Руководство администратора

Развёртывание ПО

Для развёртывания ПО потребуется сервер со следующими характеристиками:
1. Процессор x86_64, 3+ ГГц, 8 ядер
2. 32 ГБ ОЗУ
3. 500 ГБ SATA или NVMe SSD
4. Сетевой интерфейс 100 Мбит
5. Операционная система — на базе Linux, Debian-based дистрибутив (Debian, Ubuntu, Mint)

Установка дополнительного ПО

После установки базового образа ОС необходимо выполнить установку дополнительного ПО:
1. СУБД MySQL (community edition)
2. Интерпретатор PHP с Fast/CGI и библиотеками расширений
3. Веб-сервер Apache2
4. Redis
5. Node.js
6. Composer
7. Дополнительные пакеты

Перед началом установки дополнительного ПО в Debian-based дистрибутиве необходимо обновить список пакетов, доступных пакетному менеджеру:

```
sudo apt update
```

Для установки нужного пакета в Debian-based дистрибутиве Linux выполните следующую команду:

```
sudo apt install имя_пакета
```

1 Установка MySQL

Установите MySQL:

```
sudo apt install mysql-server mysql-client
```

2 Установка PHP

Установите PHP с CGI и библиотеками расширений:

```
sudo apt install php php-fpm php-gd php-xml php-zip php-intl php-mysql php-
curl
```

3 Установка Apache2

Установите веб-сервер Apache2:

```
sudo apt install apache2 libapache2-mod-php
```

4 Установка Redis

Установите сервер Redis:

```
sudo apt install redis-server
```

5 Установка Node.js

5.1 Установка дополнительных пакетов

Установите утилиту командной строки `curl` для скачивания файлов:

```
sudo apt install curl
```

5.2 Скачивание и запуск установочного скрипта `NodeSource` для `Node.js`

Скачайте установочный скрипт `NodeSource` для `Node.js` (в данном случае скачивается скрипт для установки Node.js версии 22):

```
curl -fsSL https://deb.nodesource.com/setup_22.x -o nodesource_setup.sh
```

Пожалуйста, проверьте содержимое установочного скрипта любым удобным вам способом, например:

```
cat nodesource_setup.sh
nano nodesource_setup.sh
vi nodesource_setup.sh
# и т.п.
```

И в случае, если вас устраивают изменения, которые скрипт вносит в систему, запустите его с правами root:

```
sudo -E bash nodesource_setup.sh
```

При успешном выполнении скрипт установит необходимые вспомогательные пакеты и добавит в систему репозиторий NodeSource, из которого можно установить Node.js.

5.3 Установка пакета `Node.js`

Для установки `Node.js` (вместе с `npm`) выполните команду:

```
sudo apt install nodejs
```

Проверьте версию Node.js. При успешной установке команда должна выдать запрошенную версию пакета. Например, в данном случае (установка `Node.js` версии 22) команда:

```
node -v
```

должна выдать версию 22 (минорная версия и номер патча могут отличаться):

```
v22.14.0
```

При возникновении проблем, пожалуйста, обратитесь к инструкции, доступной по адресу:
🔗 NodeSource GitHub

6 Установка Composer

6.1 Установка дополнительных пакетов

Установите программу `unzip` для распаковки zip-архивов и версию интерпретатора PHP для командной строки:

```
sudo apt install php-cli unzip
```

6.2 Запустите следующий скрипт для скачивания и установки Composer

```
#!/bin/sh
```

```
EXPECTED_CHECKSUM="$(php -r
'copy("https://composer.github.io/installer.sig", "php://stdout");')"
php -r "copy('https://getcomposer.org/installer', 'composer-setup.php');"
ACTUAL_CHECKSUM="$(php -r "echo hash_file('sha384', 'composer-
setup.php');")"

if [ "$EXPECTED_CHECKSUM" != "$ACTUAL_CHECKSUM" ]
then
    >&2 echo 'ERROR: Invalid installer checksum'
    rm composer-setup.php
    exit 1
fi

php composer-setup.php
RESULT=$?
rm composer-setup.php
exit $RESULT
```

Для этого сохраните скрипт на сервер и задайте для него права на запуск, например:

```
scp -p install-composer.sh вашсервер:
ssh вашсервер
# далее в шелле сервера:
chmod 755 install-composer.sh
./install-composer.sh
```

Примечание: если скрипт при запуске выдаёт ошибки, например, "`Error in argument 1, char 2: no argument for option r`", пожалуйста, убедитесь, что после копирования скрипта из PDF-документа и вставки в файле отсутствуют лишние переносы строк скрипта.

При успешном выполнении скрипт выведет сообщение вида:

```
Composer (version X.Y.Z) successfully installed to:
/home/username/composer.phar
Use it: php composer.phar
```

где X.Y.Z — версия установленного composer, а username — имя вашего пользователя.
Для удобства рекомендуется скопировать исполняемый файл composer в привычное расположение под именем без расширения:

```
sudo cp /home/username/composer.phar /usr/local/bin/composer
```

Тогда composer можно будет запускать из любого места без указания пути к исполняемому файлу, используя имя команды `composer`.

7 Установка дополнительных пакетов

Для поддержки генерации QR на данном этапе требуется установить пакет `libgl1`:

```
sudo apt install libgl1
```

Настройка дополнительного ПО

1 Настройка MySQL

Для настройки MySQL с нуля потребуется сделать следующее:
1. Создать базу данных.
2. Создать пользователя-администратора базы данных.
3. Выдать пользователю права на доступ к базе данных.

Запустите клиент MySQL от имени пользователя `root`:

```
sudo mysql
```

Создание базы данных

Внутри оболочки клиента MySQL выполните следующий оператор SQL, чтобы создать базу данных:

```
CREATE DATABASE ostrich;
```

Создание пользователя в MySQL

Внутри оболочки клиента MySQL выполните следующий оператор SQL, указав желаемый пароль пользователя:

```
CREATE USER 'ostrich_admin'@'localhost' IDENTIFIED BY 'пароль_админа_БД';
```

Примечание: в некоторых дистрибутивах Linux в СУБД MySQL, поставляемой в пакетном репозитории, включёна строгая политика паролей, поэтому пароли должны иметь длину не менее 8 символов и содержать строчные и прописные буквы, цифры, а также специальные символы. Рекомендуется генерировать пароль с помощью утилиты `pwgen`, в которой можно задать требования к паролю. Например, следующий вызов команды сгенерирует несколько 12-символьных паролей, удовлетворяющих перечисленным условиям:

```
pwgen -s -y 12
```

Выдача пользователю прав на базу данных

Внутри оболочки клиента MySQL выполните следующие операторы SQL для выдачи прав на базу данных новому пользователю:

```
GRANT ALL PRIVILEGES ON ostrich.* TO 'ostrich_admin'@'localhost' WITH GRANT
OPTION;
GRANT CREATE USER, CREATE ROLE, RELOAD ON *.* to
'ostrich_admin'@'localhost';
GRANT ROLE_ADMIN ON *.* TO 'ostrich_admin'@'localhost';
GRANT CREATE ON *.* TO 'ostrich_admin'@'localhost';
-- выдайте следующие права, чтобы работала функция SQL CONVERT_TZ():
GRANT SELECT ON mysql.time_zone_name TO 'ostrich_admin'@'localhost';
GRANT SELECT ON mysql.time_zone TO 'ostrich_admin'@'localhost';
GRANT SELECT ON mysql.time_zone_transition TO 'ostrich_admin'@'localhost';
GRANT SELECT ON mysql.time_zone_transition_type TO
'ostrich_admin'@'localhost';
GRANT SELECT ON mysql.time_zone_leap_second TO 'ostrich_admin'@'localhost';
```

На всякий случай можно перезагрузить привилегии (необязательно, т.к. делается автоматически):

```
FLUSH PRIVILEGES;
```

Примечание 1: для большей безопасности можно создать дополнительного пользователя БД с ограниченным набором прав на доступ к БД. Это можно сделать следующим образом:

```
-- создать нового пользователя базы данных:
CREATE USER 'ostrich_user'@'localhost' IDENTIFIED BY 'пароль_пользователя';
-- разрешить SELECT на всех таблицах базы:
GRANT SELECT ON ostrich.* TO 'ostrich_user'@'localhost';
-- по аналогии выдать INSERT, UPDATE и DELETE на нужные таблицы
```

Созданного таким образом ограниченного пользователя можно затем использовать в ПО Ostrich для доступа к БД вместо `ostrich_admin`. Однако следует помнить, что для корректной работы всех функций Ostrich системе требуется доступ на чтение (`SELECT`) и на запись (`INSERT`, `UPDATE` и `DELETE`) для всех таблиц базы данных Ostrich. В отдельных случаях (миграция данных и импорт логов регистраторов) требуются также права `CREATE`, `ALTER` и `DROP`.
Во избежание случайных сбоев рекомендуется всё же использовать полноправный аккаунт.

Примечание 2: опция `WITH GRANT OPTION`, а также права `CREATE USER, CREATE ROLE, RELOAD, ROLE_ADMIN` — требуются для корректной работы скрипта начального развёртывания (см. раздел "Запуск скрипта начального развёртывания" ниже).

2 Настройка Apache2

Для поддержки работы PHP-FPM требуется разрешить модуль FCGI-прокси:

```
sudo a2enmod proxy_fcgi
```

Для ускорения работы, возможно, потребуется разрешить модуль `socache_shmcb`:

```
sudo a2enmod socache_shmcb
```

Для работы ПО необходимо создать каталог, в котором будут размещаться исходные тексты и который будет обрабатываться веб-сервером. Рекомендуется создавать такой каталог в каталоге `/var/www`. Например: `/var/www/ostrich`

```
sudo mkdir -p /var/www/ostrich
```

Смените владельца на своего пользователя:

```
sudo chown $USER:$USER /var/www/ostrich
```

Далее необходимо создать конфигурацию веб-сервера. Создайте новый файл в каталоге Apache2 `/etc/apache2/sites-available` с именем `ostrich.conf`. Вставьте в него следующий текст, заменив имя домена `example.com` на имя вашего домена:

```
<VirtualHost *:80>
    ServerName example.com
    ServerAlias www.example.com

    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/ostrich/public
    DirectoryIndex /index.php

    <FilesMatch \.php$>
        SetHandler proxy:unix:/var/run/php/php-fpm.sock|fcgi://dummy
    </FilesMatch>

    <Directory /var/www/ostrich/public>
        AllowOverride None
        Require all granted

        FallbackResource /index.php
    </Directory>

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
```

```
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/ostrich_error.log
    CustomLog ${APACHE_LOG_DIR}/ostrich_access.log combined
</VirtualHost>
```

Сохраните файл и выполните команды для отключения конфигурации по умолчанию и включения новой конфигурации:

```
sudo a2dissite 000-default
sudo a2ensite ostrich
```

Прежде чем перезапустить Apache, пожалуйста, разверните основное ПО.

Развёртывание основного ПО

Распаковка исходных текстов

Скопируйте архив с основным ПО в ранее созданный каталог `/var/www/ostrich` и распакуйте его:

```
tar xf ostrich-X.Y.Z.tar.gz
```

В каталоге `/var/www/ostrich` создайте файл `.env.local` следующего вида:

```
APP_ENV=prod
APP_DEBUG=0
SYMFONY_ENV=prod

DATABASE_NAME="ostrich"
DATABASE_USER="ostrich_admin"
DATABASE_PASS="пароль_админа_БД"

FAILURE_CHECK_USER=failure_check_user
FAILURE_CHECK_PASS=

MAILER_DSN="smtp://smtp_user_name:smtp_password@smtp.example.com:25"

JWT_PASSPHRASE=
```

Вместо `пароль_админа_БД` подставьте пароль, сгенерированный при создании пользователя БД (см. раздел "Создание пользователя в MySQL" выше).
В переменную `MAILER_DSN` подставьте имя и пароль пользователя SMTP и адрес вашего SMTP-сервера.

Запустите composer для установки всех необходимых зависимостей:

```
composer install
```

Запуск скрипта начального развёртывания

После этого можно запускать скрипт начального развёртывания. Перейдите в каталог, где размещено ПО Ostrich (`/var/www/ostrich`) и запустите команду:

```
php bin/console app:initial:deploy
```

Примечание: по умолчанию интерфейс скрипта имеет русский язык, но если заголовки и сообщения скрипта на экране отображаются не на русском, пожалуйста, укажите язык явно при запуске:

```
php bin/console app:initial:deploy --locale=ru_RU
```

На первом этапе скрипт развёртывания запросит подтверждение пользователя о том, что скрипт запущен на правильном хосте. Если пользователь не подтвердит этого, выполнение скрипта будет прервано.

```
Скрипт начального развёртывания Ostrich
=======================================

Основные
--------

Текущая среда: dev
Ожидаемое имя хоста: <не указано>
Имя данного хоста: home


 [WARNING] Пожалуйста, проверьте дважды, что вы делаете и на каком хосте
запускаете этот скрипт!



 Этот инструмент предназначен для запуска на продакшен-сервере и может
сломать что-то работающее, поэтому — убедительно
 вас просим — проверьте имя хоста. Спасибо!

Резюме
------

 ------------------ --------------
  Переменная         Значение
```

```
 ------------------ --------------
  env                dev
  expectedHostname   <не указано>
  hostname           home
 ------------------ --------------

 Это верные сведения? (yes/no) [no]:
 > _
```

Далее скрипт последовательно предложит задать важнейшие настройки и выполнить инициализирующие действия:
1. Задать параметры доступа к БД
2. Выполнить миграцию базы
3. Задать парольную фразу и ключи для генерации JWT
4. Создать первого пользователя (администратора) в разворачиваемой системе Ostrich
5. Создать некоторые другие конфигурационные файлы

1 Задание параметров доступа к БД

Скрипт проверит наличие доступа к БД и предложит задать параметры, если не обнаружит такового. Пользователь должен будет ввести адрес (или имя) хоста с БД, номер порта, имя пользователя и пароль, а также имя БД.

```
Доступ к БД
-----------


 ! [NOTE] Кажется, доступ к БД не настроен

Имя БД:
Имя пользователя:
Пароль: <вырезано>
Хост БД: localhost
Порт хоста: 3306

 Вы хотите задать параметры доступа к БД? (yes/no) [yes]:
 > _
```

Ответьте `yes` (или `y`) и задайте параметры доступа в диалоге.

```
Имя БД []:
 > ostrich

 Имя пользователя []:
 > dbadmin

 Пароль:
 > ************
```

```
 Пароль ещё раз:
 > ************



 [INFO] Пароли совпадают.


 Хост БД [localhost]:
 >

 Порт хоста [3306]:
 >

Резюме
------

 --------------- ------------
  Переменная      Значение
 --------------- ------------
  DATABASE_NAME   ostrich
  DATABASE_USER   dbadmin
  DATABASE_PASS   <вырезано>
  DATABASE_HOST   localhost
  DATABASE_PORT   3306
 --------------- ------------

 Это верные сведения? (yes/no) [no]:
 > y
```

После этого скрипт будет завершён с сообщением: "[WARNING] Скрипт необходимо перезапустить, чтобы изменения вступили в силу!".
Перезапустите скрипт, чтобы продолжить работу с ним.

Если же соединение с БД уже настроено и работает, скрипт определит это и выдаст соответствующее сообщение и уже заданные параметры доступа:

```
Доступ к БД
-----------


 ! [NOTE] Кажется, БД уже настроена и к ней есть доступ.

Имя БД: ostrich
Имя пользователя: dbadmin
Пароль: <вырезано>
Хост БД: localhost
Порт хоста: 3306
```

```
 Вы хотите задать параметры доступа к БД? (yes/no) [no]:
 > _
```

Ответьте `no` (или `n`) и скрипт перейдёт к следующему этапу.

2 Выполнение миграции базы

Скрипт запросит у пользователя подтверждение на запуск миграции базы данных. Если вы уже создали схему вручную или восстановили базу из резервной копии на этапе "Настройка MySQL", вы можете пропустить этот шаг. Также следует помнить, что миграцию при начальном развёртывании можно выполнить только один раз.

```
Миграция
--------

 Вы хотите выполнить миграцию? (yes/no) [no]:
 > _
```

3 Создание парольной фразу и ключей для генерации JWT

Скрипт предложит сгенерировать или задать вручную парольную фразу для работы с JWT, если она ещё не задана.

Внимание: если вы решили задать новую парольную фразу JWT, необходимо перезапустить скрипт и после перезапуска выбрать генерацию новой пары ключей JWT!

```
JWT
---

 Парольная фраза JWT не задана! Хотите сгенерировать новую? (yes/no) [yes]:
 > _
```

Ответьте `да` — скрипт сгенерирует новую парольную фразу, сохранит её в файле переменных окружения, очистит кеш завершит работу. Перезапустите скрипт.

Также скрипт предложит сгенерировать пару ключей для работы с JWT, если ключи не найдены в каталоге `config/jwt`.

```
 Пара ключей JWT не найдена! Хотите сгенерировать новую? (yes/no) [yes]:
 > _
```

Ответьте `да` — скрипт сгенерирует новую пару ключей и сохранит файлы в каталоге `config/jwt`.

Если же ключи существуют в каталоге `config/jwt` на момент запуска скрипта, то будет выведено предупреждение, что при генерации новой пары ключей старая будет утеряна. Если генерировать новую пару не требуется, откажитесь от этого, ответив "нет".

4 Создание первого пользователя (администратора) в разворачиваемой системе Ostrich

При наличии доступа к БД скрипт проверит, существует ли в системе пользователь с именем admin и предложит создать админа, если его не существует.

Примечание: Пользователю в любом случае будет дана возможность создать нового админа — и необязательно с именем admin.

```
Создание первого пользователя (admin)
-------------------------------------

 Пользователь {admin} не найден в БД! Хотите создать его? (yes/no) [yes]:
 > _
```

Ответьте "да" — скрипт предложит задать имя аккаунта нового пользователя, а также пароль и ФИО:

```
 Имя пользователя [admin]:
 > _
```

Примечание: на этапе задания имени аккаунта можно как согласиться с предложенным вариантом, так и ввести свой.

Для задания пароля существует два варианта: генерация и ввод вручную.
При выборе варианта "генерация" скрипт автоматически сгенерирует новый пароль и выведет его в консоль вместе с предупреждением о том, что пользователь должен запомнить пароль или скопировать его в буфер обмена. После нажатия клавиши Enter консоль будет очищена.

```
 Выберите, генерацию пароля или ввод вручную [генерация]:
  [0] генерация
  [1] ввод вручную
 > 0



 [INFO] Ваш новый пароль: 2bxXdw1sX4WGY49d



 !
 ! [CAUTION] Пожалуйста, сразу же запомните пароль или скопируйте его в
буфер обмена!
 !           Нажатие клавиши Enter очистит консоль!
```

```
 !

 Нажмите Enter для продолжения. Пароль будет удалён из консоли!:
 > Enter



 [INFO] Консоль была очищена
```

При выборе варианта "ввод вручную" скрипт запросит у пользователя пароль и его подтверждение. Введите свой пароль дважды. В случае если пароли не совпадают, скрипт потребует ввести их заново.

```
Пароль:
 > 2bxXdw1sX4WGY49d

 Пароль ещё раз:
 > ZuZY8cxbqY1VrzZA



 [ERROR] Пароли не совпадают!


 Пароль:
 >
```

Введите один и тот же пароль два раза, чтобы перейти к заданию других полей.

```
 Пароль:
 > 9SSNzMDy5YetBf42

 Пароль ещё раз:
 > 9SSNzMDy5YetBf42



 [INFO] Пароли совпадают.
```

```
Фамилия:
 > Пупкин

 Имя:
 > Василий
```

```
 [OK] Пользователь "admin" успешно создан.
```

Если же на момент запуска скрипта пользователь `admin` уже существует в БД системе, скрипт выведет сообщение об этом и предложит создать другого пользователя.

```
Создание первого пользователя (admin)
-------------------------------------

 Пользователь {admin} уже существует. Хотите добавить создать нового?
(yes/no) [no]:
 > _
```

5 Создание дополнительных конфигурационных файлов

Скрипт проверит наличие следующих дополнительных конфигурационных файлов:

```
config/Organization.json
config/UserQrCodeBurn.json
config/UserQrCodePrint.json
```

и предложит создать их, если они не существуют.

Для файла `config/Organization.json`, который содержит сведения об организации, возможно задание основных значений (название организации и т.п.) в диалоге прямо во время работы скрипта. Значения, введённые пользователем, попадут в этот файл и в дальнейшем будут использованы при генерации документов (например, в паспортах сборок).

```
Генерация конфигов Ostrich
--------------------------

 Файл "Organization.json" НЕ существует. Хотите создать его? (yes/no)
[yes]:
 > y

 Полное наименование организации []:
 > Общество с ограниченной ответственностью "Рога и копыта"

 Краткое наименование организации []:
 > ООО "Рога и копыта"

 Город []:
 > Бобруйск
```

```
 [OK] Файл Organization.json успешно сохранён


 Файл "UserQrCodeBurn.json" НЕ существует. Хотите создать его? (yes/no)
[yes]:
 > y



 [OK] Файл UserQrCodeBurn.json успешно сохранён


 Файл "UserQrCodePrint.json" НЕ существует. Хотите создать его? (yes/no)
[yes]:
 > y



 [OK] Файл UserQrCodePrint.json успешно сохранён
```

Если же скрипт обнаружит, что какой-либо из файлов уже существует, будет выдан соответствующий запрос, ответ по умолчанию на который — "нет":

```
Генерация конфигов Ostrich
--------------------------

 Файл "Organization.json" уже существует. Хотите его отредактировать?
(yes/no) [no]:
 >

 Файл "UserQrCodeBurn.json" уже существует. Хотите создать его заново?
ВНИМАНИЕ: содержимое существующего файла будет перезаписано значениями по
умолчанию! (yes/no) [no]:
 >

 Файл "UserQrCodePrint.json" уже существует. Хотите создать его заново?
ВНИМАНИЕ: содержимое существующего файла будет перезаписано значениями по
умолчанию! (yes/no) [no]:
 >
```

По окончании скрипт выводит сообщение:

```
 [OK] Команда успешно завершилась
```

Запуск скрипта развёртывания

После этого можно запустить скрипт собственно развёртывания ПО:

```
./deploy.sh
```

Скрипт скомпилирует используемые ресурсы, очистит кеш и выполнит некоторые другие необходимые действия.

Переназначение прав на каталоги

После того как отработает скрипт развёртывания, необходимо переназначить права доступа на следующие каталоги, так чтобы в них могли писать процессы Apache:

```
sudo chown -R www-data:www-data /var/www/ostrich/var
sudo chown -R www-data:www-data /var/www/ostrich/public
```

Перезапуск веб-сервера Apache

На данном этапе необходимо перезапустить Apache, если вы не сделали этого при установке и настройке веб-сервера или если его служба докладывает об ошибке:

```
sudo systemctl restart apache2
```

После этого ПО готово к использованию. Откройте страницу вашего веб-сервера в браузере и войдите в систему Ostrich под именем админа, созданного при развёртывании (или любого другого возможно существующего в БД Ostrich).