



# **ИНСТРУКЦИЯ ПО УСТАНОВКЕ И РАЗВЕРТЫВАНИЮ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ «ERP TERPO. ПРОИЗВОДСТВО», ПРЕДОСТАВЛЕННОГО ДЛЯ ПРОВЕДЕНИЯ ПРОВЕРКИ**

# Оглавление

|                                                                               |    |
|-------------------------------------------------------------------------------|----|
| Введение .....                                                                | 3  |
| 1. Технические требования .....                                               | 4  |
| 2. Описание .....                                                             | 5  |
| 2.1. Используемый стек технологий .....                                       | 5  |
| 2.2. Конфигурационные файлы .....                                             | 5  |
| 2.3. Адреса, порты и данные для авторизации использующиеся по умолчанию ..... | 9  |
| 3. Шаги по развертыванию системы .....                                        | 10 |
| 3.1. Перезапуск системы .....                                                 | 11 |
| 3.2. Штатный механизм создания резервных копий системы .....                  | 11 |
| Ссылки .....                                                                  | 14 |

# Введение

Данный документ содержит технические требования к серверу на котором будет развернута система, описание конфигурационных файлов и порядок действий для развертывания экземпляра системы на сервере.

# 1. Технические требования

Рекомендованные технические характеристики сервера:

- Канал связи от 10 Мбит/с и выше.
- Ubuntu 22.04 LTS и выше.
- 8 vCPU.
- 16 ГБ RAM.
- 120 ГБ NVMe SSD.

## 2. Описание

### 2.1. Используемый стек технологий

Используемый стек:

- Docker.
- Python.
- PostgreSQL.
- Nginx.

### 2.2. Конфигурационные файлы

**Compose.yml** — главный файл, в нем перечислены все сервисы, которые будет запускать docker. Остальные файлы — конфигурационные, они подключаются внутри compose.yml.

```
1  services:
2    web:
3      image: registry.nexus.terpo.ru/web/erp-infra/web/clientname:latest
4      pull_policy: always
5      depends_on:
6        - erp
7      ports:
8        - "80:80"
9        - "443:443"
10     volumes:
11       - ./nginx.conf.template:/etc/nginx/templates/default.conf.template
12     restart: always
13   erp:
14     image: registry.nexus.terpo.ru/web/erp-infra/erp/clientname:latest
15     pull_policy: always
16     env_file: "erp.env"
17     depends_on:
18       - db
19     ports:
20       - "127.0.0.1:8069:8069"
21       - "127.0.0.1:8071:8071"
22       - "127.0.0.1:8072:8072"
23     volumes:
24       - erp-web-data:/var/lib/erp
25       - ./erp.conf:/etc/erp/erp.conf
26     restart: always
27   erp_modules_updater:
28     image: registry.nexus.terpo.ru/web/erp-infra/erp/clientname:latest
29     pull_policy: always
30     env_file: "erp.env"
31     depends_on:
32       - db
33     volumes:
34       - erp-web-data:/var/lib/erp
35       - ./erp.conf:/etc/erp/erp.conf
36     command: --update=all --stop-after-init --database=dbname
37   db:
38     image: postgres:16.0-alpine
39     env_file: "erp.env"
40     volumes:
41       - erp-db-data:/var/lib/postgresql/data
42     restart: always
43 volumes:
44   erp-web-data:
45   erp-db-data:
```

- services — список контейнеров.
- image — имя образа, который будет использоваться для создания контейнера.
- pull\_policy — определяет то, как параметр будет получать последние версии образов.
- depends\_on — зависимости между контейнерами.

- ports — используется для сопоставления портов хоста с портами контейнера.
- volumes — определение томов, которые будут монтированы в контейнеры.
- restart — поведение контейнера при падении.
- env\_file — наименование файла окружения.
- command — определение команды, которая будет выполняться при запуск контейнера.

**Erp.conf** — параметры конфигурации сервера ERP-системы.

```

1 [options]
2 addons_path = /mnt/addons ; defines the addons path folders, which are added in priority order.
3 data_dir = /var/lib/erp ; specifies the path to the directory, containing the filestore for data linked to Erp.
4 admin_passwd = changeme ; Names the Master password, which can also be changed from the front end. The master password for Erp will be hashed and saved in the conf file when it is set or modified.
5 ; proxy mode = False ; When deploying our application behind a proxy, we can set this to True. True is the default setting
6 ; csv_internal_sep = , ; Specifies the separator to use in the CSV file.
7 ; db_maxconn = 64 ; specifies the maximum number of PostgreSQL physical connections.
8 ; db_name = False ; Names the database specifically. The default setting is False. If a database name is specified, only the Erp database selector and manager will display that database.
9 ; db_template = template1 ; Template1 is the default value.
10 ; dbfilter = * ; Makes use of REGEXP to filter the database.
11 ; debug_mode = False ; Activates Developer tools and technical menu
12 ; email_from = False ; Gives the SMTP email address that should be used to send emails.
13 ; limit_memory_hard = 2684354560 ; Defines the strict upper limit for virtual memory. Any worker that exceeds the limit will be terminated right away without waiting for the current request processing to finish.
14 ; limit_memory_soft = 2147483648 ; Specifies the most virtual RAM that a worker is permitted to use. Any worker who goes over this cap will be killed and recycled at the conclusion of the current request.
15 ; limit_request = 8192 ; Specifies how many requests a worker can handle before recycling and beginning over.
16 ; limit_time_cpu = 60 ; When the worker uses more than the allotted amount of CPU seconds to process the request, the worker will stop.
17 ; limit_time_real = 120 ; If a worker takes more than the allotted number of seconds to complete a request, the worker will be terminated.
18 list_db = False ; True/False. If False, hides the database list.
19 ; log_db = False ; True/False. If True, also write log to the 'jr_logging' table in the database.
20 ; log_handler = ['!INFO'] ; Chooses a value pair from the available "module:log level" pairings. The default value is '!INFO', which means all modules' default logging level is 'INFO'
21 ; log_level = info ; Any of the following values can be assigned: info, debug rpc, warn, test, critical, debug sql, error, debug, and debug rpc answer.
22 ; logfile = None ; Specifies the log filename. If not set, use stdout.
23 longpolling_port = False ; Specifies the TCP port that is utilized in multiprocessing or gevent mode for long-polling connections. The 8072 default value. In the default mode, it is not utilized (threaded mode).
24 gevent_port = 8072 ; Listen port for the gevent worker
25 max_cron_threads = 1 ; Maximum number of threads processing concurrently cron jobs (default 2).
26 ; osv_memory_age_limit = 1.0 ; Deprecated alias to the transient-age-limit option
27 ; osv_memory_count_limit = False ; Force a limit on the maximum number of records kept in the virtual osv_memory tables. The default is False, which means no count-based limit.
28 ; smtp_password = False ; specify the SMTP password for sending email
29 ; smtp_port = 25 ; specify the SMTP port
30 ; smtp_server = localhost ; specify the SMTP server for sending email
31 ; smtp_ssl = False ; if passed, SMTP connections will be encrypted with SSL (STARTTLS)
32 ; smtp_user = False ; specify the SMTP username for sending email
33 workers = 4 ; Specify the number of workers, 0 disable prefork mode.
34 ; xmlrpc = True ; It is used to turn on or off the XML-RPC protocol. Set to False to disable it.
35 ; xmlrpc_interface = ; Specifies the TCP IP address for the XML-RPC protocol. We can use the empty string, which binds to all interfaces.
36 ; xmlrpc_port = 8069 ; Specifies the TCP port for the XML-RPC protocol.
37 ; xmlrpcs = True ; It manages the on/off state of the XML-RPC security protocol. To turn it off, set it to False.
38 ; xmlrpcs_interface = ; XML-RPC protocol TCP IP address specified by xmlrpc_interface. The empty string can be used because it binds to all interfaces.
39 ; xmlrpcs_port = 8071 ; The XML-RPC protocol's TCP port is specified via the xmlrpc port variable.

```

- addons\_path — путь к каталогу с дополнительными модулями (аддонами) Системы.
- data\_dir — путь к каталогу, используемому для хранения данных Системы.
- admin\_passwd — мастер-пароль, обеспечивающий административный доступ к Системе.
- list\_db — настройка отображения списка баз данных Системы при входе в систему.
- longpolling\_port — параметр TCP-порта, используемого для обработки событий long polling.
- gevent\_port — TCP-порт для прослушивания соединений Gevent-сервера Системы.
- max\_cron\_threads — максимальное количество потоков, одновременно выполняющих задачи планировщика (cron jobs) Системы, обеспечивающее контроль нагрузки на сервер.
- workers — количество процессов, обрабатывающих запросы пользователей, обеспечивающее масштабируемость и стабильность работы Системы.

**erp.env** - параметры конфигурации подключения odoo сервера к базе данных postgresql

```
# erp settings
# в HOST указывается имя сервиса db из файла compose.yml
HOST=db
# в USER указывается имя пользователя бд. должно совпадать со значением переменной POSTGRES_USER
USER=erp
# в PASSWORD указывается пароль пользователя бд. должно совпадать со значением переменной POSTGRES_PASSWORD
PASSWORD=erp

# db settings
# POSTGRES_DB - имя базы данных по умолчанию. можно оставить postgres
POSTGRES_DB=postgres
# POSTGRES_USER - имя пользователя базы данных. должно совпадать со значением переменной USER
POSTGRES_USER=erp
# POSTGRES_PASSWORD - пароль пользователя базы данных по умолчанию. должно совпадать со значением переменной POSTGRES_PASSWORD
POSTGRES_PASSWORD=erp
```

- host — имя контейнера db из файла compose.yml
- user — имя пользователя базы данных, должно совпадать со значением переменной postgres\_user
- password — пароль пользователя базы данных, должен совпадать со значением postgres\_password
- postgres\_db — имя базы данных
- postgres\_user — имя пользователя базы данных, должно совпадать со значением переменной user
- postgres\_password — пароль пользователя базы данных, должен совпадать со значением password

**nginx.conf.template** — параметры конфигурации nginx для доступа к ERP через доменное имя.

```

1  server {
2      server_name "~^www.(.*)$" ;
3      return 301 $scheme://$1$request_uri ;
4  }
5
6  server {
7      listen 80 default;
8
9      # increase proxy buffer to handle some Erp web requests
10     proxy_buffers 16 64k;
11     proxy_buffer_size 128k;
12
13     proxy_intercept_errors on;
14     error_page 502 503 504 /50x.html;
15
16     # Specifies the maximum accepted body size of a client request,
17     # as indicated by the request header Content-Length.
18     client_max_body_size 200m;
19
20
21     location / {
22         proxy_pass http://erp:8069;
23         # force timeouts if the backend dies
24         proxy_read_timeout 600s;
25         proxy_next_upstream error timeout invalid_header http_500 http_502 http_503 http_504;
26         proxy_redirect off;
27
28         # set headers
29         proxy_set_header Host $host;
30         proxy_set_header X-Real-IP $remote_addr;
31         proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
32     }
33
34     location /websocket {
35         proxy_http_version 1.1;
36         proxy_set_header Upgrade $http_upgrade;
37         proxy_set_header Connection "upgrade";
38         proxy_set_header Host $host;
39         proxy_set_header X-Real-IP $remote_addr;
40         proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
41         proxy_set_header X-Forwarded-Proto $scheme;
42         proxy_pass http://erp:8072;
43     }
44
45     # cache some static data in memory for 60mins.
46     # under heavy load this should relieve stress on the OpenERP web interface a bit.
47     location ~* /web/static/ {
48         proxy_cache_valid 200 60m;
49         proxy_buffering on;
50         expires 864000;
51         proxy_pass http://erp:8069;
52     }
53 }

```

С описанием параметров файла nginx.conf.template можно ознакомиться в [официальной документации Nginx](#)

## 2.3. Адреса, порты и данные для авторизации использующиеся по умолчанию

По умолчанию веб-сервер Nginx слушает порты 80 (HTTP) и 443 (HTTPS).

Взаимодействие веб-сервера Nginx с Системой осуществляется по внутренней сети Docker с использованием порта 8069 для HTTP-соединений и порта 8072 для WebSocket-соединений.

Система взаимодействует с сервером базы данных PostgreSQL по внутренней сети Docker с использованием порта 5432.

Данные для авторизации по умолчанию задаются в конфигурационном файле `egr.env` (логин: `admin`, пароль: `admin`).

### 3. Шаги по разворачиванию системы

#### 1. Получите экземпляр ПО для экспертизы:

- Скачайте архив с экземпляром ПО для экспертизы по ссылке (запросить у правообладателя ПО);
- Разархивируйте полученный архив в любой удобной директории.

#### 2. Подготовьте конфигурацию Docker.

- Убедитесь, что у вас установлена последняя стабильная версия Docker с compose-плагином;
- Откройте терминал и перейдите в директорию с разархивированным экземпляром ПО;
- Загрузите Docker-образы, сохраненные как .tar архивы: одной командой через подготовленный скрипт:

```
./images/load.sh
```

или несколькими командами вручную:

```
docker load -i images/odoo:18.0-20250401.tar
```

```
docker load -i images/terpo_l10n_ru_doc:latest.tar
```

```
docker load -i images/terpo_translations:latest.tar
```

```
docker load -i images/terpo_web_theme:latest.tar
```

#### 3. Запустите приложение.

- Убедитесь, что порты 80 и 443 открыты и свободны. Запустите контейнеры (в фоновом режиме) командой:

```
docker compose up -d
```

При первом запуске приложению может потребоваться несколько минут для настройки и установки модулей.

#### 4. Проверьте что запущены все сервисы, перечисленные в compose.yml.

| CONTAINER ID | IMAGE                                             | COMMAND                  | CREATED     | STATUS     | PORTS                                                                        | NAMES        |
|--------------|---------------------------------------------------|--------------------------|-------------|------------|------------------------------------------------------------------------------|--------------|
| 7234888747a7 | registry.nexus.terpo.ru/web/erp-infra/web/:latest | "/docker-entrypoint..."  | 3 hours ago | Up 3 hours | 0.0.0.0:80->80/tcp, [::]:80->80/tcp, 0.0.0.0:443->443/tcp, [::]:443->443/tcp | erpdev-web-1 |
| ffe04e5ba13  | registry.nexus.terpo.ru/web/erp-infra/erp/:latest | "/entrypoint.sh odoo"    | 3 hours ago | Up 3 hours | 127.0.0.1:8069->8069/tcp, 127.0.0.1:8071-8072->8071-8072/tcp                 | erpdev-erp-1 |
| 67825bef58fa | postgres:16.0-alpine                              | "docker-entrypoint.s..." | 3 hours ago | Up 3 hours | 5432/tcp                                                                     | erpdev-db-1  |

После выполнения этих шагов ваше приложение будет успешно развернуто и доступно по адресу <http://localhost>

Для смены адреса нужно изменить значение с `listen 80 default;` на `listen 80 example.com;` в файле `nginx/templates`, где `example.com` – требуемое новое доменное имя, и перезапустить web сервис командой:

```
docker compose up --build -d web
```

## 5. Остановка приложения.

Для того чтобы остановить и удалить контейнеры, а также сети, которые были созданы с помощью Docker compose, выполните следующие действия:

- Перейдите в директорию, где находится ваш файл `compose.yml`.
- Выполните команду:  
*docker-compose down*

## 3.1. Перезапуск системы

Система автоматически перезапускается после незапланированного отключения (например, из-за отключения электроэнергии).

Для ручного перезапуска системы необходимо выполнить команду:

```
docker compose restart web erp db
```

## 3.2. Штатный механизм создания резервных копий системы

### Создание резервной копии через менеджера баз данных

Для создания резервной копии системы через менеджера баз данных необходимо:

1. Изменить в файле `erp.conf` параметр:  
*list\_db = False на list\_db = True*
2. Перезапустить ERP сервис введя в командную строку сервера команду:  
*docker compose restart erp*
3. Перейти по url-адресу:  
*/web/database/manager*
4. Нажать кнопку Backup напротив нужной базы данных;

5. Ввести мастер пароль, указанный в параметре `admin_passwd` в `erp.conf`;
6. Скачать zip-архив с копией БД.

Для импорта дампа базы данных через менеджера баз данных необходимо:

1. Перейдите по url-адресу:  
`/web/database/manager`
2. Нажмите кнопку: Restore Database;
3. Выберите zip-архив с дампом;
4. Введите мастер пароль;
5. Введите имя новой БД;
6. Поменяйте в файле `erp.conf` параметр:  
`list_db = True` на `list_db = False`;
7. Перезапустите сервис ERP, введя в командную строку сервера команду:  
`docker compose restart erp`.

## Создание резервной копии через командную строку

Для создания резервной копии системы через командную строку необходимо:

1. Убедиться, что в файле `erp.conf` установлены параметры:  
`db_host = db`  
`db_user = erp`  
`db_password = erp`
2. Выполнить выгрузку базы данных в файловую систему контейнера, выполнив следующую команду:  
`docker compose exec erp odoo db -c /etc/erp/erp.conf dump <имя старой бд> /var/lib/erp/dump.zip`
3. Скопировать файл дампа из файловой системы контейнера в файловую систему хоста:  
`docker compose cp erp:/var/lib/erp/dump.zip dump.zip`

Для импорта дампа базы данных через командную строку необходимо:

1. Скопировать файл дампа из файловой системы хоста в файловую систему контейнера:

```
docker compose cp dump.zip erp:/var/lib/erp/dump.zip
```

2. Импортировать дамп в контейнере командой:

```
docker compose exec erp odoo db -c /etc/erp/erp.conf load <имя новой бд>  
/var/lib/erp/dump.zip
```

## Ссылки

- [Официальная документации Nginx](#)
- [Официальная документации Odoo](#)