TCP снаружи → 9104/TCP на хосте	Файлы и вложения.
Только Docker-сервер	backend	9100/TCP на localhost	Диагностика API.
Только Docker-сервер	PostgreSQL	9101/TCP на localhost	Администрирование и диагностика БД.
Только Docker-сервер	Redis	9102/TCP на localhost	Диагностика Redis.
Только Docker-сервер	MinIO Console	9103/TCP на localhost	Администрирование MinIO.

Порты 9100, 9101, 9102 и 9103 не должны быть доступны из внешней сети. При временном запуске без reverse proxy рабочим местам нужны прямые доступы к 9090/TCP и 9104/TCP.

Исходящие соединения

Откуда	Куда	Порт	Когда требуется
Docker-сервер	Container Registry	443/TCP или порт registry, например 5050/TCP	Загрузка образов в контуре с registry.
Docker-сервер и backend	DNS-серверы	53/UDP, при необходимости 53/TCP	Разрешение имён.
Docker-сервер	NTP-сервер	123/UDP	Синхронизация времени.
Backend	LDAP/LDAPS	389/636 TCP	Аутентификация через LDAP/AD.
Backend	IMAP/IMAPS	143/993 TCP	Получение входящей почты.
Backend	SMTP	25/465/587 TCP	Отправка почты.
Backend	DaData	443/TCP	Подсказки организаций при включённой интеграции.
Рабочие места	Публичный S3 endpoint	443/TCP	Загрузка и скачивание файлов.

Если используется LDAP/AD, контейнеру backend должны быть доступны корпоративный DNS, контроллеры домена и требуемые LDAP/LDAPS-порты.

DNS и TLS

Для промышленной установки подготовьте два DNS-имени, например:

- `l1.example.org` — веб-интерфейс;
- `s3.l1.example.org` — S3 API.

Оба адреса должны быть доступны с рабочих мест и защищены действующими TLS-сертификатами. Внутреннее имя контейнера `minio` нельзя использовать как публичный S3 endpoint.

Reverse proxy должен разрешать загрузку файлов настроенного максимального размера, передавать заголовки `Host`, `X-Forwarded-For`, `X-Forwarded-Proto` и поддерживать долгоживущие HTTP-соединения приложения.

Развёртывание

Перед установкой ознакомьтесь с системными требованиями и подготовьте DNS, сетевые доступы и диски.

1. Получение комплекта поставки

Комплект содержит два каталога верхнего уровня:

```
l1-support
├── docker-compose.yaml
├── .env.example
├── config/
└── logs/
```

Также должны быть предоставлены точные версии образов backend и frontend. Для закрытого контура комплект должен включать архивы всех образов из `l1-support/docker-compose.yaml`, включая PostgreSQL, Redis, MinIO и MinIO Client.

2. Подготовка каталога

Распакуйте ZIP, затем скопируйте содержимое каталога `l1-support` в `/opt/l1-support` или укажите свой путь:

```
unzip /path/to/l1-support-release.zip -d /tmp/l1-support-release
sudo install -d -o "$USER" -g "$(id -gn)" /opt/l1-support
cp -a /tmp/l1-support-release/l1-support/. /opt/l1-support/
cd /opt/l1-support

sudo install -d -o 10001 -g 10001 -m 0750 config logs
sudo chown -R 10001:10001 config logs
```

При первой установке создайте `.env` из шаблона и ограничьте доступ к нему:

```
cp .env.example .env
chmod 600 .env
```

При обновлении существующий `.env` не заменяется шаблоном из новой поставки. Новые и изменённые параметры нужно переносить вручную после сравнения файлов.

3. Настройка окружения

Полный перечень параметров приведён в справочнике конфигурации. До первого запуска обязательно настройте следующие группы в `.env`.

Версии образов

Укажите пути и точные неизменяемые версии, полученные с поставкой:

```
BACKEND_IMAGE=registry.legionsecurity.ru/l1-support/l1-server
BACKEND_VERSION=v1.2.3 (получить у поставщика)
FRONTEND_IMAGE=registry.legionsecurity.ru/l1-support/l1-client
FRONTEND_VERSION=v1.2.3 (получить у поставщика)
```

Тег `latest` в промышленной установке не используется.

Публичные адреса

Для временного запуска без reverse proxy:

```
FRONTEND_PUBLIC_URL=http://192.168.1.10:9090
S3_PUBLIC_ENDPOINT=http://192.168.1.10:9104
```

Для промышленной установки:

```
FRONTEND_PUBLIC_URL=https://l1.example.org
S3_PUBLIC_ENDPOINT=https://s3.l1.example.org
```

`S3_PUBLIC_ENDPOINT` должен быть доступен с рабочих мест пользователей.

Секреты

Сгенерируйте независимые значения для каждого секрета:

```
printf 'JWT_SECRET=%s\n' "$(openssl rand -hex 32)"
printf 'APP_SECRETS_KEY=%s\n' "$(openssl rand -base64 32)"
printf 'DB_PASSWORD=%s\n' "$(openssl rand -hex 24)"
printf 'REDIS_PASSWORD=%s\n' "$(openssl rand -hex 24)"
printf 'MINIO_ROOT_PASSWORD=%s\n' "$(openssl rand -hex 24)"
printf 'S3_SECRET_KEY=%s\n' "$(openssl rand -hex 24)"
```

Перенесите значения в `.env`. Сохраните `.env` и `APP_SECRETS_KEY` в защищённом хранилище: потеря ключа сделает ранее зашифрованные настройки интеграций недоступными.

Корпоративный DNS

Если используется LDAP/AD:

```
DOCKER_DNS_SERVER=192.168.20.100
DOCKER_DNS_SEARCH=corp.example.org
```

Указанный DNS-сервер и контроллеры домена должны быть доступны с Docker-сервера и из контейнера backend.

Проверка конфигурации

```
grep -n 'CHANGE_ME' .env
ENVFILE=.env docker compose --env-file .env config --quiet
```

Строки в выводе grep означают, что соответствующие параметры необходимо заполнить. Команда `docker compose config` не должна возвращать ошибок.

4. Загрузка образов

Сервер с доступом к Container Registry

Если на сервере ещё нет действующих учётных данных, выполните авторизацию в Container Registry:

> Учётные данные для доступа можно получить у поставщика.

```
docker login registry.legionsecurity.ru
```

Загрузите образы приложения:

```
docker compose pull
```

Повторная авторизация перед каждым запуском не требуется. Выполните `docker login` повторно, только если текущие учётные данные перестали действовать.

Закрытый контур

Передайте полученные от поставщика архивы образов на сервер разрешённым способом и загрузите каждый архив:

```
for archive in /path/to/images/*.tar; do
  docker load --input "$archive"
done
```

Убедитесь, что загружены все образы и их теги совпадают со значениями в `.env` и `docker-compose.yaml`:

```
docker image ls
docker compose config --images
```

В закрытом контуре не выполняйте `docker compose pull`. Если хотя бы один необходимый образ отсутствует, запуск завершится попыткой обращения к registry или ошибкой получения образа.

5. Запуск

```
docker compose up -d
docker compose ps -a
```

Ожидаемое состояние:

Сервис	Статус
`postgres`, `redis`, `minio`, `backend`, `frontend`	`Up` и `healthy`.
`migrate`, `minio-mc`	`Exited (0)`.

`migrate` и `minio-mc` являются одноразовыми сервисами. Их завершение с кодом `0` является штатным.

Если запуск не завершился успешно:

```
docker compose ps -a
docker compose logs --tail=200 migrate minio-mc backend
```

После исправления причины повторите `docker compose up -d`.

6. Проверка на Docker-сервере

```
curl -f http://127.0.0.1:9100/api/v1/system/bootstrap/status
curl -fI http://127.0.0.1:9090
curl -f http://127.0.0.1:9104/minio/health/live
```

Если локальные проверки проходят, а система недоступна с рабочего места, проверьте DNS, межсетевой экран, маршрутизацию и reverse proxy.

7. Публикация через HTTPS

Опубликуйте два адреса:

Публичный адрес	Адрес назначения
`https://l1.example.org`	`http://127.0.0.1:9090`
`https://s3.l1.example.org`	`http://127.0.0.1:9104`

Reverse proxy должен:

- использовать действующие TLS-сертификаты;
- передавать `Host`, `X-Forwarded-For` и `X-Forwarded-Proto`;
- разрешать загрузку файлов размером не меньше `FILES_MAX_SIZE_BYTES`;
- поддерживать долгоживущие соединения приложения;

- не публиковать порты `9100`, `9101`, `9102` и `9103`.

Публичные адреса должны в точности совпадать со значениями `FRONTEND_PUBLIC_URL` и `S3_PUBLIC_ENDPOINT`.

8. Первоначальная настройка

1. Откройте `FRONTEND_PUBLIC_URL` в браузере.
2. Создайте первого администратора на начальной странице.
3. Войдите в систему.
4. Загрузите и скачайте тестовый файл.
5. Настройте почту и используемые провайдеры аутентификации.
6. Проверьте внешнее резервное копирование PostgreSQL и MinIO.

Дальнейшие процедуры приведены в руководстве по эксплуатации.

Конфигурация

Основная конфигурация развёртывания находится в файле `.env`. Создавайте его только из `.env.example` той же версии поставки.

Правила работы с `.env`

- Используйте разные случайные значения для всех паролей и ключей.
- Не меняйте APP_SECRETS_KEY после первоначальной настройки без отдельной процедуры ротации.
- Не используйте тег образа `latest` в промышленной среде.
- Перед запуском проверяйте конфигурацию командой `ENVFILE=.env docker compose --env-file .env config --quiet`.
- При обновлении сравнивайте новый `.env.example` с действующим `.env`; не заменяйте рабочий файл целиком.

Каталог `config` монтируется в `/app/config`, а `logs` — в `/app/logs`. Оба каталога должны быть доступны пользователю контейнера с UID/GID `10001:10001`.

Приложение

Переменная	Назначение	Пример
BACKEND_IMAGE	Образ серверной части.	registry.legionsecurity.ru/l1-support/l1-server
BACKEND_VERSION	Точная версия серверной части.	v1.2.3
FRONTEND_IMAGE	Образ клиентской части.	registry.legionsecurity.ru/l1-support/l1-client
FRONTEND_VERSION	Точная версия клиентской части.	v1.2.3
SERVICE_NAME	Имя сервиса в журналах.	l1-support
APP_ENV	Режим работы.	prod
SERVER_ADDRESS	Адрес прослушивания внутри контейнера.	0.0.0.0
SERVER_PORT	Внутренний порт backend.	3000
SERVER_HOST_PORT	Диагностический порт backend на localhost.	9100
FRONTEND_HOST_PORT	Порт веб-интерфейса на хосте.	9090
FRONTEND_PUBLIC_URL	Публичный адрес веб-интерфейса.	https://l1.example.org

Переменная	Назначение	Пример
DOCKER_DNS_SERVER	Корпоративный DNS-сервер для контейнера backend. Используется для LDAP/AD	192.168.20.100
DOCKER_DNS_SEARCH	Поисковый домен DNS.	corp.example.org

PostgreSQL

Переменная	Назначение	Пример
DB_TYPE	Тип базы данных.	postgres
DB_HOST	Имя сервиса PostgreSQL.	postgres
DB_PORT	Внутренний порт PostgreSQL.	5432
DB_USER	Пользователь базы данных.	l1_support
DB_PASSWORD	Пароль пользователя.	<случайный_пароль>
DB_NAME	Имя базы данных.	l1_support
DB_HOST_PORT	Диагностический порт PostgreSQL на localhost.	9101

Redis

Переменная	Назначение	Пример
REDIS_HOST	Имя сервиса Redis.	redis
REDIS_PORT	Внутренний порт Redis.	6379
REDIS_HOST_PORT	Диагностический порт Redis на localhost.	9102
REDIS_PASSWORD	Пароль Redis.	<случайный_пароль>
REDIS_DB	Номер логической базы.	0
REDIS_MAX_RETRIES	Число повторов подключения.	3
REDIS_DIAL_TIMEOUT	Тайм-аут подключения.	5s
REDIS_READ_TIMEOUT	Тайм-аут чтения.	3s
REDIS_WRITE_TIMEOUT	Тайм-аут записи.	3s

MinIO и S3

Переменная	Назначение	Пример
MINIO_ROOT_USER	Администратор MinIO.	l1-minio-admin

Переменная	Назначение	Пример
MINIO_ROOT_PASSWORD	Пароль администратора MinIO.	<случайный_пароль>
MINIO_API_HOST_PORT	Порт S3 API на хосте.	9104
MINIO_CONSOLE_HOST_PORT	Порт консоли MinIO на localhost.	9103
S3_BUCKET	Имя приватного бакета.	l1-files
S3_REGION	Регион S3.	us-east-1
S3_ACCESS_KEY	Технический пользователь backend.	l1-app
S3_SECRET_KEY	Пароль технического пользователя.	<случайный_пароль>
S3_ENDPOINT	Внутренний адрес MinIO для backend.	http://minio:9000
S3_PUBLIC_ENDPOINT	Публичный адрес S3 API для рабочих мест.	https://s3.l1.example.org
S3_USE_PATH_STYLE	Использование path-style адресов S3.	true

S3_ENDPOINT используется контейнером backend, а S3_PUBLIC_ENDPOINT — браузерами пользователей. Эти адреса не взаимозаменяемы.

Аутентификация и шифрование

Переменная	Назначение	Пример
JWT_SECRET	Секрет подписи JWT.	<случайный_секрет>
ACCESS_TOKEN_TTL	Время жизни access token.	5m
REFRESH_TOKEN_TTL	Время жизни refresh token.	720h
PASSWORD_RESET_TOKEN_TTL	Время жизни токена восстановления пароля.	30m
APP_SECRETS_KEY	Base64-ключ шифрования настроек.	результат openssl rand -base64 32
APP_TOTP_ISSUER	Название в приложении-аутентификаторе.	Л1 Первая Линия Техподдержки

Значение APP_SECRETS_KEY должно декодироваться в 32 байта. Его потеря или несогласованная замена сделает сохранённые зашифрованные пароли интеграций недоступными.

Файловый модуль

Переменная	Назначение	Пример
FILES_CONFIG_PATH		

Переменная	Назначение	Пример
	Путь к конфигурации файлов внутри контейнера.	`config/files.json`
`FILES_CONFIG_RELOAD_INTERVAL`	Интервал перечитывания конфигурации.	`30s`
`FILES_MAX_SIZE_BYTES`	Максимальный размер одного файла.	`500mb`
`FILES_MULTIPART_THRESHOLD_BYTES`	Порог multipart-загрузки.	`50mb`
`FILES_MULTIPART_PART_SIZE_BYTES`	Размер части multipart-загрузки.	`16mb`
`FILES_UPLOAD_URL_TTL_SECONDS`	Время жизни ссылки на загрузку.	`3600`
`FILES_DOWNLOAD_URL_TTL_SECONDS`	Время жизни ссылки на скачивание.	`300`
`FILES_CLEANUP_INTERVAL`	Интервал очистки незавершённых загрузок.	`20m`
`FILES_CLEANUP_BATCH_SIZE`	Размер пачки при очистке.	`100`

Ограничение размера запроса на reverse proxy должно быть не меньше `FILES_MAX_SIZE_BYTES`.

Интеграции и фоновые обработчики

Переменная	Назначение	Пример
`APPEARANCE_CONFIG_PATH`	Конфигурация внешнего вида.	`config/appearance.json`
`NOTIFICATION_TEMPLATES_CONFIG_PATH`	Шаблоны уведомлений.	`config/notification_templates.json`
`NOTIFICATION_TEMPLATES_RELOAD_INTERVAL`	Интервал перечитывания шаблонов.	`30s`
`DADATA_DEFAULT_TOKEN`	Токен DaData.	пусто
`DADATA_BASE_URL`	Адрес API DaData.	`https://suggestions.dadata.ru`
`DADATA_TIMEOUT`	Тайм-аут обращения к DaData.	`5s`
`DADATA_CONFIG_PATH`	Конфигурация DaData.	`config/dadata.json`
`DADATA_CONFIG_RELOAD_INTERVAL`	Интервал перечитывания конфигурации.	`30s`
`EMAIL_WORKERS_ENABLED`	Обработка электронной почты.	`true`
`EMAIL_IMAP_POLL_INTERVAL`	Интервал проверки входящей почты.	`30s`
`EMAIL_OUTBOUND_POLL_INTERVAL`	Интервал отправки исходящей почты.	`30s`

Переменная	Назначение	Пример
`EMAIL_WORKER_BATCH_SIZE`	Размер пачки почтовых сообщений.	`100`
`EMAIL_LOCK_TIMEOUT`	Время блокировки почтовой задачи.	`15m`
`SLA_WORKER_ENABLED`	Обработка SLA.	`true`
`SLA_POLL_INTERVAL`	Интервал обработки SLA.	`30s`
`SLA_WORKER_BATCH_SIZE`	Размер пачки SLA-записей.	`100`

Параметры конкретных LDAP, IMAP и SMTP-подключений задаются в интерфейсе системы. До их включения обеспечьте сетевые доступы, перечисленные в системных требованиях.

Эксплуатация

Все команды выполняются из каталога установки. В текущем описании указан `/opt/l1-support`.

Проверка состояния

```
cd /opt/l1-support
docker compose ps -a
docker compose logs --tail=200
```

Постоянные сервисы `postgres`, `redis`, `minio`, `backend` и `frontend` должны быть `Up` и `healthy`. Состояние `Exited (0)` для одноразовых сервисов `migrate` и `minio-mc` является штатным.

Проверка локальных endpoint:

```
curl -f http://127.0.0.1:9100/api/v1/system/bootstrap/status
curl -fI http://127.0.0.1:9090
curl -f http://127.0.0.1:9104/minio/health/live
```

Журналы

```
docker compose logs --tail=200
docker compose logs -f backend
docker compose logs -f postgres redis minio
docker compose logs --since=30m backend
```

Не передавайте `.env`, токены, cookies, пароли и полные диагностические архивы третьим лицам без проверки содержимого.

Остановка и запуск

Остановить и запустить существующие контейнеры:

```
docker compose stop
docker compose start
```

Применить изменённую конфигурацию или пересоздать необходимые контейнеры:

```
ENVFILE=.env docker compose --env-file .env config --quiet
docker compose up -d
```

Не используйте `docker compose down -v`: параметр `-v` удаляет volumes PostgreSQL и MinIO вместе с данными.

Резервное копирование

В резервную копию должны входить:

- дамп PostgreSQL;
- содержимое бакета MinIO;
- `.env` и неизменяемый APP_SECRETS_KEY;
- каталог `config`;
- номера версий backend и frontend.

Redis содержит временные данные и отдельно не резервируется.

PostgreSQL

```
docker compose exec -T postgres sh -c \
  'pg_dump -U "$POSTGRES_USER" -d "$POSTGRES_DB" --format=custom' \
  > l1-support-postgres-$(date +%F-%H%M%S).dump
```

Команда должна завершиться с кодом `0`, а полученный файл — иметь ненулевой размер.

MinIO

Данные находятся в volume `minio_data`. Настройте регулярное копирование бакета во внешнее S3-совместимое или файловое хранилище, например средствами `mc mirror` или корпоративной системы резервного копирования.

Не ограничивайтесь копированием файлов активного Docker volume без остановки MinIO: такая копия может быть несогласованной.

Требования к резервным копиям

- PostgreSQL и MinIO должны относиться к одному согласованному периоду.
- Копии должны храниться за пределами Docker-сервера.
- Доступ к `.env` и APP_SECRETS_KEY должен быть ограничен.
- Срок хранения и допустимая потеря данных должны соответствовать принятым RPO/RTO.
- Восстановление необходимо регулярно проверять на отдельном стенде.

Для строгой согласованности остановите запись данных на время создания обеих копий:

```
docker compose stop backend frontend
# Создайте дамп PostgreSQL и копию бакета MinIO.
docker compose start backend frontend
```

Восстановление

Перед восстановлением сохраните текущее состояние и убедитесь, что выбраны согласованные копии PostgreSQL и MinIO.

Остановите приложение:

```
docker compose stop backend frontend
```

Восстановите PostgreSQL:

```
docker compose exec -T postgres sh -c \
  'pg_restore -U "$POSTGRES_USER" -d "$POSTGRES_DB" --clean --if-exists' \
  < l1-support-postgres-YYYY-MM-DD-HHMMSS.dump
```

Восстановите бакет MinIO из соответствующей копии принятой в организации процедурой, затем запустите и проверьте систему:

```
docker compose start backend frontend
docker compose ps -a
```

Проверьте API, веб-интерфейс, открытие существующего вложения и загрузку нового файла.

Обновление

Перед обновлением:

1. Зафиксируйте текущие `BACKEND_VERSION` и `FRONTEND_VERSION`.
2. Создайте согласованные резервные копии PostgreSQL и MinIO.
3. Сравните новый `.env.example` с действующим `.env`.
4. Проверьте примечания к релизу и совместимость отката схемы БД.

После переноса новых файлов поставки:

```
cd /opt/l1-support
sudo chown -R 10001:10001 config logs
ENVFILE=.env docker compose --env-file .env config --quiet
```

В контуре с registry загрузите новые образы:

```
docker compose pull backend frontend migrate
```

В закрытом контуре вместо `pull` загрузите переданные архивы через `docker load` и проверьте теги командой `docker compose config --images`.

Примените обновление:

```
docker compose up -d
docker compose ps -a
```

Compose применит миграции до запуска новой версии backend. После обновления проверьте API, веб-интерфейс, загрузку и скачивание файлов, почтовые обработчики и используемые провайдеры аутентификации.

Откат

Откат только образов допустим, если предыдущая версия приложения совместима с уже применённой схемой базы данных.

Если схема совместима, верните предыдущие версии в `.env`, обеспечьте наличие образов на сервере и выполните:

```
docker compose up -d --no-deps backend frontend
docker compose ps -a
```

В контуре с registry отсутствующие предыдущие образы можно предварительно получить командой `docker compose pull backend frontend`.

Если предыдущая версия несовместима с новой схемой, остановите приложение и восстановите согласованные резервные копии PostgreSQL и MinIO. Не запускайте down-миграции без явно предусмотренной для релиза процедуры.

Диагностика

Проблема	Что проверить	Команда или действие
`pull access denied` или `manifest unknown`	Путь к образу, точную версию и авторизацию в registry.	`docker compose config --images`, `docker login <registry>`.
В закрытом контуре выполняется обращение к registry	На сервер загружены не все образы или не совпадают теги.	`docker image ls`, `docker compose config --images`.
`migrate` завершился с `Exited (1)`	Параметры PostgreSQL, готовность БД и журнал миграций.	`docker compose logs --tail=200 migrate postgres`.
Backend имеет состояние `unhealthy`	PostgreSQL, Redis, MinIO, завершение миграций и значения `.env`.	`docker compose ps -a`, `docker compose logs --tail=200 backend`.
Интерфейс работает, файлы не загружаются	`S3_PUBLIC_ENDPOINT`, DNS, TLS, лимит reverse proxy и доступ рабочих мест к S3.	Открыть `${S3_PUBLIC_ENDPOINT}/minio/health/live` с рабочего места.
	DNS, firewall, reverse proxy и совпадение публичных URL.	Сравнить `.env` с конфигурацией reverse proxy.

Проблема	Что проверить	Команда или действие
Система работает локально, но недоступна пользователям		
LDAP не подключается	Корпоративный DNS, маршрутизацию, сертификат и порты 389/636.	Проверить разрешение имени и TCP-доступ из сети Docker-сервера.
Письма не приходят или не отправляются	IMAP/SMTP, DNS, TLS и журналы почтовых обработчиков.	`docker compose logs --since=30m backend`.
После перезагрузки сервисы не запустились	Автозапуск Docker и состояние контейнеров.	`systemctl status docker`, `docker compose ps -a`.
На диске заканчивается место	Docker Root Dir, volumes, журналы и незавершённые загрузки.	`df -h`, `docker system df`, `docker volume inspect ...`.

Если проблема сохраняется, соберите версии образов, вывод `docker compose ps -a`, относящийся к ошибке фрагмент журналов и время возникновения. Перед передачей материалов удалите секреты и персональные данные.