ООО «Мелвад»
390026, г. Рязань, ул. Ленинского Комсомола, д. 106, кв. 18
ИНН 6234193052, КПП 623401001
ОГРН 1206200011907
тел. 8 (920) 961-27-77
support@melvad.com

Инструкция по установке Melvad DMP

Оглавление

1. Общие сведения 3
1.1. Назначение документа 3
1.2. Состав ПО 3
1.3. Комплект поставки 3
2. Требования к системе 4
2.1. Аппаратные требования 4
2.2. Программные требования 4
2.3. Сеть и доменные имена 4
3. Подготовка сервера 5
3.1. Установка Docker 5
3.2. Параметры ядра для ScyllaDB 5
3.3. Метрики сервера (необязательно) 5
3.4. Доступ к реестру образов и комплект файлов 5
4. Установка 5
4.1. Файлы настроек 5
4.2. Базы данных PostgreSQL и ScyllaDB 6
4.3. API 6
4.4. Сервис матчинга 6
4.5. Веб-интерфейс 7
4.6. nginx и сертификат HTTPS 7
4.7. Мониторинг 7
4.8. Первый администратор 8
5. Проверка установки 8
6. Обновление 8
7. Резервное копирование и удаление 8
8. Развёртывание проверочного стенда 9
9. Техническая поддержка 9

1. Общие сведения

1.1. Назначение документа

Документ описывает установку (развёртывание) программного обеспечения Melvad DMP на сервере: требования к системе, подготовку сервера, порядок установки компонентов, первичную настройку и проверку работоспособности.
Melvad DMP распространяется как интернет-сервис (SaaS). Пользователям (рекламодателям, администраторам) устанавливать ПО не требуется: работа ведётся в браузере по адресу веб-интерфейса, партнёры (SSP, DSP, DMP) обращаются к сервису по HTTP. Развёртывание экземпляра выполняет ООО «Мелвад» на своих серверах или администратор заказчика при размещении ПО в инфраструктуре заказчика — по этому документу.
По этой инструкции развёрнут экземпляр для экспертной проверки (проверочный стенд, см. раздел 8 и документ «Инструкция по эксплуатации проверочного стенда Melvad DMP»).

1.2. Состав ПО

Все компоненты работают в Docker-контейнерах в сети хоста (network_mode: host), внутренние порты открыты только на 127.0.0.1; снаружи доступны только порты 80 и 443 nginx.

Компонент	Контейнер	Образ	Назначение
Веб-интерфейс (UI)	`ui-<env>`	`dmp/ui`	веб-интерфейс на Vue 3 (nginx), проксирует `/api/` в API
API	`api-<env>`	`dmp/api`	управление пользователями, SSP, DSP, DMP, сегментами, статистика; миграции схем баз
Сервис матчинга	`matching-<env>-dmp` (и реплика `matching-<env>-replica-dmp`)	`dmp/matching`	синхронизация кук, пиксели сегментов, выдача сегментов партнёрам
PostgreSQL 17.5	`dmp-postgres-<env>`	`postgres:17.5`	пользователи, справочники, сегменты, сессии
ScyllaDB 2026.1	`dmp-scylla-<env>`	`scylladb/scylla:2026.1.9`	пары синхронизации кук, участники сегментов
nginx	`dmp-nginx-<env>`	`nginx:1.29-alpine`	HTTPS-публикация доменов, сертификат Let's Encrypt
Мониторинг	`dmp-prometheus-<env>`, `dmp-grafana-<env>`, `dmp-postgres-exporter-<env>`, `dmp-cadvisor-<env>`, `dmp-blackbox-<env>`	`dmp/api/monitoring/*`	метрики (90 дней) и дашборды

Здесь и далее `<env>` — имя окружения (переменная `ENV_NAME`), например `dev` или `prod1`.

1.3. Комплект поставки

Комплект поставки состоит из:

- Docker-образов компонентов в реестре образов ООО «Мелвад» `gitlab.melvad.com:5050` (таблица ниже). Доступ к реестру (логин и токен) выдаёт техническая поддержка ООО «Мелвад»;

- комплекта файлов развёртывания: compose-файлы, скрипты запуска и инициализации, примеры настроек. Комплект — копия каталогов `deployments` проектов dmp/api, dmp/matching и dmp/ui; на проверочном стенде он размещён в каталоге `/opt/dmp`, заказчику передаётся архивом.

Образ	Пример версии	Компонент
`gitlab.melvad.com:5050/dmp/api/<ветка>:<версия>`	`develop:7101`	API, утилиты `migrate`, `useradd`
`gitlab.melvad.com:5050/dmp/matching/<ветка>:<версия>`	`develop:7092`	сервис матчинга
`gitlab.melvad.com:5050/dmp/ui/<ветка>:<версия>`	`develop:5717`	веб-интерфейс
`gitlab.melvad.com:5050/dmp/api/monitoring/{prometheus,grafana,postgres-exporter,cadvisor,blackbox}:<версия>`	`7101`	мониторинг

Состав комплекта файлов (`/opt/dmp`):

Путь	Содержимое
`api/deployments/infra`	PostgreSQL и ScyllaDB: `docker-compose.yml`, `up.sh` — запуск и инициализация
`api/deployments/postgres`, `api/deployments/scylla`	начальная схема: база, роли, keyspace
`api/deployments/docker-compose.yml`	сервис API
`api/deployments/proxy`	nginx и сертификат Let's Encrypt: `up.sh`, `docker-compose.yml`
`api/deployments/monitoring`	Prometheus, Grafana, экспортеры
`matching/deployments/docker-compose.yml`	сервис матчинга
`ui/deployments/docker-compose.yml`	веб-интерфейс
`env/*.env.example`	примеры файлов настроек

2. Требования к системе

2.1. Аппаратные требования

Параметр	Минимум (проверочный стенд)	Рекомендуется для промышленной нагрузки
Процессор	4 ядра x86-64 с поддержкой SSE4.2	от 16 ядер
Оперативная память	8 ГБ	от 64 ГБ
Диск	60 ГБ SSD	NVMe; данные ScyllaDB на отдельном диске с файловой системой XFS
Сеть	100 Мбит/с, публичный IPv4-адрес	1 Гбит/с

2.2. Программные требования

- 64-битная ОС Linux; проверено на Ubuntu Server 24.04 LTS;
- Docker Engine 27 или новее с плагином Docker Compose v2;
- утилиты `sh`, `sed`, `wget`, `openssl` (есть в стандартной установке Ubuntu);
- параметр ядра `fs.aio-max-nr` не меньше 1048576 (требование ScyllaDB).

2.3. Сеть и доменные имена

- Четыре доменных имени с A-записями на публичный IP сервера: веб-интерфейс, API, сервис матчинга, Grafana. В примерах: `ui.example.ru`, `api.example.ru`, `match.example.ru`, `grafana.example.ru`.

- Входящие соединения на порты 80 и 443 сервера открыты (сертификат Let's Encrypt выпускается проверкой HTTP-01 на порту 80).
- Исходящие соединения на порты 80 и 443 (Let's Encrypt, загрузка образов и пакетов) и 5050 (реестр образов `gitlab.melvad.com:5050`).

3. Подготовка сервера

Все команды выполняются от пользователя root.

3.1. Установка Docker

```
apt-get update && apt-get install -y ca-certificates curl
install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
echo "deb [arch=amd64 signed-by=/etc/apt/keyrings/docker.asc]
https://download.docker.com/linux/ubuntu noble stable" >
/etc/apt/sources.list.d/docker.list
apt-get update && apt-get install -y docker-ce docker-ce-cli containerd.io docker-compose-
plugin
docker version && docker compose version
```

3.2. Параметры ядра для ScyllaDB

```
echo 'fs.aio-max-nr = 1048576' > /etc/sysctl.d/90-scylla.conf
sysctl -p /etc/sysctl.d/90-scylla.conf
```

3.3. Метрики сервера (необязательно)

Для графиков загрузки сервера в Grafana установите node-exporter на 127.0.0.1:9100:

```
apt-get install -y --no-install-recommends prometheus-node-exporter
echo 'ARGS="--web.listen-address=127.0.0.1:9100"' > /etc/default/prometheus-node-exporter
systemctl restart prometheus-node-exporter
```

3.4. Доступ к реестру образов и комплект файлов

Войдите в реестр образов логином и токеном, выданными технической поддержкой:

```
docker login gitlab.melvad.com:5050
```

Распакуйте комплект файлов развёртывания в `/opt/dmp`:

```
mkdir -p /opt && tar -xzf dmp-deploy.tar.gz -C /opt
ls /opt/dmp
```

4. Установка

Компоненты устанавливаются в порядке: базы данных → API (применяет миграции схем) → сервис матчинга → веб-интерфейс → nginx с сертификатом → мониторинг → первый администратор. Каждый шаг можно повторять: скрипты и `docker compose up -d` идемпотентны.

4.1. Файлы настроек

Скопируйте примеры и заполните значения. Пароли сгенерируйте командой `openssl rand -hex 16`.

```
cd /opt/dmp/env
cp infra.env.example infra.env && cp api.env.example api.env && cp matching.env.example
matching.env
chmod 600 *.env
```

Файл	Обязательные параметры
infra.env	INFRA_POSTGRES_PASSWORD — пароль суперпользователя PostgreSQL; POSTGRES_PORT, SCYLLA_PORT, SCYLLA_METRICS_PORT — порты на 127.0.0.1; SCYLLA_ARGS, SCYLLA_MEM_LIMIT — ресурсы ScyllaDB (по умолчанию 1 ядро и 1 ГБ)
api.env	API_POSTGRES_PASSWORD (роль matching_app), API_POSTGRES_MIGRATOR_PASSWORD (роль matching_migrator), API_SWAGGER_HOST — домен API; порты и адреса баз — как в примере
matching.env	MATCHING_POSTGRES_PASSWORD — тот же пароль роли matching_app; MATCHING_SWAGGER_HOST — домен матчинга; RESYNC_DAYS — срок свежести пары SSP в днях; UBS_UBU_TIMEOUT — таймаут методов /match/ubs и /match/ubu (например 10ms)

Задайте общие переменные сеанса (имя окружения, домены, IP сервера, версии образов):

```
export ENV_NAME=dev
export API_DOMAIN=api.example.ru SYNC_DOMAIN=match.example.ru UI_DOMAIN=ui.example.ru
GRAFANA_DOMAIN=grafana.example.ru
export SYNC_IP=<публичный IP сервера>
export REG=gitlab.melvad.com:5050/dmp
export API_IMAGE=$REG/api/develop:7101 MATCHING_IMAGE=$REG/matching/develop:7092
UI_IMAGE=$REG/ui/develop:5717 MON_TAG=7101
```

Для имени окружения в комплекте должны быть файлы api/deployments/scylla/keyspace_<ENV_NAME>.cql и api/deployments/monitoring/prometheus/prometheus-<ENV_NAME>.yml (поставляются для dev и prod1; для нового окружения они создаются по образцу и включаются в образ мониторинга).

4.2. Базы данных PostgreSQL и ScyllaDB

```
cd /opt/dmp/api
sh deployments/infra/up.sh /opt/dmp/env/infra.env /opt/dmp/env/api.env
```

Скрипт запускает контейнеры dmp-postgres-<env> и dmp-scylla-<env>, дожидается их готовности, создаёт базу matching, роли matching_app, matching_migrator, dmp_monitoring (её пароль сохраняется в томе dmp-monitoring-secrets-<env>) и keyspace matching. Ожидаемый вывод заканчивается строками:

```
postgres: database matching and roles created
postgres: role dmp_monitoring ready, its password is in the volume dmp-monitoring-secrets-
dev
scylla: keyspace matching ready
```

4.3. API

```
cd /opt/dmp/api/deployments
cp /opt/dmp/env/api.env .env
printf
'\nIMAGE=%s\nENV_NAME=%s\nSYNC_DOMAIN=%s\nAPI_SWAGGER_HOST=%s\nAPI_PROMETHEUS_URL=http://12
7.0.0.1:19090\n' "$API_IMAGE" "$ENV_NAME" "$SYNC_DOMAIN" "$API_DOMAIN" >> .env
docker compose -p api-$ENV_NAME -f docker-compose.yml up -d
docker logs --tail 20 api-$ENV_NAME
```

При старте с API_MIGRATE_ON_START=true контейнер применяет миграции PostgreSQL и ScyllaDB. Контейнер должен оставаться в состоянии running без перезапусков: docker inspect -f '{{.State.Status}} {{.RestartCount}}' api-dev → running 0.

4.4. Сервис матчинга

```
cd /opt/dmp/matching/deployments
```

```
cp /opt/dmp/env/matching.env .env
printf '\nIMAGE=%s\nENV_NAME=%s\nSYNC_DOMAIN=%s\n' "$MATCHING_IMAGE" "$ENV_NAME"
"$SYNC_DOMAIN" >> .env
COMPOSE_PROFILES=cluster docker compose -p matching-$ENV_NAME -f docker-compose.yml up -d
docker ps --filter name=matching-$ENV_NAME
```

Профиль `cluster` запускает второй экземпляр (`matching-<env>-replica-dmp`, порт 18084); без него запускается один экземпляр. Оба контейнера должны перейти в состояние healthy.

4.5. Веб-интерфейс

```
cd /opt/dmp/ui/deployments
printf 'IMAGE=%s\nENV_NAME=%s\n' "$UI_IMAGE" "$ENV_NAME" > .env
docker compose -p ui-$ENV_NAME -f docker-compose.yml up -d
```

Контейнер слушает 127.0.0.1:18083 и проксирует запросы `/api/` в API (127.0.0.1:18082).

4.6. nginx и сертификат HTTPS

```
cd /opt/dmp/api
OWN_NGINX=true MATCHING_REPLICA_PORT=18084 HELPER_IMAGE=$API_IMAGE sh
deployments/proxy/up.sh
```

Скрипт запускает контейнер `dmp-nginx-<env>` на портах 80 и 443, выпускает сертификат Let's Encrypt на четыре домена (клиент lego, файлы в `/home/nginx/certs/dmp`) и публикует сервисы:

Адрес	Протокол	Назначение
`https://<UI_DOMAIN>`	https	веб-интерфейс
`https://<API_DOMAIN>`	https	API, документация `/swagger/index.html`
`https://<SYNC_DOMAIN>`	https	сервис матчинга (браузеры пользователей)
`http://<SYNC_DOMAIN>/match/ubs`, `/match/ubu`; `http://<SYNC_IP>/match/ubs`, /match/ubu	только http	запросы серверов партнёров (по https — ответ 403)
`https://<GRAFANA_DOMAIN>`	https	мониторинг

Сертификат продлевается повторным запуском той же команды (продление выполняется за 30 дней до окончания срока). Добавьте запуск в планировщик, например ежедневно в 05:00:

```
cat > /etc/cron.d/dmp-proxy <<'EOF'
0 5 * * * root cd /opt/dmp/api && ENV_NAME=dev API_DOMAIN=api.example.ru
SYNC_DOMAIN=match.example.ru UI_DOMAIN=ui.example.ru GRAFANA_DOMAIN=grafana.example.ru
SYNC_IP=<IP> OWN_NGINX=true MATCHING_REPLICA_PORT=18084 HELPER_IMAGE=<образ API> sh
deployments/proxy/up.sh >> /var/log/dmp-proxy.log 2>&1
EOF
```

4.7. Мониторинг

```
cd /opt/dmp/api
export PROMETHEUS_IMAGE=$REG/api/monitoring/prometheus:$MON_TAG
GRAFANA_IMAGE=$REG/api/monitoring/grafana:$MON_TAG
export POSTGRES_EXPORTER_IMAGE=$REG/api/monitoring/postgres-exporter:$MON_TAG
CADVISOR_IMAGE=$REG/api/monitoring/cadvisor:$MON_TAG
BLACKBOX_IMAGE=$REG/api/monitoring/blackbox:$MON_TAG
export POSTGRES_PORT=25432 COMPOSE_PROFILES=exporters
docker compose -p dmp-monitoring-$ENV_NAME -f deployments/monitoring/docker-compose.yml up
-d
sh deployments/monitoring/check.sh
```

Prometheus (127.0.0.1:19090) хранит метрики 90 дней, Grafana (127.0.0.1:13000) доступна по `https://<GRAFANA_DOMAIN>`. Пароль пользователя admin Grafana: `docker exec dmp-grafana-dev cat /var/lib/grafana/admin-password`.

4.8. Первый администратор

```
docker exec -e USERADD_PASSWORD='<пароль>' api-$ENV_NAME ./useradd -name "Администратор" -
role admin -email admin@example.ru
```

Команда создаёт пользователя с ролью «Администратор» и печатает его идентификатор. Под этим email и паролем выполняется вход в веб-интерфейс; остальных пользователей администратор создаёт в веб-интерфейсе (раздел «Пользователи»).

5. Проверка установки

Все контейнеры запущены и не перезапускаются:

```
docker ps --format '{{.Names}}  {{.Status}}' | sort
```

Ожидаемый список: `api-<env>`, `matching-<env>-dmp`, `matching-<env>-replica-dmp`, `ui-<env>`, `dmp-postgres-<env>`, `dmp-scylla-<env>`, `dmp-nginx-<env>`, `dmp-prometheus-<env>`, `dmp-grafana-<env>`, `dmp-postgres-exporter-<env>`, `dmp-cadvisor-<env>`, `dmp-blackbox-<env>`. Доступность сервисов:

```
curl -s -o /dev/null -w '%{http_code}\n' https://ui.example.ru/                 # 200
curl -s https://api.example.ru/ping                                              # 200
curl -s https://match.example.ru/ping                                            # 200
curl -s 'http://match.example.ru/match/ubs?user_id=cHJvYmU%3D&user_type=1'       # 200,
{"segments":[]}
curl -s -o /dev/null -w '%{http_code}\n' 'https://match.example.ru/match/ubs'    # 403
curl -s https://grafana.example.ru/api/health                                    # 200
```

Откройте `https://<UI_DOMAIN>` в браузере и войдите под первым администратором (п. 4.8). Сценарий полной проверки работы (создание SSP, DSP, сегментов, синхронизация кук) приведён в документе «Руководство пользователя Melvad DMP».

6. Обновление

Новая версия устанавливается заменой образа и повторным запуском `docker compose up -d` компонента; миграции схем API применяет при старте. Порядок: API → сервис матчинга → веб-интерфейс.

```
cd /opt/dmp/api/deployments && sed -i "s#^IMAGE=.*#IMAGE=$REG/api/develop:<новая версия>#"
.env
docker compose -p api-$ENV_NAME -f docker-compose.yml up -d
```

7. Резервное копирование и удаление

Данные хранятся в Docker-томах:

Том	Содержимое
`dmp-infra-<env>_postgres`	база PostgreSQL
`dmp-infra-<env>_scylla`	данные ScyllaDB
`dmp-monitoring-<env>_prometheus`	метрики Prometheus
`dmp-monitoring-<env>_grafana`	данные Grafana
`dmp-monitoring-secrets-<env>`	пароль роли мониторинга PostgreSQL

Резервная копия базы PostgreSQL и снимок ScyllaDB:

```
docker exec dmp-postgres-dev pg_dump -U postgres -Fc matching > /root/matching-$(date
+%F).dump
docker exec dmp-scylla-dev nodetool snapshot matching
```

Удаление ПО (тома с данными сохраняются, удаляются командой `docker volume rm`):

```
for p in ui matching api dmp-monitoring dmp-proxy dmp-infra; do docker compose -p $p-
$ENV_NAME down; done
```

8. Развёртывание проверочного стенда

Экземпляр для экспертной проверки развёрнут по этой инструкции на отдельной виртуальной машине `dmp-demo` (Ubuntu 24.04 LTS, 4 vCPU, 8 ГБ ОЗУ, 60 ГБ) на сервере ООО «Мелвад» в дата-центре Selectel (г. Санкт-Петербург), имя окружения `dev`. Установка и обновления стенда выполняются автоматически конвейером CI/CD GitLab (`gitlab.melvad.com`, также в Selectel, г. Санкт-Петербург) теми же командами через Docker API виртуальной машины:

Задание CI/CD	**Пункт инструкции**
`deploy-infra-dev` (dmp/api)	4.2
`deploy-dev` (dmp/api)	4.3
`deploy-dev` (dmp/matching)	4.4
`deploy-dev` (dmp/ui)	4.5
`deploy-proxy-dev` (dmp/api, также ежедневно по расписанию)	4.6
`deploy-monitoring-dev` (dmp/api)	4.7
`ui-login-dev` (dmp/api)	4.8

Комплект файлов развёртывания на стенде — `/opt/dmp`, доступ к стенду описан в документе «Инструкция по эксплуатации проверочного стенда Melvad DMP».

9. Техническая поддержка

По вопросам установки и настройки обращайтесь в техническую поддержку ООО «Мелвад»:

- Стуколкин Александр, e-mail: a.stukolkin@melvad.com, телефон: +7 (920) 961-27-77;
- support@melvad.com.