# Руководство администратора

Программа «NORA — многопротокольный реестр артефактов»

Версия документа: 2.1 · Дата: 2026-06-05 · Правообладатель: ООО «ТАИАРС» · Версия программы: 1.0.0

## 1. Общие сведения

NORA — многопротокольный реестр артефактов, предназначенный для хранения, кэширования и распространения программных компонентов. Программа обеспечивает централизованное управление зависимостями при разработке и сборке программного обеспечения.

### 1.1. Назначение

- Хранение и раздача артефактов по 16 протоколам: Docker (OCI), Helm OCI, npm, Maven, PyPI, Cargo, Go, NuGet, RubyGems, Terraform, Ansible Galaxy, Pub (Dart/Flutter), Conan, APT (deb), YUM (rpm) и Raw.
- Проксирование и кэширование внешних репозиториев с сохранением полученных артефактов в локальное хранилище на файловой системе сервера — для ускорения сборок и обеспечения доступности при отсутствии соединения с сетью Интернет.
- Курирование зависимостей: блокировка, разрешение и изоляция пространств имён.
- Контроль целостности артефактов посредством верификации SHA-256.
- Аудит и протоколирование всех операций.

### 1.2. Системные требования

| Параметр | Минимальные | Рекомендуемые |
|---|---|---|
| ОС | Linux (amd64, arm64), ядро 4.15+ | Astra Linux, РЕД ОС, ALT Linux (а также Debian 11+, Ubuntu 22.04+, RHEL 8+) |
| ЦПУ | 1 ядро | 2+ ядра |
| ОЗУ | 256 МБ | 1+ ГБ |
| Диск | 1 ГБ | 50+ ГБ (зависит от объёма хранимых артефактов) |
| Сеть | TCP-порт (по умолчанию 4000) | — |

### 1.3. Зависимости

Программа поставляется как единый статически слинкованный исполняемый файл. Внешние зависимости отсутствуют. Перечень библиотек, включённых в состав программы, приведён в файле `nora.cdx.json` (формат CycloneDX).

## 2. Установка

### 2.1. Автоматическая установка

```
curl -fsSL https://getnora.io/install.sh | bash
```

Скрипт выполняет следующие действия:

- Определяет архитектуру процессора (amd64 или arm64).
- Загружает исполняемый файл с git.getnora.io (репозиторий правообладателя).
- Создаёт системного пользователя nora.
- Создаёт каталоги: `/etc/nora/`, `/var/lib/nora/`, `/var/log/nora/`.
- Устанавливает файл конфигурации `/etc/nora/nora.env`.
- Устанавливает и активирует systemd-сервис.

### 2.2. Ручная установка

```
wget https://git.getnora.io/nora/nora/releases/download/latest/nora-linux-amd64
chmod +x nora-linux-amd64
mv nora-linux-amd64 /usr/local/bin/nora

useradd --system --shell /usr/sbin/nologin --home-dir /var/lib/nora --create-home nora
mkdir -p /etc/nora /var/lib/nora /var/log/nora
chown nora:nora /var/lib/nora /var/log/nora

cp dist/nora.service /etc/systemd/system/
systemctl daemon-reload
systemctl enable nora
```

### 2.3. Установка из Docker-образа

```
docker run -d \
  --name nora \
  -p 4000:4000 \
  -v nora-data:/data \
  git.getnora.io/nora/nora:latest
```

### 2.4. Установка из deb-пакета (Debian, Ubuntu, Astra Linux)

```
echo "deb https://git.getnora.io/api/packages/nora/debian stable main" | \
  sudo tee /etc/apt/sources.list.d/nora.list
sudo apt update && sudo apt install nora
```

### 2.5. Установка из rpm-пакета (РЕД ОС, ALT Linux, RHEL, CentOS)

```
sudo yum-config-manager --add-repo https://git.getnora.io/api/packages/nora/rpm/stable.repo
sudo yum install nora
```

### 2.6. Установка через Helm (Kubernetes)

```
helm install nora oci://git.getnora.io/nora/helm/nora --version 1.0.0
```

## 3. Конфигурация

Конфигурация задаётся через переменные окружения, файл config.toml или их комбинацию. Приоритет: переменные окружения > config.toml > значения по умолчанию. Программа поддерживает горячую перезагрузку конфигурации без перезапуска сервиса.

### 3.1. Основные параметры

| Переменная | Описание | По умолчанию |
|---|---|---|
| NORA_HOST | Адрес привязки | 127.0.0.1 |
| NORA_PORT | Порт | 4000 |
| NORA_PUBLIC_URL | Внешний URL (для генерации ссылок) | — |
| NORA_STORAGE_PATH | Путь к каталогу хранилища на файловой системе | data/storage |
| NORA_STORAGE_MODE | Тип хранилища: local или s3 | local |
| NORA_BODY_LIMIT_MB | Максимальный размер тела запроса (МБ) | 2048 |

### 3.2. Аутентификация

| Переменная | Описание | По умолчанию |
|---|---|---|
| NORA_AUTH_ENABLED | Включить аутентификацию | false |
| NORA_AUTH_HTPASSWD_FILE | Путь к файлу htpasswd | users.htpasswd |
| NORA_OIDC_ENABLED | Включить единый вход (OIDC) | false |
| NORA_OIDC_ISSUER_URL | URL провайдера OIDC | — |
| NORA_OIDC_CLIENT_ID | Идентификатор клиента OIDC | — |

Создание пользователя: `htpasswd -Bc /etc/nora/users.htpasswd admin`. Пароли хешируются алгоритмом Argon2id с зануливанием памяти после использования. Роли: admin (полный доступ), write (чтение и запись), read (только чтение, по умолчанию).

### 3.3. Проксирование внешних репозиториев

При проксировании полученные артефакты сохраняются в локальное хранилище (на файловой системе сервера либо в S3-совместимом объектном хранилище). По умолчанию используются следующие внешние реестры:

| Переменная | Описание | По умолчанию |
|---|---|---|
| NORA_NPM_PROXY | URL npm-реестра | https://registry.npmjs.org |

| Переменная | Описание | По умолчанию |
|---|---|---|
| NORA_PYPI_PROXY | URL PyPI-реестра | https://pypi.org/simple/ |
| NORA_MAVEN_PROXIES | Список Maven-репозиториев через запятую | https://repo1.maven.org/maven2 |
| NORA_DOCKER_UPSTREAMS | Docker-реестры (url\|auth,url2) | https://registry-1.docker.io |
| NORA_GO_PROXY | URL Go module proxy | https://proxy.golang.org |
| NORA_GEMS_PROXY | URL RubyGems | https://rubygems.org |
| NORA_NUGET_PROXY | URL NuGet | https://api.nuget.org/v3/index.json |

### 3.4. Ограничение частоты запросов

| Переменная | Описание | По умолчанию |
|---|---|---|
| NORA_RATE_LIMIT_ENABLED | Включить ограничение | true |
| NORA_RATE_LIMIT_GENERAL_RPS | Запросов в секунду (общие) | 100 |
| NORA_RATE_LIMIT_AUTH_RPS | Запросов в секунду (аутентификация) | 1 |
| NORA_RATE_LIMIT_UPLOAD_RPS | Запросов в секунду (загрузка) | 200 |

### 3.5. Курирование (curation)

| Переменная | Описание | По умолчанию |
|---|---|---|
| NORA_CURATION_ENABLED | Включить курирование | false |
| NORA_CURATION_MODE | Режим: blocklist, allowlist | blocklist |

Курирование позволяет блокировать нежелательные пакеты, разрешать только проверенные зависимости и изолировать пространства имён. Режим fail-closed: при ошибке — блокировка.

### 3.6. Автоматический выключатель (circuit breaker)

| Переменная | Описание | По умолчанию |
|---|---|---|
| NORA_CB_ENABLED | Включить circuit breaker | false |
| NORA_CB_THRESHOLD | Количество сбоев до срабатывания | 5 |

При недоступности внешнего реестра circuit breaker предотвращает каскадные ошибки и ускоряет ответы из кэша.

## 4. Управление сервисом

### 4.1. Запуск и остановка

```
systemctl start nora      # Запуск
systemctl stop nora       # Остановка
systemctl restart nora    # Перезапуск
systemctl status nora     # Статус
journalctl -u nora -f     # Просмотр журнала
```

### 4.2. Проверка работоспособности

```
curl http://localhost:4000/health
```

Ответ при нормальной работе содержит статус `healthy`, версию, состояние хранилища и реестров по протоколам.

### 4.3. Метрики (Prometheus)

Эндпоинт `GET /metrics` экспортирует количество запросов, латентность, загрузки и выгрузки по протоколам.

## 5. Резервное копирование и восстановление

```
nora backup --output /backup/nora-$(date +%Y%m%d).tar.gz   # Создание копии
nora restore --input /backup/nora-20260520.tar.gz          # Восстановление
nora gc --dry-run                                          # Просмотр сборки мусора
nora gc                                                    # Удаление осиротевших блобов
```

## 6. Предварительное кэширование (nora mirror)

Команда `nora mirror` позволяет заранее загрузить зависимости через прокси-кэш NORA, обеспечивая доступность артефактов в изолированных средах без доступа к сети Интернет.

```
nora mirror npm --lockfile package-lock.json --registry http://localhost:4000
nora mirror pip --lockfile requirements.txt --registry http://localhost:4000
nora mirror cargo --lockfile Cargo.lock --registry http://localhost:4000
nora mirror npm --packages lodash,express --registry http://localhost:4000
```

Параметры: `--registry` (URL экземпляра NORA), `--concurrency` (число параллельных загрузок, по умолчанию 8), `--all-versions` (загрузить все версии).

## 7. Миграция хранилища

```
nora migrate --from local --to s3 --dry-run   # Просмотр
nora migrate --from local --to s3             # Выполнение
```

## 8. Безопасность

### 8.1. Контроль целостности

При проксировании NORA вычисляет и сохраняет контрольную сумму SHA-256 для каждого артефакта. При повторной выдаче из кэша контрольная сумма проверяется; при расхождении запрос отклоняется, а в журнал записывается предупреждение уровня SECURITY.

### 8.2. Защита от подмены пакетов

- Валидация имён файлов при публикации (защита от обхода каталогов).
- Проверка соответствия имени пакета в URL и теле запроса.
- Иммутабельность версий: повторная публикация той же версии запрещена.

### 8.3. Курирование

- Blocklist — блокировка конкретных пакетов или пространств имён.
- Allowlist — разрешение только проверенных зависимостей.
- Namespace isolation — изоляция областей видимости между командами.

Режим fail-closed: при ошибке проверки пакет блокируется, а не пропускается.

### 8.4. Аудит

Все операции (загрузка, выгрузка, обращения к кэшу, ошибки) фиксируются в файле `audit.jsonl` в каталоге хранилища. Формат — JSON Lines, одна запись на строку.

### 8.5. Усиление systemd

- `NoNewPrivileges=true` — запрет повышения привилегий.
- `ProtectSystem=strict` — файловая система только для чтения, кроме указанных каталогов.
- `ProtectHome=true` — запрет доступа к домашним каталогам.
- `PrivateTmp=true` — изолированный каталог временных файлов.

## 9. Точки подключения (endpoints)

| Протокол | Endpoint | Описание |
|---|---|---|
| Docker / OCI | /v2/ | Docker Registry V2 API |
| Helm | /v2/ (OCI) | Helm-чарты через OCI-протокол |
| npm | /npm/ | npm-реестр (прокси + публикация) |
| Maven | /maven2/ | Maven-репозиторий |
| PyPI | /simple/ | Python Simple API (PEP 503) |
| Cargo | /cargo/ | Cargo-реестр |
| Go | /go/ | Go Module Proxy (GOPROXY) |

| Протокол | Endpoint | Описание |
|---|---|---|
| NuGet | /nuget/ | NuGet V3 API |
| RubyGems | /gems/ | RubyGems API |
| Terraform | /terraform/ | Terraform Provider Registry |
| Ansible | /ansible/ | Ansible Galaxy API |
| Pub | /pub/ | Pub Repository (Dart/Flutter) |
| Conan | /conan/ | Conan v2 API (C/C++) |
| APT | /apt/ | Репозиторий deb-пакетов |
| YUM | /yum/ | Репозиторий rpm-пакетов |
| Raw | /raw/ | Произвольные файлы |
| Мониторинг | /health, /ready, /metrics | Проверка и метрики |
| Интерфейс | /ui/ | Веб-интерфейс управления |
| Документация API | /api-docs | OpenAPI (Swagger UI) |

## 10. Устранение неполадок

**Сервис не запускается:** `journalctl -u nora --no-pager -n 50`. Частые причины: занят порт, недоступен каталог хранилища, ошибка в конфигурации.

**Прокси-кэш не работает:** проверьте доступность внешнего реестра (`curl https://registry.npmjs.org/lodash`), корректность `NORA_NPM_PROXY`, при приватном реестре — `NORA_NPM_PROXY_AUTH`; при включённом circuit breaker проверьте его состояние в `/health`.

**Ошибка целостности:** контрольная сумма кэшированного артефакта не совпала с сохранённой (повреждение ФС либо изменение файла). Удалите повреждённый файл из каталога хранилища — NORA загрузит его заново из внешнего реестра.