

Инструкция по установке и обновлению ПО

ПО: Платформа «BSTech ZeroCode Platform», версия 1.54.0

Правообладатель: ООО «БС Технологии»

1. Назначение документа

Настоящая инструкция описывает порядок получения дистрибутива, установки, первоначальной настройки, запуска и остановки платформы «BSTech ZeroCode Platform» версии 1.54.0.

2. Требования к серверной инфраструктуре

Аппаратные и программные требования к серверу приведены в документе «Описание требований к инфраструктуре».

Перед установкой убедитесь, что на сервере установлены Docker Engine 24+ и Docker Compose v2:

```
docker --version      # Docker Engine 24+  
docker compose version # Docker Compose v2.x
```

Если Docker Engine ещё не установлен, выполните установку в соответствии с документацией для вашего дистрибутива ОС. Добавьте текущего пользователя в группу docker, чтобы работать без sudo:

```
sudo usermod -aG docker $USER
```

После добавления в группу выполните повторный вход в систему.

3. Установка и обновление в режиме онлайн

Платформа поставляется в двух вариантах установки: онлайн и оффлайн.

В варианте онлайн Заказчику предоставляются инструкция по установке (настоящий документ), и для выполнения онлайн-установки достаточно выполнение действий, описанных ниже в этом разделе.

3.1. Онлайн-установка на одном сервере

Выполните команды ниже, предварительно подставив:

- вместо USERLOGIN и PASSWORD — предоставленные Правообладателем логин и пароль к реестру registry.bstechnology.ru для используемой лицензии на Платформу;
- вместо zerocode.example.ru — домен, на котором должен открываться веб-интерфейс Платформы.

```
echo 'PASSWORD' | docker login registry.bstechnology.ru -u USERLOGIN --password-stdin
docker run --rm --pull always -v "$PWD":/out registry.bstechnology.ru/zerocode/vela-install
cd vela && ./install.sh zerocode.example.ru
```

Если требуется установить конкретную версию, вместо vela-install укажите vela-install:<версия>.

3.2. Онлайн-установка на двух серверах — платформа и БД отдельно

При раздельной установке компоненты распределяются между двумя серверами:

- сервер данных: PostgreSQL, MinIO, Operaton, ChromaDB;
- сервер приложения: vela-core, gost-signer.

На сервере данных выполните:

```
echo 'PASSWORD' | docker login registry.bstechnology.ru -u USERLOGIN --password-stdin
docker run --rm --pull always -v "$PWD":/out registry.bstechnology.ru/zerocode/vela-install
cd vela && ./install.sh data
```

На сервере приложения выполните те же две первые команды для загрузки скриптов, затем:

1. Скопируйте файл vela/.env с сервера данных на сервер приложения.
2. Отредактируйте .env на сервере приложения: раскомментируйте и укажите IP-адрес сервера данных в параметрах POSTGRES_HOST, MINIO_ENDPOINT, OPERATON_BASE_URL=http://<IP>:8080, VECTOR_DB_URL=http://<IP>:8000.
3. Выполните установку (предварительно заменив zerocode.example.ru на домен, на котором должен открываться веб-интерфейс Платформы):

```
cd vela && ./install.sh app zerocode.example.ru
```

3.3. Онлайн-обновление

Обновление выполняется из той же папки, где была выполнена установка.

Предварительно подставьте вместо USERLOGIN и PASSWORD предоставленные Правообладателем логин и пароль к реестру.

Если установка была на одном сервере:

```
echo 'PASSWORD' | docker login registry.bstechnology.ru -u USERLOGIN --password-stdin
docker run --rm --pull always -v "$PWD":/out registry.bstechnology.ru/zerocode/vela-install
cd vela && ./upgrade.sh
```

Скрипт upgrade.sh автоматически создаёт резервную копию базы данных перед обновлением. Настройки в файле .env сохраняются.

Если установка была на двух серверах (раздельно):

На сервере данных:

```
echo 'PASSWORD' | docker login registry.bstechnology.ru -u USERLOGIN --password-stdin
docker run --rm --pull always -v "$PWD":/out registry.bstechnology.ru/zerocode/vela-install
cd vela && ./upgrade.sh data
```

На сервере приложения:

```
echo 'PASSWORD' | docker login registry.bstechnology.ru -u USERLOGIN --password-stdin
docker run --rm --pull always -v "$PWD":/out registry.bstechnology.ru/zerocode/vela-install
cd vela && ./upgrade.sh app
```

При раздельном обновлении сначала обновляется сервер данных, затем сервер приложения. Резервное копирование базы данных выполняется автоматически при обновлении сервера данных.

4. Установка и обновление в режиме оффлайн

4.1. Получение и распаковка дистрибутива

Для установки в изолированных сетях без доступа к интернету заказчику предоставляется автономный дистрибутив vela`-<версия`>.tar.gz, содержащий все скрипты, файлы конфигурации и контейнерные образы.

Если офлайн-дистрибутив поставлен в виде частей, перед распаковкой соберите его:

```
cat vela-<версия>.tar.gz.part* > vela-<версия>.tar.gz
```

Проверьте контрольную сумму:

```
sha256sum -c vela-<версия>.tar.gz.sha256
```

Ожидаемый результат: ОК.

Распакуйте дистрибутив и перейдите в каталог:

```
mkdir -p /opt/vela
tar -xzf vela-<версия>.tar.gz -C /opt/vela
cd /opt/vela
```

Каталог содержит:

Файл / каталог	Назначение
install.sh	Скрипт установки
upgrade.sh	Скрипт обновления
backup.sh	Скрипт резервного копирования
docker-compose.yml	Базовая конфигурация контейнеров
docker-compose.app.yml	Конфигурация тира приложения (для раздельной установки)
docker-compose.data.yml	Конфигурация тира данных (для раздельной установки)
``env``.example	Шаблон параметров установки
VERSION	Версия дистрибутива

Файл / каталог	Назначение
SHA256SUMS	Контрольные суммы содержимого
registry.env	Учётные данные реестра образов (при онлайн-установке)
images``/	Контейнерные образы (только при офлайн-установке)

4.2. Оффлайн-установка на одном сервере

Для установки всех компонентов на одном сервере выполните:

```
cd /opt/vela
```

```
bash install.sh all
```

Параметр all является значением по умолчанию, поэтому допустимо также:

```
bash install.sh
```

Скрипт выполнит следующие действия:

1. Создаст файл .env из шаблона .env.example.
2. Сгенерирует секреты (пароль PostgreSQL, ключи MinIO, JWT-секрет, ключ шифрования хранилища секретов) и подставит их в .env.
3. Определит способ получения образов:
 - если обнаружен каталог images/ — загрузит образы из локальных файлов (docker load);
 - если обнаружен файл registry.env — выполнит авторизацию в реестре и загрузит образы (docker compose pull);
 - в остальных случаях — загрузит образы из публичного реестра.
4. Запустит контейнеры (docker compose up -d).
5. Применит миграции базы данных автоматически при первом запуске контейнера vela-core.

4.3. Оффлайн-установка на двух серверах — платформа и БД отдельно

При раздельной установке компоненты распределяются между двумя серверами:

- сервер данных: PostgreSQL, MinIO, Operaton, ChromaDB;
- сервер приложения: vela-core, gost-signer.

Порядок установки:

1. На сервере данных распакуйте дистрибутив и выполните:

```
bash install.sh data
```

2. Скопируйте файл .env с сервера данных на сервер приложения.

Секреты в обоих файлах должны совпадать.

3. На сервере приложения отредактируйте .env, указав адреса компонентов на сервере данных:

POSTGRES_HOST=<IP-адрес сервера данных>

MINIO_ENDPOINT=<IP-адрес сервера данных>:9010

OPERATON_BASE_URL=http://<IP-адрес сервера данных>:8083/engine-rest

VECTOR_DB_URL=http://<IP-адрес сервера данных>:8000

4. На сервере приложения выполните:

```
bash install.sh app
```

4.2. Оффлайн-обновление

Для обновления платформы запустите скрипт upgrade.sh в папке, где установлена платформа.

5. Настройка обратного прокси-сервера

Для обеспечения защищённого доступа по HTTPS рекомендуется настроить обратный прокси-сервер (nginx, Traefik или аналогичный) с TLS-терминацией, проксирующий запросы на порт платформы (VELA_PORT).

Пример минимальной конфигурации nginx:

```
server {  
    listen 443 ssl;  
    server_name platform.company.ru;  
  
    ssl_certificate /path/to/cert.pem;  
    ssl_certificate_key /path/to/key.pem;  
  
    location / {  
        proxy_pass http://127.0.0.1:8080;  
        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;  
    }  
}
```

6. Настройка параметров и пользователей после установки

6.1. Настройка параметров в файле .env

После выполнения установки файл .env содержит параметры по умолчанию. Перед первым запуском (или сразу после установки) рекомендуется проверить и при необходимости изменить следующие параметры:

Параметр	Описание	Значение по умолчанию
APP_BASE_URL	Внешний URL платформы (используется в ссылках уведомлений, привязке WebAuthn)	http://localhost:8080
VELA_PORT	Порт, на котором платформа доступна на хосте	8080

В параметре APP_BASE_URL необходимо установить реальный адрес, по которому платформа будет доступна пользователям (например, <https://platform.company.ru>).

После изменения .env перезапустите платформу (см. раздел 6).

Рекомендуется ограничить доступ к файлу .env, так как он содержит секреты:

```
chmod 600 .env
```

6.2. Первый вход и создание учетной записи администратора

После установки и запуска откройте платформу в браузере по адресу, указанному в APP_BASE_URL (или http://<IP-сервера>:<VELA_PORT>).

При первой попытке входа Платформа предлагает зарегистрировать учётную запись первого пользователя, который автоматически получает роль администратора платформы с полным доступом ко всем функциям.

6.3. Создание дополнительных пользователей

Для создания дополнительных пользователей:

1. Войдите в платформу под учётной записью администратора.
2. Перейдите в раздел «Конфигурация» → «Пользователи».
3. Нажмите «Создать пользователя».
4. Укажите данные пользователя и выберите тип:
 - «Человек» — для интерактивной работы;
 - «Система» — для интеграций через API.
5. Назначьте пользователю роль:
 - «Admin» — полный доступ ко всем функциям платформы;
 - «Дизайнер» — доступ к проектированию в рамках назначенных проектов;
 - «Guest» — минимальный доступ.

7. Запуск и остановка платформы

7.1. Запуск

Для запуска платформы выполните из каталога установки:

```
cd /opt/vela
```

```
docker compose up -d
```

Дождитесь, пока все контейнеры перейдут в состояние healthy/running.

7.2. Проверка состояния

Убедитесь, что все контейнеры работают:

```
docker compose ps
```

Все сервисы должны находиться в состоянии running или healthy.

Проверьте доступность REST API:

```
curl -fsS http://localhost:${VELA_PORT:-8080}/api/health
```

Ожидаемый результат: HTTP 200.

Откройте платформу в браузере по адресу `http://<IP-сервера>:<VELA_PORT>` — должна отобразиться страница входа.

7.3. Остановка

Для остановки платформы без удаления данных:

```
cd /opt/vela
```

```
docker compose down
```

Данные сохраняются в постоянных томах Docker и будут доступны при следующем запуске.

7.4. Перезапуск

Для перезапуска платформы:

```
cd /opt/vela
```

```
docker compose down && docker compose up -d
```

7.5. Просмотр журналов

Для просмотра журналов всех контейнеров:

```
docker compose logs -f
```

Для просмотра журнала конкретного контейнера:

```
docker compose logs -f vela-core
```

8. Резервное копирование

Для создания резервной копии данных платформы:

```
cd /opt/vela
```

```
bash backup.sh
```

Скрипт создаст архив, содержащий резервную копию базы данных PostgreSQL и файл `.env` с параметрами установки.

Рекомендуется выполнять резервное копирование регулярно, а также перед каждым обновлением платформы. При обновлении скриптом `upgrade.sh` резервное копирование выполняется автоматически.

9. Удаление

Например, если была неудачная установка, то перед новой установкой лучше сделать удаление.

В папке установки платформы, например `vela`:

Что нужно	Команда
Убрать платформу, данные сохранить (база, файлы, пароли)	<code>./uninstall.sh</code> — вернуть: <code>./install.sh</code> в той же папке
Удалить всё : данные, ключи ГОСТ, образы	<code>./uninstall.sh --purge</code> — спросит подтверждение словом УДАЛИТЬ

Перед полным удалением делается бэкап базы в эту же папку (`backup-before-uninstall-....sql.gz`); без бэкапа — `--no-backup`, без вопроса (для скриптов) — `--yes`.
Файлы `.env` и бэкапы остаются — папку удалите сами, когда они станут не нужны.