

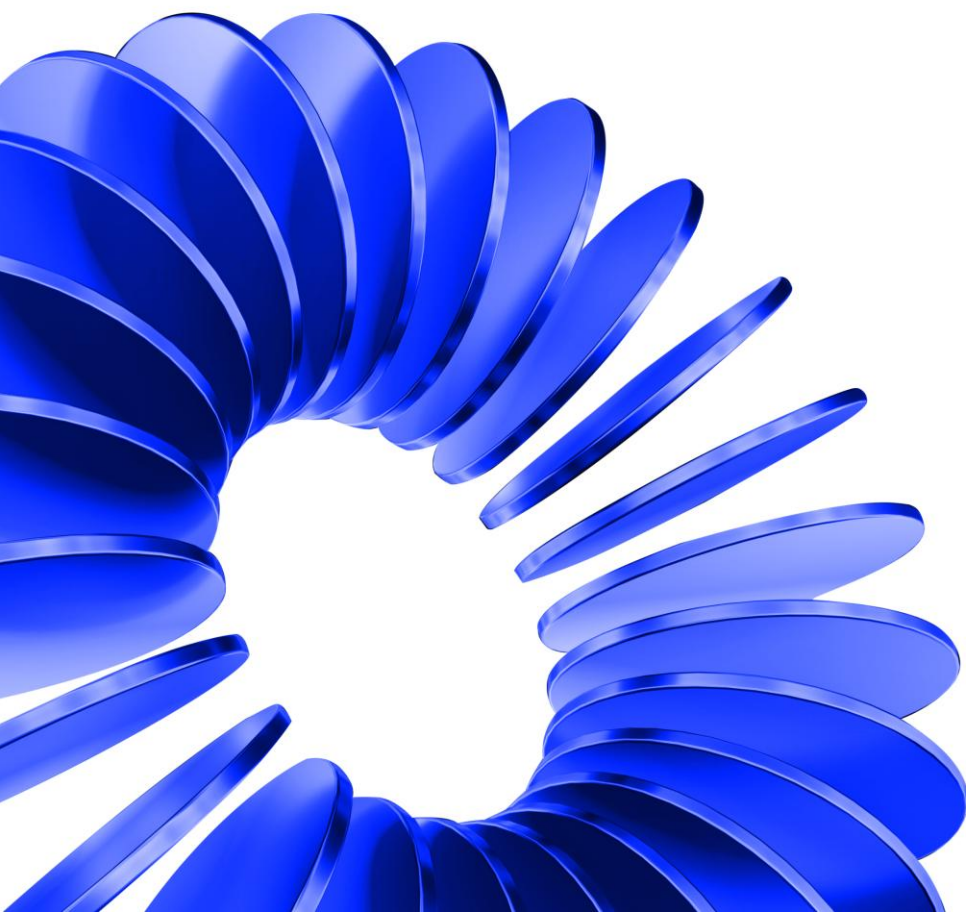
ОБЩЕСТВО С ОГРАНИЧЕННОЙ
ОТВЕТСТВЕННОСТЬЮ «РУСБИТЕХ-АСТРА»

TROK

СИСТЕМА ХРАНЕНИЯ ДАННЫХ TROK

РУКОВОДСТВО ПО УСТАНОВКЕ И ОБНОВЛЕНИЮ

Москва, 2026г.



СОДЕРЖАНИЕ

1.	ОПИСАНИЕ ПРОЕКТА	5
2.	ОПИСАНИЕ КОМПОНЕНТОВ.....	6
3.	ТРЕБОВАНИЯ.....	8
3.1.	Выбор способа установки.....	13
4.	ПОДГОТОВКА ОПЕРАЦИОННОЙ СИСТЕМЫ К УСТАНОВКЕ	14
4.1.	Подготовка операционной системы специального назначения Astra Linux Special Edition к установке ПО TROK	17
4.1.1.	Настройка репозитория для систем с доступом к сети Интернет (настройка для использования ветки репозитория «frozen»)	18
4.1.2.	Перенастройка репозитория для автономных систем (локальные репозитории iso образа ОС).....	20
4.2.	Монтирование образа TROK.....	22
5.	КОНФИГУРАЦИЯ	26
6.	УСТАНОВКА С ИСПОЛЬЗОВАНИЕМ КОМАНДНОЙ СТРОКИ.....	28
7.	УСТАНОВКА В ИНТЕРАКТИВНОМ РЕЖИМЕ (GUI)	30
8.	УСТАНОВКА С ИСПОЛЬЗОВАНИЕМ ПАКЕТОВ.....	33
8.1.	Подготовка локального репозитория	33
8.2.	Установка Controller, Worker или Combined.....	34
8.3.	Установка DRBD при ручном способе.....	35
9.	ДЕПЛОЙ КЛАСТЕРА БЕЗ НА.....	36
9.1.	Требования к управляющему хосту (узел управления Ansible)	36
9.2.	Установка коллекции TROK	37
9.3.	Подготовка структуры проекта.....	38
9.4.	Конфигурация inventory.ini	39
9.5.	Настройка переменных (group_vars/all.yml)	40
9.5.1.	Базовая часть group_vars/all.yml	40
9.5.2.	Опционально: глобальная настройка DRBD на worker-узлах	41
9.5.3.	Пул хранения	43
9.5.4.	iSCSI-экспорт.....	45
9.5.5.	NFS-экспорт	47
9.5.6.	Versity S3-шлюз	50
9.6.	Плейбук	52
9.7.	Запуск деплоя.....	53

9.8.	Проверка после деплоя	54
9.8.1.	Базовые сервисы	54
9.8.2.	Пул хранения	55
9.8.3.	iSCSI	56
9.8.4.	NFS.....	58
9.8.5.	Versity S3-шлюз	60
10.	СЦЕНАРИЙ ДЛЯ ДЕПЛОЯ НА КЛАСТЕРА	62
10.1.	Общая информация о ha_controllers	62
10.2.	Конфигурация файла инвентаря inventory.ini.....	63
10.3.	Настройка переменных (group_vars/all.yml)	64
10.3.1.	База для HA	64
10.3.2.	Пул хранения для HA.....	64
10.3.3.	HA-переменные	65
10.3.4.	Опциональная глобальная настройка DRBD на HA worker-узлах	65
10.4.	Плейбук site.yml	65
10.5.	Диагностика после оборванного первого запуска HA.....	67
10.5.1.	Проверка marker-файлов на всех ha_controllers	67
10.5.2.	Проверка DRBD на всех ha_controllers	68
10.5.3.	Проверка файловых систем и mounts	68
10.5.4.	Проверка сервисов и VIP	69
10.5.5.	Правило принятия решения.....	70
10.6.	Проверка корректности поднятия HA.....	71
10.7.	Возможные ошибки конфигурации	72
10.8.	Дальнейшие шаги после развёртывания HA	72
11.	ЗАВЕРШЕНИЕ УСТАНОВКИ	74
12.	ОСОБЕННОСТИ ПЕРЕУСТАНОВКИ КОМПОНЕНТОВ	78
13.	ЛОГИРОВАНИЕ	79
14.	ПОДГОТОВКА РЕСУРСОВ К ОБНОВЛЕНИЮ ПО	80
15.	РЕЗЕРВНОЕ КОПИРОВАНИЕ КОНФИГУРАЦИЙ СЕРВИСОВ ПЕРЕД ОБНОВЛЕНИЕМ	81
16.	РЕЗЕРВНОЕ КОПИРОВАНИЕ ДАННЫХ ТОМОВ DRBD-РЕСУРСОВ ПЕРЕД ОБНОВЛЕНИЕМ	83
17.	ОБНОВЛЕНИЕ ПАКЕТОВ ИЗ РЕПОЗИТОРИЯ.....	86
18.	ОБНОВЛЕНИЕ HA-КЛАСТЕРА TROK	89

18.1. Перед началом обновления	89
18.1.1. Проверка состояния НА.....	90
18.2. Создание резервной копии	91
18.3. Вариант А — обновление через Ansible	92
18.3.1. Установка коллекции из нового ISO	92
18.3.2. Проверка inventory	93
18.3.3. Запуск обновления	94
18.4. Вариант Б — ручное обновление через APT	94
18.4.1. Подключение репозитория из ISO	95
18.4.2. Остановка и обновление workers	96
18.4.3. Остановка НА control plane	97
18.4.4. Обновление controller-пакетов	98
18.4.5. Переключение НА WebUI на Apache	99
18.4.6. Запуск НА control plane.....	100
18.4.7. Запуск workers	101
18.5. Проверка результата.....	101
18.5.1. Проверка НА controllers.....	101
18.5.2. Проверка WebUI и workers	103
18.6. Действия при ошибке.....	103
19. КРИТИЧЕСКИЕ ПРЕДУПРЕЖДЕНИЯ	106
20. УДАЛЕНИЕ СТРУКТУРНЫХ КОМПОНЕНТОВ SDS	107
21. ПОЛНОЕ УДАЛЕНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ	108
21.1. Удаление пакетов формата deb	108
21.2. Очистка LVM-томов и блочных устройств	108

1. ОПИСАНИЕ ПРОЕКТА

ISO-образ включает программное обеспечение, предназначенное для установки и настройки компонентов TROK.

Программа, которая позволяет установить TROK, реализована на языке C++ с применением фреймворка Qt, стандартной библиотеки и библиотеки YAML-CPP для работы с конфигурационными файлами.

Основные компоненты программного обеспечения TROK разработаны на языке Go с использованием пакетов и основных библиотек protobuf, gRPC и yaml. Пользовательский интерфейс реализован с применением JavaScript и библиотеки React.

2. ОПИСАНИЕ КОМПОНЕНТОВ

Программа поддерживает установку следующих компонентов:

- Controller – центральный контроллер для управления кластером. Отвечает за координацию работы всех узлов, обработку команд администратора, управление конфигурацией и состояние кластера.

- Worker – системная утилита для управления хранилищем. Выполняется на каждом узле хранения, обрабатывает команды контроллера, управляет локальными пулами хранения (LVM, ZFS) и обеспечивает репликацию данных через DRBD.

- WEBUI – веб-интерфейс для управления TROK. Предоставляет графический интерфейс для мониторинга и администрирования кластера, позволяя выполнять основные операции без использования командной строки.

- Harbor — компонент, необходимый для обеспечения возможности предоставления ресурсов по протоколам NFS, iSCSI или SMB. Реализует шлюзы для экспорта томов DRBD в виде сетевых ресурсов, доступных клиентам по стандартным протоколам файлового или блочного доступа. Включает в себя настройку и управление сервисами NFS, Samba и iSCSI Target.

- trok-s3-gw — компонент, обеспечивающий предоставление ресурсов по протоколу S3. Реализует S3-совместимый шлюз, позволяющий использовать хранилище TROK как объектное хранилище. Клиенты могут подключаться к нему через стандартные S3 API-интерфейсы, используя любые S3-совместимые приложения. В комплекте поставляется отдельный веб-интерфейс, позволяющий удобно управлять файлами и папками в объектном хранилище через браузер без использования командной строки и дополнительных клиентских приложений.

Программа установки поддерживает три режима работы:

- Консольный режим: установка с использованием командной строки.
- Режим с ключами: запуск с указанием параметров через аргументы командной строки.

– Интерактивный режим: если программа установки запущена без ключей, пользователю будет предложено выбрать тип установки и дополнительные компоненты через графический интерфейс.

3. ТРЕБОВАНИЯ

Минимальная конфигурация аппаратных средств для корректной работы TROK представлена ниже:

– Операционная система специального назначения Astra Linux Special Edition. Конкретные поддерживаемые версии операционной системы указаны в релизных примечаниях к требуемой версии программного обеспечения. Допускается запуск программного обеспечения на операционной системе уровня защищённости Орел.

– Для установки ПО TROK пользователь должен обладать правами администратора (sudo).

Минимальные требования к аппаратному обеспечению (к серверам хранения данных, предназначенным для сохранения информации, и вычислительным узлам, которые используются для обработки этой информации):

Таблица 1 – Аппаратные требования по типам узлов

Компонент	CPU	RAM	Дисковое пространство	Примечания
Контроллер	1 чип с 4 ядрами и более Частота: от 2 ГГц	Минимальный объем: 8 ГБ Рекомендуемый объем: 64 ГБ и более Тип: DDR4, DDR5, ECC (защита от сбоев)	Минимум 2 накопителя: • Дисковое пространство для операционной системы и программного обеспечения емкостью 128 ГБ и более	2 сетевых порта с пропускной способностью 10 Гбит/с

			(SAS/SATA HDD/SSD); Дисковое пространство для хранения данных TROK: 1 накопитель (и более) с емкостью не менее 128 ГБ (SAS/SATA HDD).	
Worker (с хранилищем)	1 чип с 4 ядрами и более Частота: от 2 ГГц	Минимальный объем: 8 ГБ Рекомендуемый объем: 64 ГБ и более Тип: DDR4, DDR5, ECC (защита от сбоев) Тип: DDR4, DDR5, ECC (защита от сбоев)	2 накопителя: - Дисковое пространство для операционной системы и программного обеспечения емкостью 128 ГБ и более (SAS/SATA HDD/SSD); - Дисковое пространство для хранения данных TROK	2 сетевых порта с пропускной способностью 10 Гбит/с

			(серверы хранения данных): 1 накопитель (и более) с емкостью не менее 1 ТБ (SAS/SATA HDD).	
Diskless-worker	1+ ядро	Минимум: 1 ГБ	Накопитель для операционной системы и программного обеспечения емкостью 128 ГБ и более (SAS/SATA HDD/SSD)	Используется для кворума или как initiator NVMe-oF
Клиент (CLI/GUI)	1 ядро	512 МБ	Накопитель для операционной системы и программного обеспечения емкостью 128 ГБ и более (SAS/SATA HDD/SSD)	Только для управления

Таблица 2 – Список используемых портов

Служба	Порт	Протокол	Назначение	Критичность
--------	------	----------	------------	-------------

SSH	22	TCP	Управление узлами и работа Ansible	Высокая
WebUI	80, 443	TCP	Доступ к веб-интерфейсу TROK	Высокая
Controller API	8080	TCP	Внутренний HTTP API controller, используемый ролями и сервисами TROK	Высокая
Controller	50000, 50001	TCP	Службы управляющего слоя TROK	Высокая
Worker	50002	TCP	Внутренняя коммуникация worker с управляющим слоем	Высокая
DRBD Replication	7000–7999	TCP	Синхронная репликация данных	Максимальная
iSCSI	3260	TCP	Экспорт блочных устройств по iSCSI	Опционально
NFS	2049	TCP, UDP	Доступ к NFS-экспорту	Опционально
RPC bind	111	TCP, UDP	Регистрация RPC-служб NFS	Опционально

Дополнительные RPC-службы NFS	Согласно конфигурации ОС	TCP, UDP	mountd, statd, lockd; при использовании межсетевого экрана порты необходимо зафиксировать в конфигурации ОС	Опционально
NVMe-oF	4420	TCP	Экспорт томов через NVMe/TCP	Опционально
S3 API	7070	TCP	S3-совместимый API	Опционально
WebUI S3	7071	TCP	Веб-интерфейс S3-шлюза	Опционально
Admin API S3	7080	TCP	Административный API S3-шлюза	Опционально

Важно. Таблица определяет базовый набор портов. Окончательный перечень должен быть сверен с Release Notes выбранного выпуска и фактической конфигурацией сервисов.

Диапазон портов DRBD по умолчанию – 7000–7999. Если в кластере используются собственные номера DRBD minor или отдельные диапазоны репликации, правила межсетевого экрана должны соответствовать фактической конфигурации.

Для просмотра слушающих сокетов выполните:

```
sudo ss -tulnp
```

Проверку следует выполнять после установки и запуска соответствующего компонента. Наличие открытого порта само по себе не подтверждает

работоспособность сервиса; после развёртывания выполните функциональные проверки, приведённые в пункте 9.8.

3.1. Выбор способа установки

Перед началом работ выберите один способ установки. Способы, приведённые в руководстве, являются альтернативными и не должны выполняться последовательно на одних и тех же узлах.

Поддерживаются следующие варианты:

- Инсталлятор:
 - консольный режим – пункт 6;
 - графический режим – пункт 7.
- Ручная установка пакетов из локального APT-репозитория – пункт 8.
- Развёртывание кластера с помощью Ansible collection `astra.trok` – пункты 9 и 10.

Важно. Роли `astra.trok.controller`, `astra.trok.worker`, `astra.trok.s3gw` и связанные роли самостоятельно подготавливают локальный репозиторий из ISO и устанавливают необходимые пакеты. Если выбран Ansible, не выполняйте перед ним установку через инсталлятор или пункт 8.

Инсталлятор использует репозиторий `file:///opt/TROK/SDS/apt-repo`. Ansible collection использует собственный каталог репозитория. Одновременное подключение обоих источников не поддерживается.

4. ПОДГОТОВКА ОПЕРАЦИОННОЙ СИСТЕМЫ К УСТАНОВКЕ

Выполните стандартную процедуру установки операционной системы специального назначения Astra Linux Special Edition с соблюдением требований к аппаратному обеспечению и файловым системам. Поддерживаемые версии операционной системы указаны в примечаниях к выпуску (release notes SDS TROK).

Важно. В процессе установки операционной системы, если виртуальный или физический сервер будет использоваться в качестве узла хранения данных, обязательно произведите разметку дисков, заранее выделив необходимое пространство для хранения (отдельный диск для хранения данных).

В ходе установки необходимо выбрать уровень защищённости операционной системы «Орел».

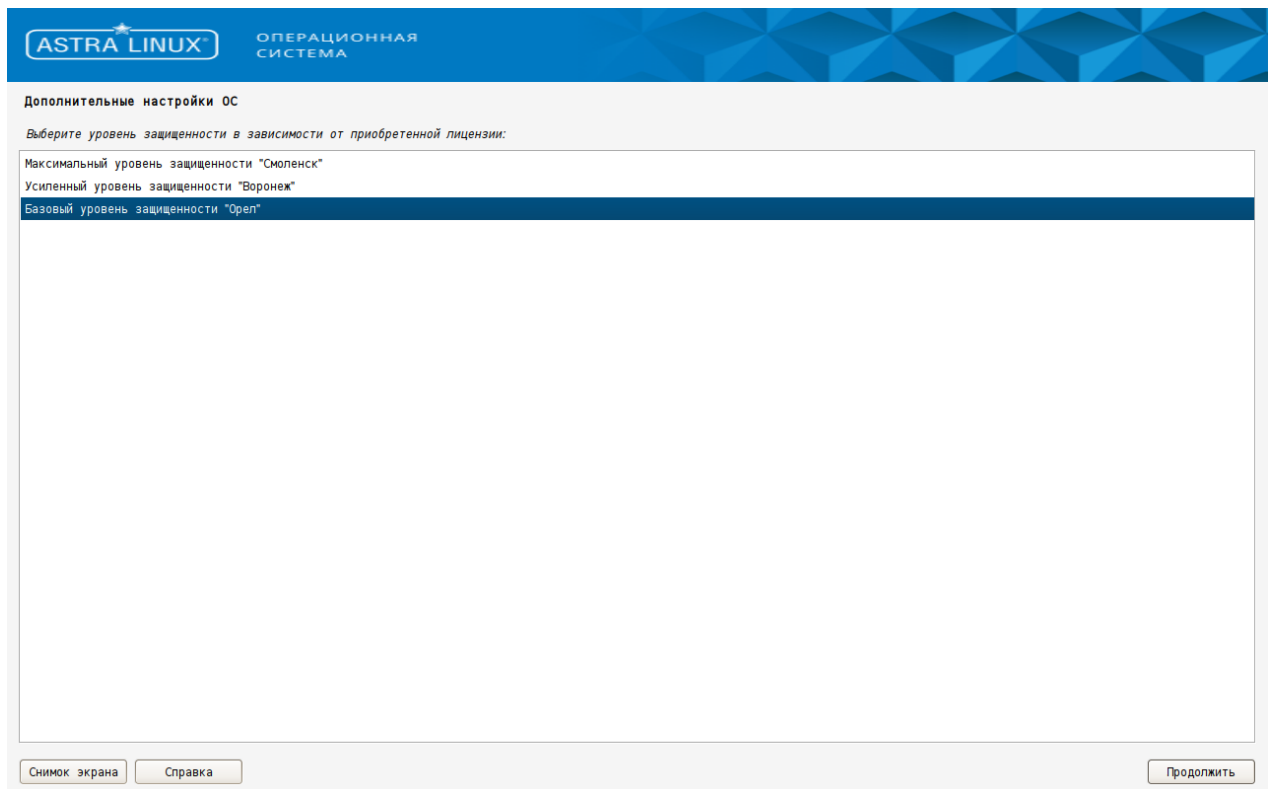


Рисунок 1 – Меню выбора уровня защищенности

В зависимости о типа iso образа, который вы используете, меню выбора выглядит немного иначе.

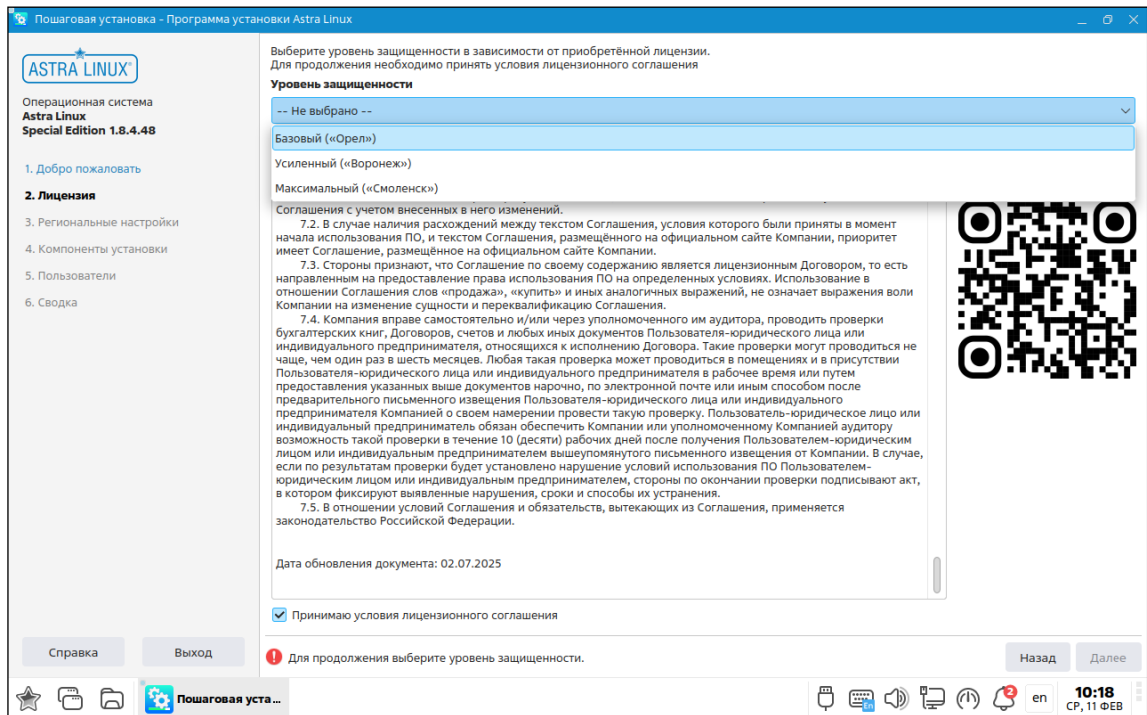


Рисунок 2 – Альтернативное меню выбора уровня защищенности

На последующих этапах установки, если вы устанавливаете Astra Linux SE версии 1.8.x система предложит выбрать одну из двух версий ядра Linux.

Рекомендуется использовать ядро linux-6.1-generic.

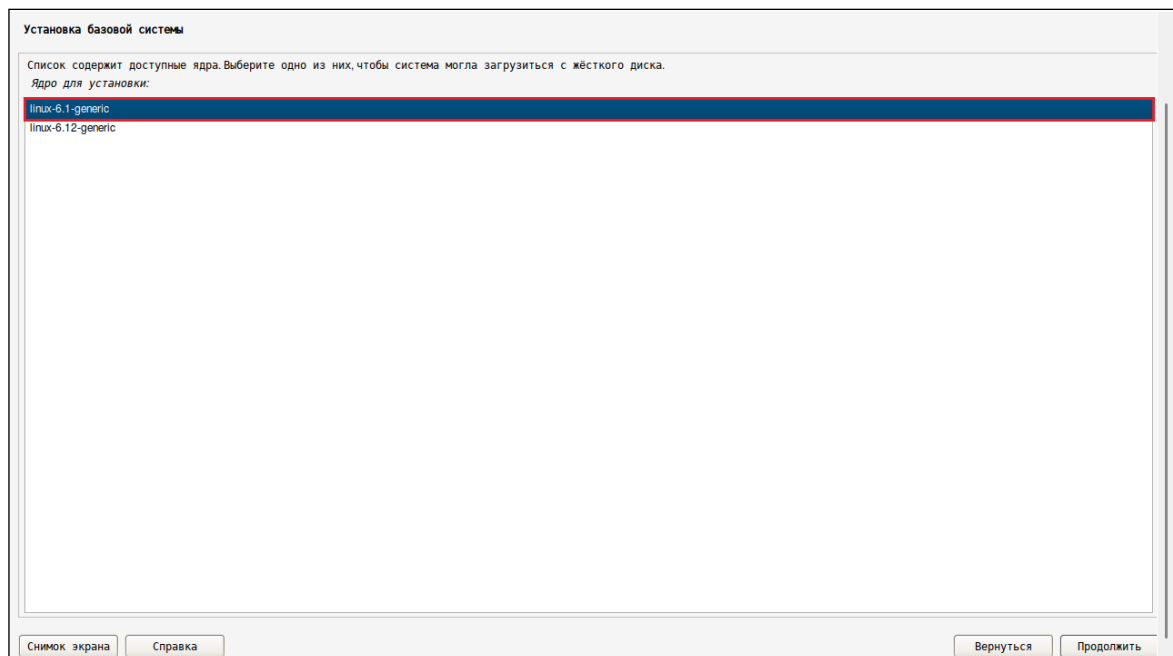


Рисунок 3 – Меню выбора ядра Linux в процессе установки Astra Linux SE 1.8.x

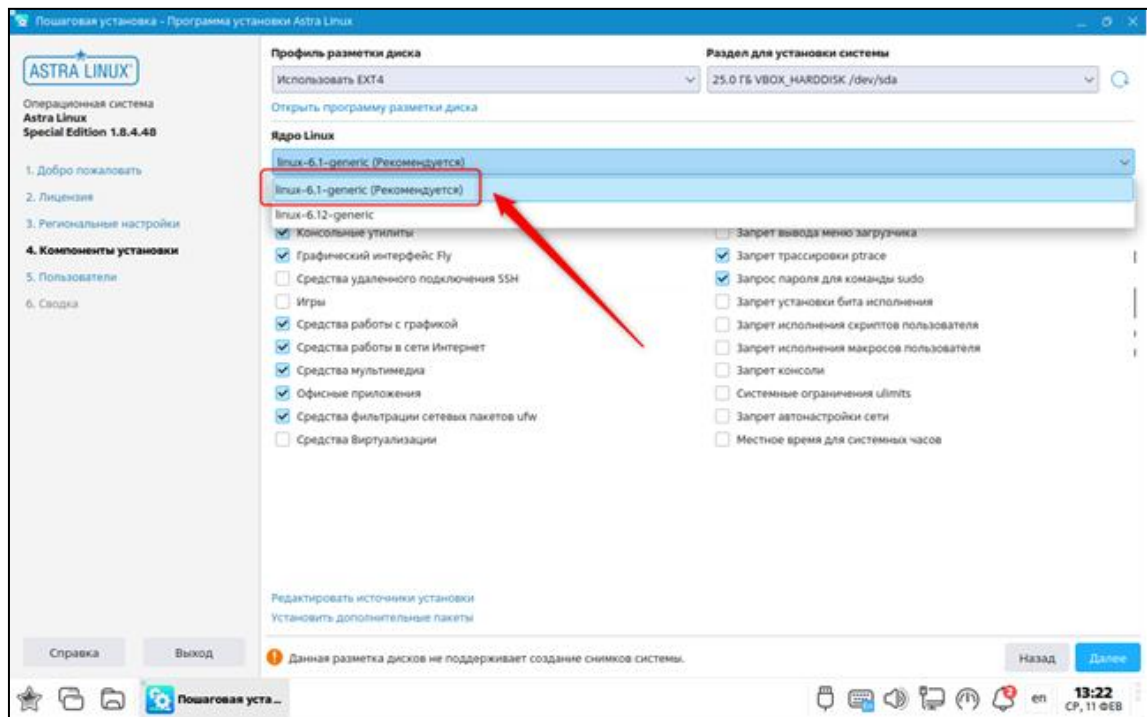


Рисунок 4 – Альтернативное меню выбора ядра Linux в процессе установки Astra Linux SE 1.8.x

Если вы устанавливаете Astra Linux SE версии 1.7.x система предложит выбрать альтернативные версии ядра Linux:

- linux-5.10-generic;
- linux-5.10-hardened;
- linux-5.15-generic;
- linux-5.15-hardened;
- linux-5.15-lowlatency;
- linux-5.4-generic;
- linux-5.4-hardened.

В этом случае допускается выбор любой версии ядра Linux, удовлетворяющей вашим задачам. Такой выбор не оказывает влияния на процедуру установки программного обеспечения.

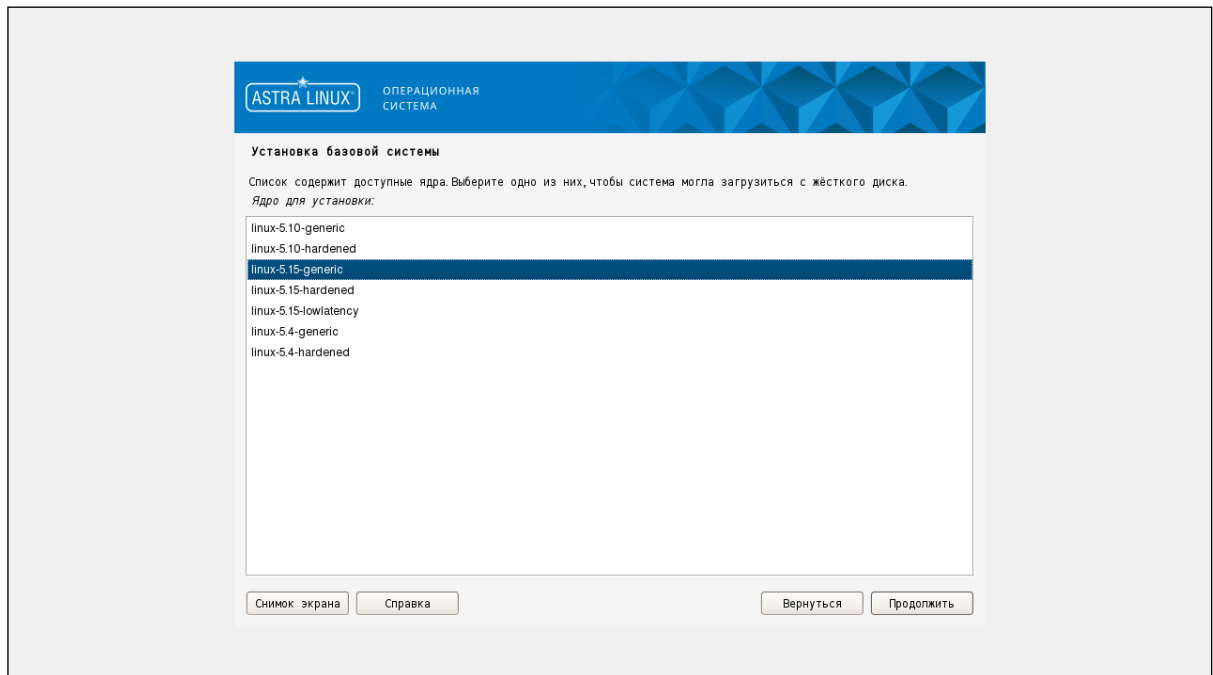


Рисунок 5 – Меню выбора ядра Linux в процессе установки Astra Linux SE 1.7.x

После завершения процесса установки операционной системы дождитесь её перезагрузки, затем выполните вход в систему, используя учётные данные администратора, указанные на этапе установки.

4.1. Подготовка операционной системы специального назначения Astra Linux Special Edition к установке ПО TROK

Откройте файл конфигурации репозитория операционной системы (например, с помощью команды `sudo vim /etc/apt/sources.list`) и проанализируйте его содержимое согласно дальнейшей инструкции. Данный файл содержит перечень репозитория программного обеспечения, используемых системой для установки и обновления пакетов. Состав и структура этого списка зависят от используемой версии операционной системы.

Для установки программного обеспечения TROK рекомендуется использовать операционные системы, настроенные на работу с репозиториями ветки «frozen» либо с репозиториями, размещёнными на установочном дистрибутиве Astra Linux Special Edition. Выбор и использование репозитория должен осуществляться строго в соответствии

с версией установленной операционной системы, чтобы обеспечить корректную установку и функционирование программного обеспечения.

Использование репозитория ветки «stable» нежелательно. При работе с данным типом репозитория существует риск автоматической установки неподходящей версии ядра, что может привести к конфликтам зависимостей и некорректной работе системы.

В пунктах 4.1.1 и 4.1.2 представлены два возможных варианта настройки, которые следует выбирать в зависимости от уровня защищенности операционной системы, которую вы используете.

4.1.1. Настройка репозитория для систем с доступом к сети Интернет (настройка для использования ветки репозитория «frozen»)

Процесс изменения репозитория сводится к редактированию основного файла `/etc/apt/sources.list`.

Перед началом изменений узнайте номер нужного обновления (например, 1.8.4) из официального бюллетеня безопасности. Также можно проверить версию с помощью ввода команды:

```
cat /etc/astora_version
```

В файле измените строки, начинающиеся с `deb https://dl.astralinux.ru/astora/stable/...`, заменив `stable` на `frozen` и добавив номер обновления.

Для версии 1.8.4 список репозитория будет следующим образом:

Было (ветка `stable`):

```
# Основной репозиторий, включающий актуальное оперативное или срочное обновление
deb https://dl.astralinux.ru/astora/stable/1.8_x86-64/main-repository/
1.8_x86-64 main contrib non-free non-free-firmware

# Расширенный репозиторий, соответствующий актуальному оперативному обновлению
deb https://dl.astralinux.ru/astora/stable/1.8_x86-64/extended-repository/
1.8_x86-64 main contrib non-free non-free-firmware
```

Стало (ветка `frozen`, обновление 1.8.4):

```
deb      https://dl.astralinux.ru/astra/frozen/1.8_x86-64/1.8.4/main-repository/
1.8_x86-64 main contrib non-free non-free-firmware

deb https://dl.astralinux.ru/astra/frozen/1.8_x86-64/1.8.4/extended-repository/
1.8_x86-64 main contrib non-free non-free-firmware
```

Если строка закомментирована (в начале строки ссылки на репозиторий присутствует знак «#»), то ее необходимо раскомментировать (удалите знак «#» для активации доступа к репозиториям и обеспечения обновлений).

Важно. При переходе на новое оперативное или срочное обновление номер обновления в ссылке необходимо будет также изменить вручную.

Поддерево «frozen»: репозитории Astra Linux x.7 выглядят немного иначе:

```
# Базовый репозиторий

deb      https://dl.astralinux.ru/astra/frozen/1.7_x86-64/1.7.3/repository-base/
1.7_x86-64 main contrib non-free

# Расширенный репозиторий

deb https://dl.astralinux.ru/astra/frozen/1.7_x86-64/1.7.3/repository-extended/
1.7_x86-64 main contrib non-free

# Базовый репозиторий срочные обновления

deb https://dl.astralinux.ru/astra/frozen/1.7_x86-64/1.7.3/uu/2/repository-base/
1.7_x86-64 main contrib non-free

# Расширенный репозиторий срочные обновления

deb      https://dl.astralinux.ru/astra/frozen/1.7_x86-64/1.7.3/uu/2/repository-
extended/ 1.7_x86-64 main contrib non-free
```

Выполните обновление кэша пакетов, используемого системой для отслеживания актуальных версий доступного программного обеспечения, с помощью команды:

```
sudo apt update
```

Команда инициирует повторное считывание и актуализацию индексов пакетов из подключённых репозиториях.

```

success-express@Astra-1:~$ sudo apt update
Сущ:1 https://dl.astralinux.ru/astra/frozen/1.8_x86-64/1.8.4/main-repository 1.8_x86-64 InRelease
Сущ:2 https://dl.astralinux.ru/astra/frozen/1.8_x86-64/1.8.4/extended-repository 1.8_x86-64 InRelease
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Может быть обновлено 1832 пакета. Запустите «apt list --upgradable» для их показа.
success-express@Astra-1:~$

```

Рисунок 6 – Пример результата выполнения команды `sudo apt update`

После этого перейдите к процедуре монтирования ISO-образа программы TROK (п. 4.2).

4.1.2. Перенастройка репозитория для автономных систем (локальные репозитории iso образа ОС)

Для получения данных из локальных репозиториях также требуется произвести редактирование файла `/etc/apt/sources.list`, однако в случае с локальными репозиториями необходимо внести другие изменения.

Закомментируйте строки (добавьте знак «#» перед началом строки), начинающиеся с `deb https://dl.astralinux.ru/astra/stable/...` или `deb https://dl.astralinux.ru/astra/frozen/...`

Для установки в закрытом контуре укажите источник `cdrom`:

Для версии 1.8.1 Astra Linux Special Edition строка со ссылкой на локальные репозитории выглядит следующим образом:

```
deb cdrom:[OS Astra Linux 1.8.1.6 DVD]/ 1.8_x86-64 contrib main non-free non-free-firmware
```

Для версии Astra Linux Special Edition 1.7.5 следующий пример:


```
sudo apt update
```

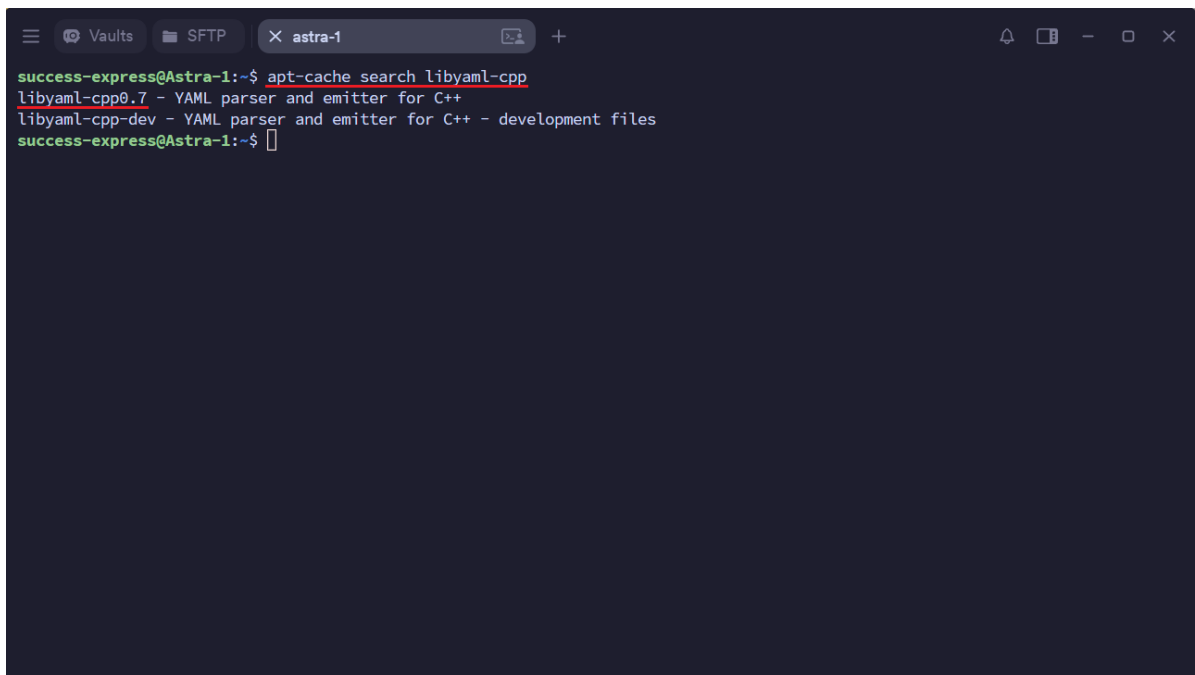
4.2. Монтирование образа TROK

Перед установкой ПО TROK необходимо убедиться в наличии необходимой версии библиотеки `libyaml-cpp` для чтения и создания YAML-файлов. Требуемая версия этой библиотеки зависит от используемой архитектуры и версии операционной системы.

Версию библиотеки для вашей конкретной платформы можно узнать с помощью команды:

```
apt-cache search libyaml-cpp
```

Команда покажет все доступные в репозиториях пакеты, содержащие `libyaml-cpp`.

A screenshot of a terminal window titled 'astra-1'. The terminal shows the command 'apt-cache search libyaml-cpp' being executed. The output lists two packages: 'libyaml-cpp0.7 - YAML parser and emitter for C++' and 'libyaml-cpp-dev - YAML parser and emitter for C++ - development files'. The prompt 'success-express@Astra-1:~\$' is visible at the bottom of the terminal output.

```
success-express@Astra-1:~$ apt-cache search libyaml-cpp
libyaml-cpp0.7 - YAML parser and emitter for C++
libyaml-cpp-dev - YAML parser and emitter for C++ - development files
success-express@Astra-1:~$
```

Рисунок 8 – Пример вывода команды `apt-cache search libyaml-cpp`

В качестве примера ниже приведена попытка установки пакета устаревшей версии. Система сообщает, что указанный пакет не найден в доступных репозиториях.

```

success-express@Astra-1:~$ sudo apt install libyaml-cpp0.6
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
E: Невозможно найти пакет libyaml-cpp0.6
E: Не удалось найти ни один пакет с помощью шаблона «libyaml-cpp0.6»
success-express@Astra-1:~$ sudo apt install libyaml-cpp0.7
Чтение списков пакетов... Готово
Построение дерева зависимостей... Готово
Чтение информации о состоянии... Готово
Уже установлен пакет libyaml-cpp0.7 самой новой версии (0.7.0+dfsg-8+b1).
libyaml-cpp0.7 помечен как установленный вручную.
Обновлено 0 пакетов, установлено 0 новых пакетов, для удаления отмечено 0 пакетов, и 1832 пакетов не обновлено.
success-express@Astra-1:~$

```

Рисунок 9 – Пример реакции системы на попытку установки неподходящего пакета libyaml-cpp

Для специальной версии TROK, предназначенной для архитектуры ARM, чаще требуется предварительная установка пакета libyaml-cpp0.6. Установку можно выполнить командой:

```
sudo apt install libyaml-cpp0.6
```

В ряде версий Astra Linux Special Edition (например, CH ALSE 1.8.3uu1 для архитектуры amd64 (x86_64)), установка или работа программного обеспечения может требовать наличие более новой версии библиотеки – libyaml-cpp0.7.

Для установки воспользуйтесь командой:

```
sudo apt install libyaml-cpp0.7
```

Для установки программного обеспечения TROK используется ISO-образ. Перед началом установки необходимо выполнить монтирование образа, чтобы получить доступ к его содержимому.

Скопируйте образ диска в удобное для вас место в операционной системе.

Создайте каталог /mnt/trok_installer используя команду:

```
sudo mkdir /mnt/trok_installer
```

```

success-express@Astra-1:~$ sudo mkdir /mnt/trok_installer
[sudo] пароль для success-express:
success-express@Astra-1:~$ ls /mnt/
my-share trok_installer
success-express@Astra-1:~$

```

Рисунок 10 – Результат выполнения команды `sudo mkdir /mnt/trok_installer`

С помощью команды:

```
sudo mount -o loop <место_расположения_ISO-образа> /mnt/trok_installer
```

производится монтирование ISO-файла в созданный каталог с использованием опции `loop`, позволяющей обращаться с образом как с виртуальным блочным устройством.

(например, `sudo mount -o loop /home/admin/TPOK-2.0.iso /mnt/trok_installer`)

```

success-express@Astra-1:~$ sudo mount -o loop /mnt/my-share/TROK-2.0.iso /mnt/trok_installer/
[sudo] пароль для success-express:
mount: /mnt/trok_installer: ВНИМАНИЕ: источник защищен от записи и примонтирован только для чтения.
success-express@Astra-1:~$

```

Рисунок 11 – Уведомление системы об успешном завершении процедуры монтирования образа

Для проверки успешного монтирования и просмотра содержимого образа используйте команду:

```
ls /mnt/trok_installer
```

Важно! Перед началом процедуры установки перейдите в каталог, в который был смонтирован образ:

```
cd /mnt/trok_installer/installer
```

Далее все команды установки предполагают, что текущая рабочая директория – /mnt/trok_installer/installer.

Если по каким-то причинам вы выполняете команды из другого каталога, замените префикс «./» в командах вида `sudo ./trok_installer` на соответствующий абсолютный или относительный путь к исполняемому файлу, например: `sudo /mnt/trok_installer/installer/trok_installer`.

5. КОНФИГУРАЦИЯ

Программа установки использует файл `config.yaml`, входящий в состав ISO. Файл содержит пути к архиву пакетов, APT source, keyring, лицензионному соглашению и перечень устанавливаемых метапакетов.

Используйте файл, поставленный внутри выбранного ISO. Не переносите `config.yaml` из другого выпуска и не меняйте имена архива или пакетов без прямого указания в Release Notes.

Структура файла имеет следующий вид:

```
options:
  archive: ../repository/TROK-<версия>.tar.gz
  repository_entry: deb [arch=amd64,arm64 signed-by=/usr/share/keyrings/trok-
archive-keyring.gpg] file:///opt/TROK/SDS/apt-repo stable main
  repository_config: /etc/apt/sources.list.d/trok_repo.list
  keyring_file: ../repository/trok-archive-keyring.gpg
  keyring_dest: /usr/share/keyrings/trok-archive-keyring.gpg
  eula_file: ../EULA_SDS_Trok.txt
  eula_dest: /usr/share/doc/trok/EULA_SDS_Trok.txt
  log: /var/log/trok-installer

packages:
  worker:
    - name: trok-worker-meta
      version: ''
  controller:
    - name: trok-controller-meta
      version: ''
  harbor:
    - name: trok-harbor-meta
      version: ''
  webui:
    - name: trok-webui
      version: ''
  s3:
    - name: trok-s3-gw
      version: ''
```

Параметр `version` с пустым значением означает установку candidate-версии из подключённого репозитория, выбранного ISO. Candidate необходимо проверить командой `apt-cache policy <имя_пакета>`.

Путь для журналов можно изменить через параметр `options.log` либо параметр командной строки `--log`.

Важно. В актуальном инсталляторе установка DRBD выполняется внутри программ `trok-installer` и `trok-installer-gui`. Параметр `autoinstall_path` и внешний файл `/usr/sbin/trok-drbd-autoinstall` не используются.

6. УСТАНОВКА С ИСПОЛЬЗОВАНИЕМ КОМАНДНОЙ СТРОКИ

Для запуска консольного интерфейса перейдите в каталог `installer` смонтированного ISO и выполните:

```
cd /mnt/trok_installer/installer
sudo ./trok-installer --console
```

Без параметра `--console` программа при наличии графического окружения пытается запустить GUI. Поэтому для гарантированного консольного режима всегда указывайте `--console`.

Программа поддерживает следующие основные параметры:

Таблица 3 – Параметры установки

Аргумент	Описание
<code>--worker</code>	Установка узла типа Worker
<code>--controller</code>	Установка узла типа Controller
<code>--combined</code>	Установка совмещённого узла Worker + Controller
<code>--harbor</code>	Установка iSCSI, NFS и NVMe-oF компонентов; только для Worker или Combined
<code>--webui</code>	Установка WebUI; только для Controller или Combined
<code>--s3</code>	Установка S3-шлюза; только для Worker или Combined; необходимые gateway-компоненты устанавливаются автоматически
<code>--log <директория></code>	Каталог журналов установки
<code>--verbose</code>	Подробный вывод журнала на экран
<code>--console</code>	Принудительный запуск консольного интерфейса

Примеры:

```
# Worker с интерфейсами iSCSI/NFS/NVMe-oF
sudo ./trok-installer --console --worker --harbor
```

```
# Worker с S3-шлюзом

sudo ./trok-installer --console --worker --s3


# Совмещённый узел с WebUI

sudo ./trok-installer --console --combined --webui


# Подробный журнал в отдельном каталоге

sudo ./trok-installer --console --combined --webui \
    --verbose --log /var/log/trok-installer
```

Неинтерактивный запуск с параметрами роли означает принятие лицензионного соглашения, входящего в состав поставки. До запуска ознакомьтесь с текстом соглашения.

Инсталлятор самостоятельно выбирает пакет DRBD, соответствующий работающему ядру. Если подходящий пакет отсутствует, установка завершается с ошибкой либо с явно указанным требованием перезагрузки. Не запускайте после инсталлятора trok-drbd-autoinstall.

После успешного завершения выполните проверки из пункта 11. Если инсталлятор сообщил о необходимости перезагрузки, сначала перезагрузите узел, затем проверьте версию ядра и загруженный модуль DRBD.

7. УСТАНОВКА В ИНТЕРАКТИВНОМ РЕЖИМЕ (GUI)

В случае использования графического интерфейса в операционной системе и запуска установочного файла trok-installer-gui посредством данного интерфейса автоматически инициируется графический режим конфигурирования.

Важно. Для запуска необходимо находиться в директории, в которой располагается установочный файл.

Для явного запуска графического интерфейса выполните из каталога installer:

```
cd /mnt/trok_installer/installer
sudo ./trok-installer-gui
```

Также допускается запуск `sudo ./trok-installer` без параметра `--console`: при наличии графического окружения программа запустит GUI, а при невозможности инициализации GUI перейдёт в консольный режим.

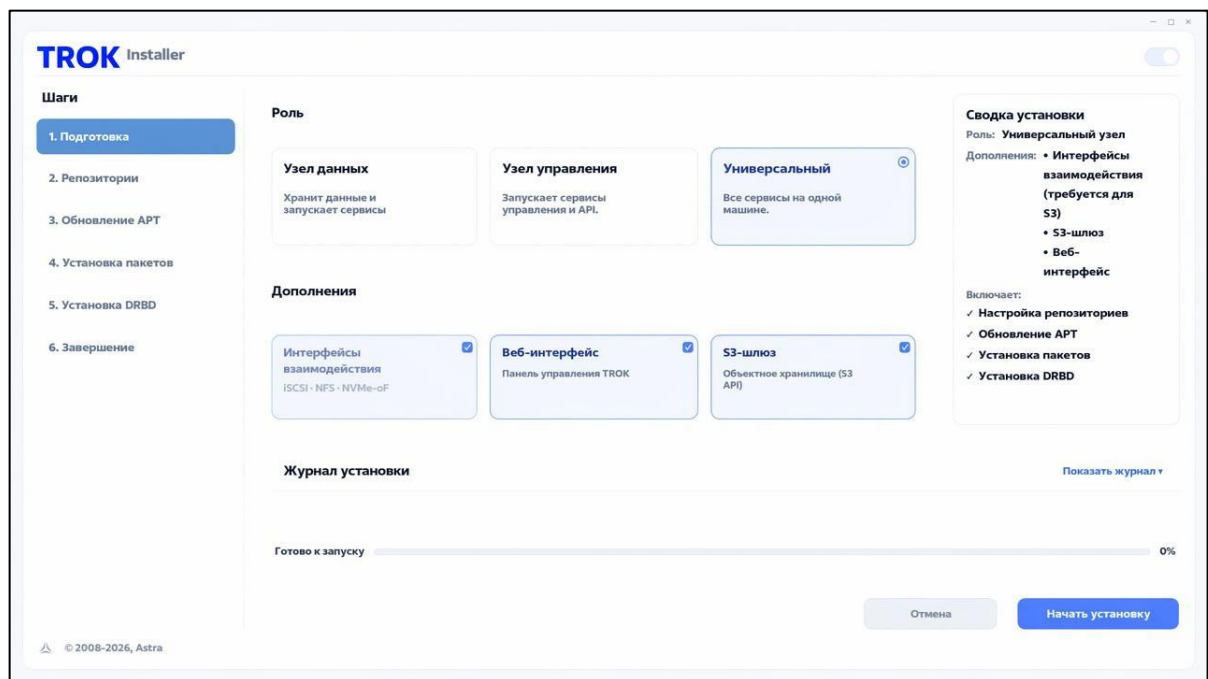


Рисунок 12 – Установка в интерактивном режиме

После запуска программы установки пользователю предоставляется выбор между следующими архитектурными конфигурациями:

- Определение топологии узла:
 - Узел данных: Установка узла категории worker (исполнительный компонент).

- Универсальный (combined): Комбинированная установка (worker + controller).
- Узел управления (controller): Развертывание управляющего узла (control plane).
- Компоненты, автоматически устанавливающиеся при установке trok-controller:
 - trok-auth: обеспечивает централизованную аутентификацию и авторизацию для всех компонентов TROK через gRPC API, управляя OAuth2 клиентами (приложениями) и выдачей JSON Web Tokens (JWT) в соответствии со спецификацией OpenID.
 - trok-cp-endpoint: обеспечивает REST API TROK для управления и мониторинга DRBD - ресурсов. Сервис выступает единой точкой входа для всех операций управления кластером, обеспечивая взаимодействие между пользователями/клиентами и бэкендом контроллера.
 - Технические особенности реализации:
 - harbor: готовый программный модуль, объединяющий службу автоматизации DRBD (drbd-reactor), набор сценариев управления ресурсами (resource-agents) и поддержку сервисов доступа (iSCSI, NFS, SMB), который поставляется в виде отдельного мета-пакета для установки на узлах хранения, а также входит в состав контроллера.
 - WEBUI: Веб-интерфейс управления (включая сервис nginx).

Перед началом установки ознакомьтесь с лицензионным соглашением и нажмите «Я принимаю условия соглашения», если готовы продолжать установку программы.

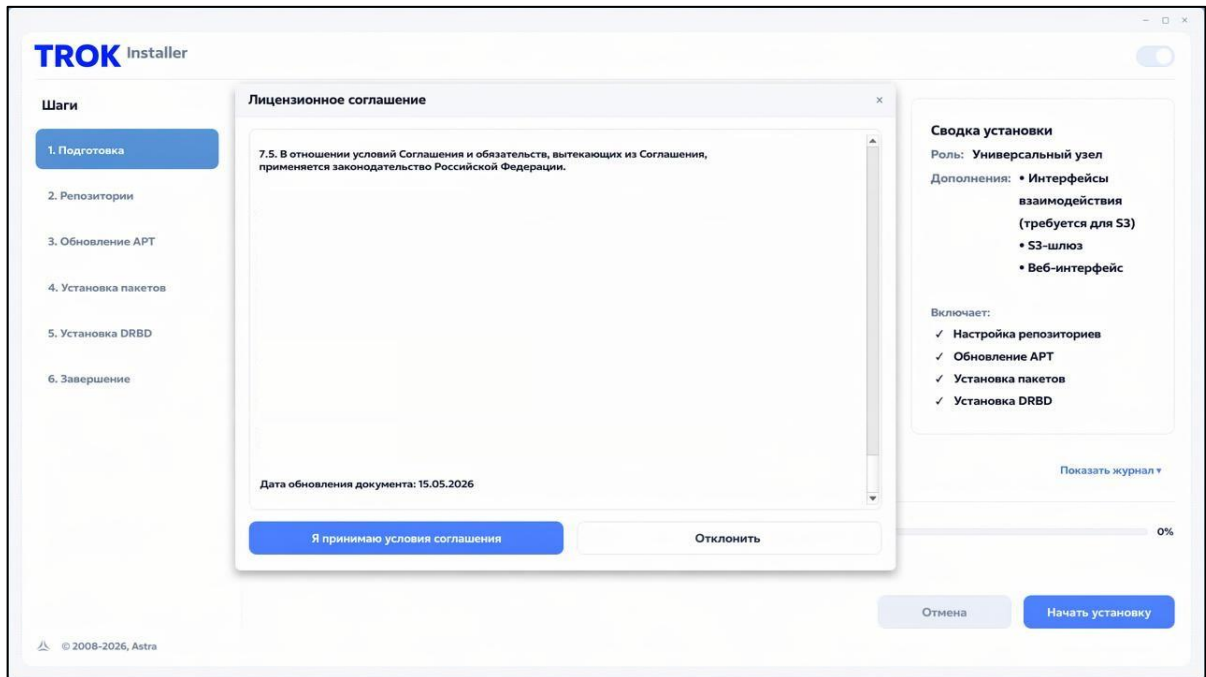


Рисунок 13 – Лицензионное соглашение

Все дальнейшие шаги установки производятся автоматически. По завершении установки отображается сообщение об успешном завершении процесса.

После выполнения установки рекомендуется перезагрузить систему.

В случае возникновения проблем с установкой DRBD смотрите пункт 8.

В GUI действуют следующие ограничения:

- Harbor и S3 устанавливаются только на Worker или Combined;
- WebUI устанавливается только на Controller или Combined;
- S3 автоматически включает необходимые gateway-компоненты.

Перед началом установки ознакомьтесь с лицензионным соглашением и подтвердите его принятие. DRBD устанавливается самим инсталлятором; отдельный запуск `trok-drbd-autoinstall` после завершения не требуется.

8. УСТАНОВКА С ИСПОЛЬЗОВАНИЕМ ПАКЕТОВ

Ручная установка используется только тогда, когда применение standalone-инсталлятора и Ansible невозможно либо прямо запрещено условиями эксплуатации. Этот способ является альтернативой пунктам 6, 7, 9 и 10.

Перед началом уточните роль узла и состав пакетов в Release Notes выбранного выпуска.

8.1. Подготовка локального репозитория

Задайте параметры выпуска:

```
export TROK_ARCHIVE='/mnt/trok_installer/repository/TROK-<версия>.tar.gz'
export TROK_REPO='/opt/TROK/SDS/apt-repo'
```

Убедитесь, что архив существует, а каталог назначения ещё не содержит репозиторий другого выпуска:

```
test -f "$TROK_ARCHIVE"
test ! -e "$TROK_REPO/dists/stable/Release"
```

Если в каталоге уже находится репозиторий, остановитесь. Не объединяйте содержимое разных выпусков.

Создайте каталог и распакуйте архив:

```
sudo install -d -o root -g root -m 0755 "$TROK_REPO"
sudo tar -xzf "$TROK_ARCHIVE" \
  --strip-components=3 \
  -C "$TROK_REPO"
test -f "$TROK_REPO/dists/stable/Release"
```

Установите keyring:

```
sudo install -d -o root -g root -m 0755 /usr/share/keyrings
sudo install -o root -g root -m 0644 \
  /mnt/trok_installer/repository/trok-archive-keyring.gpg \
  /usr/share/keyrings/trok-archive-keyring.gpg
```

Создайте `/etc/apt/sources.list.d/trok_repo.list` со следующим содержимым, подставив фактическую архитектуру из `dpkg --print-architecture`:

```
deb [arch=<архитектура> signed-by=/usr/share/keyrings/trok-archive-keyring.gpg]
file:///opt/TROK/SDS/apt-repo stable main
```

Создайте /etc/apt/preferences.d/99-trok.pref:

```
Package: *
Pin: release o=TROK
Pin-Priority: 1001
```

Обновите индекс и проверьте происхождение candidate:

```
sudo apt-get update

apt-cache policy trok-worker-meta trok-controller-meta trok-combined-meta
```

Candidate устанавливаемого пакета должен поступать из file:///opt/TROK/SDS/apt-repo. Если candidate отсутствует или выбран из другого источника, установку не начинайте.

8.2. Установка Controller, Worker или Combined

Controller:

```
sudo apt-get install --no-install-recommends \
  trok-controller-meta trok-webui
```

Worker без gateway-компонентов:

```
sudo apt-get install --no-install-recommends trok-worker-meta
```

Worker с Harbor:

```
sudo apt-get install --no-install-recommends \
  trok-worker-meta trok-harbor-meta
```

Worker с S3:

```
sudo apt-get install --no-install-recommends \
  trok-worker-meta trok-harbor-meta trok-s3-gw
```

Combined:

```
sudo apt-get install --no-install-recommends \
  trok-combined-meta trok-webui
```

Не устанавливайте nginx или Apache отдельной командой без требования Release Notes: нужный веб-сервер должен быть зависимостью target-пакета trok-webui.

8.3. Установка DRBD при ручном способе

На Worker и Combined требуется пакет DRBD, точно соответствующий работающему ядру:

```
KERNEL_RELEASE=$(uname -r)
DRBD_PACKAGE="drbd-$KERNEL_RELEASE"

if ! apt-cache show "$DRBD_PACKAGE" >/dev/null 2>&1; then
    echo "Отсутствует точный пакет $DRBD_PACKAGE. Установка остановлена." >&2
    exit 1
fi

sudo apt-get install --no-install-recommends \
    "$DRBD_PACKAGE" drbd-utils
```

Если точный пакет отсутствует, не устанавливайте произвольный drbd-dkms и не выбирайте пакет для другой версии или flavour ядра. Используйте standalone-инсталлятор либо обратитесь к поставщику выпуска.

После установки проверьте пункт 11. Команда trok-drbd-autoinstall в актуальном выпуске не используется.

9. ДЕПЛОЙ КЛАСТЕРА БЕЗ НА

Глава описывает развертывание обычного кластера TROK без HA-контроллера для тех, кто знаком с базовой работой Ansible и хочет быстро получить рабочую схему: controller, worker, пул хранения и экспорт по iSCSI, NFS, а также HA Versity S3-шлюз.

9.1. Требования к управляющему хосту (узел управления Ansible)

Для работы Ansible подготовьте отдельного технического пользователя с правами sudo без интерактивного запроса пароля и ключом SSH.

Создайте ключ:

```
ssh-keygen -t ed25519 -f ~/.ssh/trok_ansible -C 'trok-ansible'
```

Скопируйте публичный ключ на каждый целевой узел:

```
ssh-copy-id -i ~/.ssh/trok_ansible.pub <пользователь>@<узел>
```

До запуска Ansible получите SSH host key каждого узла и сравните его fingerprint со значением, полученным по доверенному каналу:

```
ssh-keyscan -t ed25519 <узел> > /tmp/trok-host-key
ssh-keygen -lf /tmp/trok-host-key
```

После подтверждения fingerprint добавьте ключ:

```
cat /tmp/trok-host-key >> ~/.ssh/known_hosts
rm -f /tmp/trok-host-key
```

Не отключайте StrictHostKeyChecking и не используйте UserKnownHostsFile=/dev/null в эксплуатационной конфигурации.

Минимальные требования коллекции:

- Ansible 2.15.0 или выше;
- Python 3 и python3-netaddr на управляющем хосте;
- community.general >=8.0.0,<9.0.0;
- ansible.posix >=1.3.0.

При установке коллекции из архива зависимости разрешаются `ansible-galaxy`. Не используйте `sudo ansible-galaxy`: коллекция должна устанавливаться от имени того пользователя, под которым запускается Ansible.

9.2. Установка коллекции TROK

Найдите архив коллекции внутри выбранного ISO. Команда должна обнаружить ровно один файл:

```
mapfile -t TROK_COLLECTIONS < <(
    find /mnt/trok-installer/ansible-collection \
        -maxdepth 1 -type f -name 'astra-trok-*.tar.gz' -print
)
test "${#TROK_COLLECTIONS[@]}" -eq 1
```

Установите коллекцию от имени Ansible-пользователя вместе с её зависимостями:

```
ansible-galaxy collection install "${TROK_COLLECTIONS[0]}" --force
ansible-galaxy collection list astra.trok
ansible-galaxy collection list community.general
ansible-galaxy collection list ansible.posix
```

Документация установленной коллекции

Команда `ansible-galaxy collection list astra.trok` выводит в строке, начинающейся с #, фактический каталог `ansible_collections`. Определить каталог коллекции можно так:

```
COLLECTIONS_ROOT=$(
    ansible-galaxy collection list astra.trok |
        awk '/^# \/.ansible_collections/ { sub(/^# /, ""); print; exit }'
)
test -n "$COLLECTIONS_ROOT"
TROK_COLLECTION_DIR="$COLLECTIONS_ROOT/astra/trok"
```

```
test -f "$TROK_COLLECTION_DIR/README_ru.md"

printf '%s\n' "$TROK_COLLECTION_DIR"
```

Начальная русскоязычная страница —
 \$TROK_COLLECTION_DIR/README_ru.md. В установленной коллекции также доступны:

- docs/ru/runbook-quickstart.md — минимальный первый запуск;
- docs/ru/runbook-standard.md — развёртывание non-HA кластера;
- docs/ru/runbook-ha.md — HA-развёртывание;
- docs/ru/runbook-upgrade.md — поддерживаемая процедура обновления;
- docs/ru/variables.md — справочник публичных и ролевых переменных;
- docs/ru/modules.md — описание модулей коллекции;
- CHANGELOG.md — изменения между версиями.

Встроенную справку по модулям можно получить через ansible-doc:

```
ansible-doc -t module -l astra.trok

ansible-doc astra.trok.trok_wait_controller
```

9.3. Подготовка структуры проекта

Для подготовки среды развёртывания (деплой) системы с использованием Ansible рекомендуется создать организованную структуру каталогов и файлов. Структура рабочего каталога trok-deploy/ должна выглядеть следующим образом:

```
trok-deploy/
├─ inventory.ini
├─ group_vars/
│   └─ all.yml
└─ site.yml
```

inventory.ini – файл инвентаря, который определяет список целевых хостов (серверов), их группы и параметры подключения (например, IP-адреса, SSH-ключи).

group_vars/all.yaml – файл переменных в формате YAML, расположенный в подкаталоге group_vars/. Он содержит глобальные переменные, доступные всем хостам, такие как версии компонентов или ключи шифрования.

site.yml – основной плейбук Ansible в формате YAML, расположенный в корневом каталоге. Файл описывает последовательность задач развертывания, включая установку ПО, конфигурацию служб и проверку состояния. Он служит точкой входа для запуска всего процесса автоматизации.

9.4. Конфигурация inventory.ini

Пример:

```
[controllers]
ctrl1.local.admin ansible_host=192.0.2.11 ansible_user=ansible ansible_port=22
ansible_connection=ssh ansible_ssh_private_key_file=~/.ssh/trok_ansible

[workers]
sat1.local.admin ansible_host=192.0.2.12 ansible_user=ansible ansible_port=22
ansible_connection=ssh ansible_ssh_private_key_file=~/.ssh/trok_ansible
sat2.local.admin ansible_host=192.0.2.13 ansible_user=ansible ansible_port=22
ansible_connection=ssh ansible_ssh_private_key_file=~/.ssh/trok_ansible

[iscsi_exporters]
sat1.local.admin
sat2.local.admin

[nfs_exporters]
sat1.local.admin
sat2.local.admin

[s3gw_gateways]
sat1.local.admin
sat2.local.admin

[all:vars]
ansible_become=true
```

Повторное объявление одного хоста в дополнительных группах не требует повторять параметры SSH: Ansible объединяет принадлежность хоста к группам по одному inventory_hostname.

9.5. Настройка переменных (group_vars/all.yml)

9.5.1. Базовая часть group_vars/all.yml

Задайте ровно один источник ISO. Одновременное непустое значение `trok_iso_local_src` и `trok_iso_url`, как и отсутствие обоих значений, является ошибкой контракта и останавливает роль до подготовки репозитория.

Локальный файл на управляющем хосте:

```
trok_iso_local_src: "/home/ansible/iso/<имя_ISO>"
trok_iso_url: ""
```

Либо URL, доступный каждому целевому узлу, а не только управляющему хосту:

```
trok_iso_local_src: ""
trok_iso_url: "https://repo.example.local/<имя_ISO>"
```

Остальные базовые переменные:

```
controllers:
  - 192.0.2.11

workers:
  - 192.0.2.12
  - 192.0.2.13

drbd_replication_network: "192.0.2.0/24"
enable_gateway: true
ufw_enable: false
dist_upgrade: false
cluster_member: true
sp_cluster_member: true
drbd_auto_reboot: true
```

Переменные `controllers` и `workers` должны быть непустыми YAML-списками, а не строками или перечнями через запятую. Каждый `worker` должен однозначно сопоставляться с одним адресом из этих списков. Некорректный тип, пустой список или неоднозначное сопоставление останавливают роль до изменения узла.

Пользователь задаёт только один из двух источников ISO;

Коллекция использует собственный репозиторий `/var/lib/trok/ansible/apt-repo`, `keyring` `/etc/apt/keyrings/trok-archive-keyring.gpg` и `source` `/etc/apt/sources.list.d/trok-sds.list`. Если активен `source` инсталлятора `file:///opt/TROK/SDS/apt-repo`, роль завершится до изменения файловой системы. Отключите конфликтующий `source` и повторите запуск; не объединяйте два репозитория.

9.5.2. Опционально: глобальная настройка DRBD на worker-узлах

Включайте настройку только в том случае, если узел с ролью `worker` должен управлять конфигурационным файлом DRBD `/etc/drbd.d/global_common.conf`. Если узлу не требуется управлять DRBD-настройками глобального уровня, переменную следует оставить выключенной или не указывать.

```
worker_drbd_tuning_enabled: true
worker_drbd_tuning_profile: "25g-ssd"
worker_drbd_tuning_resources_per_node: 20
worker_drbd_tuning_reboot: false
```

Настройка профиля производительности DRBD автоматически рассчитывает оптимальные параметры DRBD на основе двух факторов:

- Выбранного профиля (скорость сети и тип дисков).
- Количества ресурсов, которые будут работать на узле.

Поддерживаемые значения `worker_drbd_tuning_profile`:

Таблица 4 – Доступные профили настройки DRBD

Тип конфигурации	Доступные профили
Базовый (по умолчанию)	default (соответствует 10g-hdd)
Сеть 1 Гбит/с или 10 Гбит/с	1g-hdd, 10g-hdd – для HDD 1g-ssd, 10g-ssd – для SSD 1g-nvme, 10g-nvme – для NVMe 1g-nvme-aggressive, 10g-nvme-aggressive – агрессивный режим для NVMe

Сеть 25 Гбит/с или 40 Гбит/с	25g-hdd, 40g-hdd – для HDD 25g-ssd, 40g-ssd – для SSD 25g-nvme, 40g-nvme – для NVMe 25g-nvme-aggressive, 40g-nvme-aggressive – агрессивный режим для NVMe
Сеть 100 Гбит/с	100g-hdd – для HDD 100g-ssd – для SSD 100g-nvme – для NVMe 100g-nvme-aggressive – агрессивный режим для NVMe
Сеть 200 Гбит/с	200g-hdd – для HDD 200g-ssd – для SSD 200g-nvme – для NVMe 200g-nvme-aggressive – агрессивный режим для NVMe

Примечание: Профили с пометкой -aggressive используют более высокие настройки производительности, что даёт прирост скорости, но требует более стабильной сетевой инфраструктуры. Агрессивные профили нельзя использовать, если на узле запланировано 50 и более ресурсов – такие настройки создадут избыточную нагрузку и могут дестабилизировать работу системы.

Настройка количества DRBD-ресурсов на узле `worker_drbd_tuning_resources_per_node` задаёт, сколько DRBD-ресурсов будет работать на одном узле. От этого зависят параметры производительности – чем больше ресурсов, тем ниже пропускная способность у каждого из них.

Доступные значения:

Таблица 5 – Доступные значения количества ресурсов на узел

Значение	Рекомендации по применению
----------	----------------------------

1, 10, 20	Низкая плотность – на узле мало ресурсов, каждый из них может работать с максимальной производительностью.
50, 100	Средняя плотность – ресурсов уже достаточно много, поэтому производительность каждого умеренно ограничивается в пользу общей стабильности.
200, 500	Высокая плотность – на узле очень много ресурсов, приоритет отдаётся общей устойчивости системы, а не скорости отдельного ресурса.
unlimited	Особый режим – подходит для большого количества небольших ресурсов, где каждый потребляет минимум ресурсов системы.

Общий принцип: чем больше ресурсов на узле, тем более «консервативными» становятся настройки DRBD для каждого из них, чтобы избежать перегрузки системы.

Таблица 6 – Примеры комбинаций

Сценарий	Профиль	Ресурсов на узел
Консервативный базовый вариант	10g-hdd	20
Обычный быстрый узел	25g-ssd	20
Быстрый, но уже плотный узел	100g-nvme	100
Сценарий с множеством ресурсов	200g-nvme	500

9.5.3. Пул хранения

Для хранения данных используется LVM-пул. В типичном рабочем кластере (без требований высокой доступности) это хороший вариант по умолчанию, и на практике он часто работает быстрее других решений.

Настройки пулов хранения задаются в глобальном файле переменных: `trok-deploy/group_vars/all.yml` в корне проекта Ansible.

Пример настройки обычного LVM-пула:

```
storage_pool_lvm_pools:
  - name: lvm_pool
    vg_name: my_vg
    disks:
      - /dev/vdb
```

Обязательные параметры:

`name` – имя пула

`vg_name` – имя группы томов (Volume Group)

`disks` – список дисков, которые будут использоваться в пуле (каждый путь должен быть вида `/dev/...`)

Таблица 7 – Поддерживаемые типы пулов

Тип хранилища	Переменная	Роль
Обычный LVM	<code>storage_pool_lvm_pools</code>	<code>astra.trok.storage_pool_lvm</code>
LVM Thin	<code>storage_pool_lvm_thin_pools</code>	<code>astra.trok.storage_pool_lvm_thin</code>

Пример обычного LVM pool:

```
storage_pool_lvm_pools:
  - name: lvm_pool
    vg_name: my_vg
    tags:
      - FAST
      - production
    disks:
      - /dev/vdb
```

Пример LVM Thin pool:

```
storage_pool_lvm_thin_pools:
  - pool_name: thinpool
    vg_name: drbdpool
    tags:
      - SSD
    disks:
      - /dev/vdb
```

```
size: "8.5G"
```

В одном play запускайте только роль выбранного типа пула. Для обычного LVM обязательны name, vg_name и непустой список disks. Для LVM Thin обязательны pool_name, vg_name, disks и size. Поле tags необязательно и передаётся в TROK без изменения регистра, порядка и формата. Устаревшие поля Storage Pool API options, props и StorDriver aliases не используйте.

Важно. Роль astra.trok.storage_pool_zfs не поддерживается текущим Storage Pool API и всегда завершает работу до изменения дисков или ZFS pool. Не используйте ZFS в новом развёртывании и не указывайте ZFS pool в ha_pool_name, iscsi_targets, nfs_exports или настройках S3.

Важно. В плейбуке следует запускать только одну роль подготовки пула – ту, которая соответствует выбранному типу хранилища. Не смешивайте настройки разных типов в одном запуске.

9.5.4. iSCSI-экспорт

Настройки iSCSI-целей задаются в глобальном файле переменных: trok-deploy/group_vars/all.yml в корне проекта Ansible.

Пример настройки iSCSI-цели:

```
iscsi_manage_resources: true

iscsi_targets:
  - resource_name: ISCSI-APP
    storage_pool: lvm_pool
    bind_ip: "192.0.2.249"
    bind_port: 3260
    bind_cidr_netmask: 24
    iqn: "iqn.2026-05.trok.sds:iscsi-app"
    username: "chap-user"
    password: "chap-password"
    allowed_initiators: []
    target_ra:
      tid: 10
    lun_ra_defaults:
      vendor_id: "TROK"
      product_id: "TROK_LUN"
      lio_iblock: 0
    volumes:
      - size: "64MiB"
      - size: "96MiB"
```

Таблица 8 – Обязательные параметры

Параметр	Обязательность
resource_name	Всегда
bind_ip	Всегда
iqn	Всегда
volumes	Всегда
storage_pool	Если iscsi_manage_resources: true
volumes[].size	Если iscsi_manage_resources: true
username и password	Задаются только парой (оба или ни один)

Важные ограничения и правила:

– `allowed_initiators: []` – если оставить список пустым, то любой инициатор сможет подключиться к iSCSI-цели (ограничений по IQN не будет). Чтобы разрешить доступ только определённым инициаторам, укажите их IQN-имена в списке, например:

```
allowed_initiators: ["iqn.1994-05.com.redhat:client1", "iqn.1994-05.com.redhat:client2"]
```

– Группа `iscsi_exporters` должна содержать только те узлы, на которых разрешено размещать iSCSI-ресурс. Если группа слишком широкая, ресурс может быть размещён на лишних узлах.

– Роль необходимо запускать строго на группе узлов `iscsi_exporters`, например:

```
- hosts: iscsi_exporters
  roles:
    - your_iscsi_role
```

Не используйте для запуска этой роли группы `workers` или `all` – это может привести к тому, что роль будет применена на узлах, где iSCSI-экспорт не должен настраиваться.

– Если параметр `autoplace_nodes` явно указывает узлы за пределами группы `iscsi_exporters`, роль завершится ошибкой – неявного расширения размещения не происходит.

Таблица 9 – Практические рекомендации

Сценарий	Рекомендация
Небольшие и обычные ресурсы	Используйте <code>iscsi_manage_resources: true</code> – этого достаточно.
Очень большие ресурсы	Первичная синхронизация может занять много времени. Роль в этом режиме синхронно ждёт завершения синхронизации, что может привести к таймаутам.

Для больших ресурсов лучше использовать двухэтапную схему:

- Сначала создайте ресурс и дождитесь завершения его синхронизации отдельно (вне роли).
- Затем запускайте настройку `iscsi_target` с параметром `iscsi_manage_resources: false`.

Этот подход помогает избежать: таймута синхронизации (`sync_timeout`), внешних таймаутов Ansible, обрывов SSH-соединения, таймаутов в CI-заданиях.

9.5.5. NFS-экспорт

Настройки NFS-экспортов задаются в глобальном файле переменных: `trok-deploy/group_vars/all.yml` в корне проекта Ansible.

Безопасный пример NFS:

```
nfs_manage_resources: true

nfs_exports:
  - resource_name: NFS-APP
    storage_pool: lvm_pool
    size: "128MiB"
    bind_ip: "192.0.2.250"
    bind_cidr_netmask: 24
    mount_point: "/srv/nfs/nfs-app"
```

```

fsid: 200
clientspec: "192.0.2.0/24"
export_options: "rw,sync,no_subtree_check,root_squash"

```

Параметр `root_squash` не позволяет удалённому пользователю `root` автоматически получить права локального `root` на сервере NFS. `no_root_squash` допускается только для отдельно согласованного сценария с документированной моделью угроз и не должен использоваться как значение по умолчанию.

Таблица 10 – Обязательные параметры

Параметр	Обязательность
<code>resource_name</code>	Всегда
<code>bind_ip</code>	Всегда
<code>mount_point</code>	Всегда
<code>fsid</code>	Всегда
<code>clientspec</code>	Всегда
<code>storage_pool</code>	Если <code>nfs_manage_resources: true</code>
<code>size</code>	Если <code>nfs_manage_resources: true</code>

Таблица 11 – Особенности управления NFS-сервисом

Количество экспортов	Способ управления
Один экспорт	Роль управляет NFS-сервером через <code>drbd-reactor</code> (полный цикл: монтирование → IP → запуск NFS)
Несколько экспортов	NFS-сервер работает как обычный <code>systemd</code> -сервис на узлах-экспортёрах. <code>drbd-reactor</code> управляет только монтированием ФС, IP-адресом и экспортом (<code>exportfs</code>), но не перезапускает сам NFS-сервер.

Важные ограничения и правила:

- Группа `nfs_exporters` должна содержать только те узлы, на которых разрешено размещать NFS-ресурс. Если группа слишком широкая, ресурс может быть размещён на лишних узлах.
- Запускайте роль только через `hosts: nfs_exporters`, а не через более широкую группу (например, `workers` или `all`).
- Если параметр `autoplace_nodes` явно указывает узлы за пределами группы `nfs_exporters`, роль завершится ошибкой — неявного расширения размещения не происходит.

Таблица 12 – Практические рекомендации

Количество экспортов	Способ управления
Небольшие и обычные ресурсы	Используйте <code>nfs_manage_resources: true</code> — этого достаточно.
Повторное использование существующего DRBD-ресурса	На ресурсе должна уже быть создана файловая система. Если её нет — роль завершится ошибкой (форматирование не выполняется автоматически).
Очень большие ресурсы	Первичная синхронизация может занять много времени. Роль в этом режиме синхронно ждёт завершения синхронизации, что может привести к таймаутам.

Для больших ресурсов лучше использовать двухэтапную схему:

- Сначала создайте DRBD-ресурс и дождитесь завершения его синхронизации отдельно (вне роли).
- Затем запускайте настройку NFS-экспорта с параметром `nfs_manage_resources: false`.

Этот подход помогает избежать: таймаута синхронизации (`sync_timeout`), внешних таймаутов Ansible, обрывов SSH-соединения, таймаутов в CI-заданиях.

9.5.6. Versity S3-шлюз

Настройки S3-шлюза задаются в глобальном файле переменных: `trok-deploy/group_vars/all.yml` в корне проекта Ansible.

Пример настройки S3-шлюза:

```
trok_s3_gw_manage_resources: true
trok_s3_gw_instance_name: dev
trok_s3_gw_vip: "192.0.2.251"
trok_s3_gw_vip_cidr_netmask: 24
trok_s3_gw_storage_pool: lvm_pool
trok_s3_gw_resource_size: "20GiB"
trok_s3_gw_rg_place_count: 2
trok_s3_gw_port_s3: 7070
trok_s3_gw_port_admin: 7080
trok_s3_gw_port_webui: 7071
trok_s3_gw_region: us-east-1
trok_s3_gw_root_access_key_id: "admin"
trok_s3_gw_root_secret_access_key: "{{ vault_s3gw_secret }}"
```

Таблица 13 – Обязательные параметры

Параметр	Обязательность
<code>trok_s3_gw_vip</code>	Всегда (должен быть валидным IPv4-адресом)
<code>trok_s3_gw_storage_pool</code>	Если <code>trok_s3_gw_manage_resources: true</code>
<code>trok_s3_gw_resource_size</code>	Если <code>trok_s3_gw_manage_resources: true</code>

S3-шлюз работает как высокодоступный сервис на одном DRBD-ресурсе:

- В каждый момент времени ровно один узел из группы `s3gw_gateways` держит роль Primary.
- На этом узле автоматически:
 - монтируется ФС,
 - поднимается виртуальный VIP,
 - запускается сервис `trok-s3-gw@<instance>`.
- Переключение на другой узел выполняется автоматически через `drbd-reactor`.

Пакет `trok-s3-gw` устанавливается из APT-репозитория TROK через зависимость `pre_install`.

Переопределяйте `trok_s3_gw_package_name` только в том случае, если имя пакета в вашем репозитории отличается от стандартного.

Важные ограничения и правила:

- Количество узлов в группе `s3gw_gateways` должно быть не меньше значения `trok_s3_gw_rg_place_count` (иначе негде будет размещать ресурс).
- Запускайте роль только через `hosts: s3gw_gateways`, а не через более широкую группу (например, `workers` или `all`).

Безопасность учётных данных:

- Никогда не храните секретный ключ `trok_s3_gw_root_secret_access_key` в открытом виде в `all.yml`.
- Используйте Ansible Vault для шифрования чувствительных данных.
- Значения по умолчанию `admin/admin` пригодны только для лабораторного стенда – никогда не используйте их в продакшене.

Таблица 14 – Практические рекомендации по управлению ресурсами

Сценарий	Поведение роли
Создание нового ресурса	Роль создаёт DRBD-ресурс, форматирует ФС и настраивает шлюз.
Повторный запуск на существующем ресурсе	Роль работает в режиме <code>validate-only</code> (проверяет, но ничего не меняет), если на ресурсе уже есть живой Primary.
Очень большие ресурсы	Первичная синхронизация может занять много времени. Используйте двухэтапную схему.

Для больших ресурсов рекомендуется двухэтапный подход:

- Сначала создайте DRBD-ресурс и дождитесь завершения его синхронизации отдельно (вне роли).
- Затем запускайте настройку S3-шлюза с параметром `trok_s3_gw_manage_resources: false`.

Это помогает избежать таймаутов синхронизации (sync_timeout), обрывов SSH-соединения и проблем с CI-заданиями.

Хранение секрета S3 в Ansible Vault

Создайте зашифрованный файл, который применяется к группе all:

```
mkdir -p group_vars/all
ansible-vault create group_vars/all/vault.yml
```

Содержимое файла:

```
vault_s3gw_secret: "<секретный_ключ>"
```

В group_vars/all.yml укажите:

```
trok_s3_gw_root_access_key_id: "admin"
trok_s3_gw_root_secret_access_key: "{{ vault_s3gw_secret }}"
```

Запускайте playbook с vault-id, принятым в организации, либо с интерактивным запросом:

```
ansible-playbook -i inventory.ini site.yml --ask-vault-pass
```

Не помещайте открытый секрет в inventory.ini, all.yml, журнал Ansible или систему контроля версий.

9.6. Плейбук

Для возможности поэтапного запуска каждой стадии назначьте отдельный play-level tag:

```
---
- name: Развернуть controllers
  hosts: controllers
  gather_facts: true
  become: true
  become_user: root
  tags:
    - controllers
  roles:
    - { role: astra.trok.controller }

- name: Развернуть workers
  hosts: workers
  gather_facts: true
```

```

become: true
become_user: root
tags:
  - workers
roles:
  - { role: astra.trok.worker }

- name: Подготовить LVM pool
  hosts: workers
  gather_facts: true
  become: true
  become_user: root
  tags:
    - storage
  roles:
    - { role: astra.trok.storage_pool_lvm }

- name: Настроить iSCSI
  hosts: iscsi_exporters
  gather_facts: true
  become: true
  become_user: root
  tags:
    - iscsi
  roles:
    - { role: astra.trok.iscsi_target }

- name: Настроить NFS
  hosts: nfs_exporters
  gather_facts: true
  become: true
  become_user: root
  tags:
    - nfs
  roles:
    - { role: astra.trok.nfs_export }

- name: Развернуть S3-шлюз
  hosts: s3gw_gateways
  gather_facts: true
  become: true
  become_user: root
  tags:
    - s3gw
  roles:
    - { role: astra.trok.s3gw }

```

Если используется LVM Thin, замените роль `astra.trok.storage_pool_lvm` на `astra.trok.storage_pool_lvm_thin`, сохранив tag `storage`.

9.7. Запуск деплоя

Полный запуск:

```
ansible-playbook -i inventory.ini site.yml --ask-vault-pass
```

Поэтапный запуск:

```
ansible-playbook -i inventory.ini site.yml --tags controllers
ansible-playbook -i inventory.ini site.yml --tags workers
ansible-playbook -i inventory.ini site.yml --tags storage
ansible-playbook -i inventory.ini site.yml --tags iscsi --ask-vault-pass
ansible-playbook -i inventory.ini site.yml --tags nfs --ask-vault-pass
ansible-playbook -i inventory.ini site.yml --tags s3gw --ask-vault-pass
```

Tag выбирает стадию. Параметр `--limit` выбирает хосты и не должен использоваться как замена стадии: один и тот же worker может входить одновременно в `workers`, `iscsi_exporters`, `nfs_exporters` и `s3gw_gateways`.

Перед полным запуском выполните статическую проверку:

```
ansible-playbook -i inventory.ini site.yml --syntax-check
ansible-playbook -i inventory.ini site.yml --list-hosts
ansible-playbook -i inventory.ini site.yml --list-tags
ansible-playbook -i inventory.ini site.yml --tags workers --list-tasks
```

`--list-hosts` проверяет состав хостов каждого play. В ряде версий Ansible эта команда выводит все plays даже при одновременном указании `--tags`; поэтому изоляцию стадии подтверждайте через `--list-tasks --tags <стадия>`. У выбранной стадии должен быть непустой список задач, у остальных стадий – пустой.

9.8. Проверка после деплоя

9.8.1. Базовые сервисы

Выполните на каждом контроллере:

```
sudo systemctl status trok-controller
sudo systemctl status trok-cp-endpoint
sudo systemctl status trok-auth
```

На рабочих узлах (узлах с ролью worker)

Выполните на каждом рабочем узле:

```
sudo systemctl status trok-worker
```

Если сервис не запустился:

Посмотрите логи:

```
sudo journalctl -u trok-worker -n 50 --no-pager
```

Попробуйте перезапустить сервис:

```
sudo systemctl restart trok-worker
```

Если перезапуск не помог – проверьте:

- наличие конфигурационных файлов,
- доступность дисков и сетевых путей,
- наличие ошибок в логах (journalctl -xe).

9.8.2. Пул хранения

После настройки хранилища необходимо убедиться, что пул создан корректно и доступен для использования. Откройте веб-интерфейс TROK и выполните следующие проверки:

- Узлы кластера – убедитесь, что в интерфейсе отображаются все рабочие узлы (workers), которые вы указали в инвентаре. Они должны быть в статусе Online или Active.
- Пул хранения – проверьте, что созданный вами пул (lvm_pool или другой) присутствует в списке пулов хранения.
- Привязка пула к узлам – убедитесь, что созданный пул привязан именно к тем узлам, где вы планируете его использовать (обычно это все узлы из группы workers или их подмножество).

Если сущности не отображаются:

Таблица 15 – Поиск проблем

Проблема	Возможное решение
Узел не отображается или в статусе Offline	Проверьте сеть, запущен ли trok-worker на узле, доступен ли SSH.
Пул отсутствует в списке	Убедитесь, что роль storage_pool_lvm (или другая) была успешно выполнена. Проверьте логи Ansible на наличие ошибок.
Пул не привязан к ожидаемым узлам	Возможно, вы указали неправильные узлы в группе workers или параметрах пула. Проверьте переменные в all.yml.

9.8.3. iSCSI

На каждом узле-экспортёре (где работает iSCSI-цель) выполните следующие команды.

Установите iSCSI-инициатор (пакет open-iscsi), если он еще не установлен:

```
sudo apt update
sudo apt install -y open-iscsi
```

Проверьте роль DRBD-ресурса (Primary/Secondary):

```
sudo drbdadm role ISCSI-APP
```

Проверьте статус drbd-reactor (управление сервисами):

```
sudo drbd-reactorctl status
```

Посмотрите конфигурацию iSCSI-целей (LIO)

```
sudo targetcli ls
```

Что должно отображаться:

- drbdadm role – ровно один узел в кластере должен показывать Primary, остальные — Secondary.
- drbd-reactorctl status – сервисы (Filesystem, IPaddr2 и др.) должны быть в статусе Started на Primary-узле и Stopped на Secondary.

– `targetcli ls` – должна отображаться iSCSI-цель с IQN, указанным в настройках, и LUN-ы с нужным размером.

Выполните на клиентской машине, которая будет использовать iSCSI-диск (узел-инициатор или клиент, который подключается к iSCSI) следующие команды. Обнаружьте цель на экспортере:

```
sudo iscsiadm -m discovery -t sendtargets -p <ip_цели>:3260
```

Настройте аутентификацию CHAP (если используется):

```
sudo iscsiadm -m node -T iqn.2026-05.trok.sds:iscsi-app -p <ip_цели>:3260 -o update -n node.session.auth.authmethod -v CHAP
```

```
sudo iscsiadm -m node -T iqn.2026-05.trok.sds:iscsi-app -p <ip_цели>:3260 -o update -n node.session.auth.username -v chap-user
```

```
sudo iscsiadm -m node -T iqn.2026-05.trok.sds:iscsi-app -p <ip_цели>:3260 -o update -n node.session.auth.password -v chap-password
```

Подключитесь к цели:

```
sudo iscsiadm -m node -T iqn.2026-05.trok.sds:iscsi-app -p <ip_цели>:3260 --login
```

Ожидаемый результат:

После `--login` подключение должно быть успешным.

Проверьте, что на инициаторе появился новый диск:

```
lsblk
```

Должен отобразиться новый блочный диск (например, `/dev/sdb` или `/dev/sdc`).

Таблица 16 – Ожидаемое состояние системы

Компонент	Ожидание
DRBD-ресурс	Ровно один узел-экспортер в статусе Primary, остальные – Secondary
iSCSI-цель	Доступна по IP-адресу, который вы указали в <code>bind_ip</code>) на порту 3260

Подключение инициатора	Инициатор успешно обнаруживает цель и подключается по указанному IQN
Аутентификация	Если используется CHAP — подключение проходит только с правильными логином и паролем

Таблица 17 – Поиск проблем

Проблема	Проверьте
Нет Primary на экспортёре	Проверьте DRBD-статус: <code>sudo drbdadm status</code>
Цель не видна по IP	Проверьте, поднят ли плавающий VIP на экспортёре: <code>ip a</code>
Инициатор не подключается	Убедитесь, что CHAP-учётные данные совпадают, и порт 3260 открыт в файерволе
<code>targetcli ls</code> пуст	Проверьте, была ли создана цель: возможно, роль <code>iscsi_target</code> не завершилась успешно, посмотрите логи Ansible

9.8.4. NFS

Выполните на каждом узле из группы `nfs_exporters` следующие команды/
Установите NFS-клиент (пакет `nfs-common`), если он еще не установлен:

```
sudo apt update
sudo apt install -y nfs-common
```

Проверьте роль DRBD-ресурса (Primary/Secondary):

```
sudo drbdadm role NFS-APP
```

Проверьте статус `drbd-reactor` (управление сервисами):

```
sudo drbd-reactorctl status
```

Посмотрите активные NFS-экспорты:

```
sudo exportfs -v
```

Что должно отображаться:

- drbdadm role – ровно один узел в кластере должен показывать Primary, остальные – Secondary.
- drbd-reactorctl status – сервисы (Filesystem, IPaddr2, exportfs и др.) должны быть в статусе Started на Primary-узле и Stopped на Secondary.
- exportfs -v – на Primary узле должен отображаться экспортируемый каталог (/srv/nfs/nfs-app), доступный для клиентов с указанными параметрами (например, rw, sync, no_subtree_check, no_root_squash).

Выполните на клиентской машине, которая будет использовать NFS-шару следующие команды:

Создайте точку монтирования (если её нет):

```
sudo mkdir -p /mnt
```

Примонтируйте NFS-шару с версией протокола 3:

```
sudo mount -t nfs -o vers=3 192.0.2.250:/srv/nfs/nfs-app /mnt
```

Проверьте возможность записи – создайте тестовый файл:

```
sudo touch /mnt/probe.txt
```

Отмонтируйте шару:

```
sudo umount /mnt
```

Ожидаемый результат:

- Монтирование должно пройти без ошибок.
- Файл /mnt/probe.txt должен создаваться без ошибок.
- Отмонтирование должно пройти без ошибок.

Таблица 18 – Ожидаемое состояние системы

Компонент	Ожидание
-----------	----------

DRBD-ресурс	Ровно один узел-экспортёр в статусе Primary, остальные – Secondary.
NFS-экспорт	Экспорт доступен по IP-адресу, который вы указали в bind_ip.
Подключение клиента	Монтирование и запись проходят успешно.
Повторное монтирование	После переключения роли экспорт поднимается автоматически, и клиент может подключиться снова без ручных действий.

Таблица 19 – Поиск проблем

Проблема	Проверьте
Нет Primary на экспортёре	Проверьте DRBD-статус: <code>sudo drbdadm status NFS-APP</code> .
Экспорт не отображается (<code>exportfs -v</code> пуст).	Проверьте статус <code>drbd-reactorctl status</code> – возможно, сервис <code>exportfs</code> не запустился.
Клиент не может примонтировать	Проверьте, поднят ли плавающий VIP на экспортёре: <code>ip a</code> ; убедитесь, что порт NFS (2049) открыт в файерволе.
Нет прав на запись	Проверьте права на каталог <code>/srv/nfs/nfs-app</code> на экспортёре (должны быть доступны для записи).
После переключения экспорт не доступен	Проверьте, что <code>drbd-reactor</code> корректно управляет сервисами: логи <code>sudo journalctl -u drbd-reactor</code> .

9.8.5. Versity S3-шлюз

Выполните на каждом узле, входящем в группу `s3gw_gateways`, команды:
Проверьте роль DRBD-ресурса (Primary/Secondary):

```
sudo drbdadm role trok-s3-gw-root
```

Проверьте статус управления сервисами через drbd-reactor:

```
sudo drbd-reactorctl status trok-s3-gw
```

Проверьте, активен ли сервис S3-шлюза:

```
sudo systemctl is-active trok-s3-gw@dev
```

Проверьте, смонтирована ли файловая система шлюза:

```
sudo findmnt --mountpoint /srv/trok-s3-gw
```

Что должно отображаться:

- drbdadm role – ровно один узел в кластере должен показывать Primary, остальные – Secondary.
- drbd-reactorctl status – сервисы (Filesystem, IPaddr2, trok-s3-gw@dev и др.) должны быть в статусе Started на Primary-узле и Stopped на Secondary.
- systemctl is-active – на Primary-узле должен вернуть active, на Secondary – inactive.
- findmnt – на Primary-узле должна отображаться точка монтирования /srv/trok-s3-gw (с указанием устройства DRBD). На Secondary монтирования быть не должно.

Для проверки доступности сервиса, с любого хоста, который имеет сетевой доступ к плавающему VIP-адресу (включая сами gateway-узлы), выполните:

```
sudo findmnt --mountpoint /srv/trok-s3-gw
```

10. СЦЕНАРИЙ ДЛЯ ДЕПЛОЯ НА КЛАСТЕРА

Эта глава описывает переход от обычного кластера TROK к высокодоступной (HA) схеме с использованием роли `ha_controller` для тех, кто уже знаком с базовым развертыванием TROK и хочет обеспечить отказоустойчивость управляющего слоя.

Что нужно иметь перед переходом к HA:

- Базовый кластер TROK (controller, worker) – развернутый и работающий.
- Все будущие HA-узлы уже входят в обычный кластер TROK.
- Готовый пул хранения (LVM, ZFS или другой) – созданный и доступный на узлах.
- Значение `ha_pool_name` известно и совпадает с именем этого пула.
- Virtual-IP-адрес заранее выделен и доступен в нужной сети.
- Этап настройки iSCSI можно пропустить, если он не требуется в вашей схеме.

Рекомендуется не запускать HA первым шагом. Сначала разверните обычный кластер TROK. Затем создайте пул хранения. И только после этого запускайте роль `astra.trok.ha_controller`.

Что получится в итоге:

- База данных TROK controller будет вынесена в отдельный HA-ресурс на DRBD.
- На узлах, отведённых под HA, будет настроен `drbd-reactor` для автоматического переключения.
- Клиенты и сервисы смогут обращаться к контроллеру по плавающему VIP-адресу (`ha_controller_vip`), а не к фиксированному узлу.

10.1. Общая информация о `ha_controllers`

Группа `ha_controllers` – это группа инвентаря, в которой перечислены точные узлы, участвующие в HA-кластере. Именно по этому списку роль определяет, где можно размещать HA-ресурс базы данных.

Важно. Не добавляйте в `ha_controllers` лишние узлы. Это может привести к тому, что НА-ресурс будет размещён не там, где вы ожидаете.

Первый IP-адрес в списке `controllers` (`controllers[0]`) считается основным для подготовки НА. На этом узле выполняются ключевые шаги начальной настройки.

Важно. Если первым указать не тот узел, настройка НА будет выполнена на неверном узле, и кластер может работать некорректно.

Роль `ha_controller` не создаёт пул хранения самостоятельно. Она ожидает, что нужный пул уже существует (например, после выполнения роли `astra.trok.storage_pool_lvm_thin`).

Важно. `ha_pool_name` должен совпадать с уже созданным пулом. Если `ha_pool_name` указывает на несуществующий пул, создание НА-ресурса завершится ошибкой.

`ha_controller_vip` – это виртуальный IP-адрес, который всегда находится на активном НА-узле. Клиенты и сервисы должны подключаться именно к этому адресу, а не к конкретному узлу.

Важно. Если VIP не задан, конфликтует с другими адресами или неправильно маршрутизируется в вашей сети, доступность контроллера для клиентов будет нарушена.

10.2. Конфигурация файла инвентаря `inventory.ini`

Пример конфигурации:

```
[controllers]
  cmbnd      ansible_host=192.0.2.11      ansible_user=ansible      ansible_port=22
  ansible_connection=ssh      ansible_ssh_private_key_file=~/.ssh/id_ed25519
  ansible_ssh_common_args="-o      StrictHostKeyChecking=no      -o
  UserKnownHostsFile=/dev/null"

[workers]
  cmbnd      ansible_host=192.0.2.11      ansible_user=ansible      ansible_port=22
  ansible_connection=ssh      ansible_ssh_private_key_file=~/.ssh/id_ed25519
  ansible_ssh_common_args="-o      StrictHostKeyChecking=no      -o
  UserKnownHostsFile=/dev/null"
  sat1      ansible_host=192.0.2.12      ansible_user=ansible      ansible_port=22
  ansible_connection=ssh      ansible_ssh_private_key_file=~/.ssh/id_ed25519
  ansible_ssh_common_args="-o      StrictHostKeyChecking=no      -o
  UserKnownHostsFile=/dev/null"
```

```

    sat2      ansible_host=192.0.2.13      ansible_user=ansible      ansible_port=22
ansible_connection=ssh      ansible_ssh_private_key_file=~/.ssh/id_ed25519
ansible_ssh_common_args="-o      StrictHostKeyChecking=no      -o
UserKnownHostsFile=/dev/null"

    [ha_controllers]
    cmbnd      ansible_host=192.0.2.11      ansible_user=ansible      ansible_port=22
ansible_connection=ssh      ansible_ssh_private_key_file=~/.ssh/id_ed25519
ansible_ssh_common_args="-o      StrictHostKeyChecking=no      -o
UserKnownHostsFile=/dev/null"
    sat1      ansible_host=192.0.2.12      ansible_user=ansible      ansible_port=22
ansible_connection=ssh      ansible_ssh_private_key_file=~/.ssh/id_ed25519
ansible_ssh_common_args="-o      StrictHostKeyChecking=no      -o
UserKnownHostsFile=/dev/null"
    sat2      ansible_host=192.0.2.13      ansible_user=ansible      ansible_port=22
ansible_connection=ssh      ansible_ssh_private_key_file=~/.ssh/id_ed25519
ansible_ssh_common_args="-o      StrictHostKeyChecking=no      -o
UserKnownHostsFile=/dev/null"

    [all:vars]
    ansible_become=true

```

10.3. Настройка переменных (group_vars/all.yml)

10.3.1. База для НА

Пример содержания файла:

```

controllers:
  - 192.0.2.11

workers:
  - 192.0.2.11
  - 192.0.2.12
  - 192.0.2.13

drbd_replication_network: "192.0.2.0/24"
enable_gateway: true
ufw_enable: false
dist_upgrade: false
cluster_member: true
sp_cluster_member: true
drbd_auto_reboot: true

```

10.3.2. Пул хранения для НА

Пример содержания файла:

```

storage_pool_lvm_thin_pools:
  - pool_name: thinpool
    vg_name: drbdpool
  disks:

```

```
- /dev/vdb
size: "8.5G"
```

10.3.3. HA-переменные

```
ha_db_resource_group: "HA-TROK-DB-GRP"
ha_db_resource_name: "HA-TROK-DB"
ha_db_size: "500M"
ha_db_sync_timeout: 1800
ha_pool_name: "thinpool"
ha_controller_vip: "192.0.2.248"
ha_controller_vip_netmask: "24"
```

10.3.4. Опциональная глобальная настройка DRBD на HA worker-узлах

Эта настройка по-прежнему относится к роли worker и применяется на этапе развёртывания worker-узлов до того, как ha_controller настраивает HA-ресурс.

```
worker_drbd_tuning_enabled: true
worker_drbd_tuning_profile: "25g-ssd"
worker_drbd_tuning_resources_per_node: 100
worker_drbd_tuning_reboot: false
```

Правила выбора параметров настройки DRBD:

`worker_drbd_tuning_profile` – определяет, какая комбинация сетевого оборудования и типа дисков используется на узле (например, 10g-hdd или 25g-ssd). Это влияет на общую производительность DRBD-репликации.

`worker_drbd_tuning_resources_per_node` – ограничивает производительность каждого отдельного DRBD-ресурса на узлах с высокой плотностью (где одновременно работает много ресурсов). Чем больше ресурсов, тем более «консервативными» становятся настройки для каждого из них, чтобы избежать перегрузки системы.

10.4. Плейбук site.yml

Пример содержания файла site.yml:

```
- name: Развернуть controller
  hosts: controllers
  gather_facts: true
  become: true
  become_user: root
```

```

roles:
  - { role: astra.trok.controller }

- name: Развернуть workers
  hosts: workers
  gather_facts: true
  become: true
  become_user: root
  roles:
    - { role: astra.trok.worker }

- name: Подготовить LVM Thin pool
  hosts: workers
  gather_facts: true
  become: true
  become_user: root
  roles:
    - { role: astra.trok.storage_pool_lvm_thin }

- name: Настроить HA controller
  hosts: ha_controllers
  gather_facts: true
  become: true
  become_user: root
  roles:
    - { role: astra.trok.ha_controller }

```

Если HA-ресурс должен использовать другую подсистему хранения:

- замените роль подготовки пула на нужную;
- укажите в `ha_pool_name` имя созданного пула.

Запуск:

```
ansible-playbook -i inventory.ini site.yml
```

Повторный запуск:

- Повторный запуск `site.yml` допустим после успешного развёртывания HA.
- Это корректно работает даже после переключения на резервный узел.

Активный узел может отличаться от `controllers[0]` – это штатная ситуация.

Важное ограничение:

- Если первый запуск HA был прерван (оборвался на полпути), обычный повторный запуск не исправит ситуацию.
- Для восстановления используйте отдельный сценарий восстановления, а не повторный запуск `site.yml`.

10.5. Диагностика после оборванного первого запуска HA

Первичный перевод controller в HA включает остановку сервисов, создание DRBD-ресурса, подготовку файловых систем, перенос БД и передачу владения сервисами drbd-reactor. Прерывание между этими стадиями может оставить частичное состояние.

Важно. Команды этого пункта предназначены только для чтения состояния. Не удаляйте marker-файлы, каталоги `.orig`, `systemd units` или DRBD-конфигурацию; не выполняйте `drbdadm primary`, `drbdadm secondary`, `rm`, `mv`, `mkfs` и повторный запуск `site.yml`, пока состояние не классифицировано однозначно.

Роли коллекции выбирают controller endpoint по durable состоянию HA, а не по кратковременной доступности TCP-порта:

- в non-HA и fresh HA до появления хотя бы одного HA marker используется первый адрес из controllers;
- после обнаружения хотя бы одного marker на любом узле `ha_controllers` разрешён только `http://<ha_controller_vip>:8080`;
- если markers не найдены, но проверить их на всех HA-узлах не удалось, роль завершается `fail closed`;
- настроенный HA никогда не переключается на физический controller при недоступности VIP.

Модуль `astra.trok.trok_wait_controller` проверяет выбранный полный URL через аутентифицированный read-only вызов API. Временные transport errors и HTTP 408, 425, 429, 500, 502, 503, 504 повторяются внутри одной задачи; ошибки аутентификации и конфигурации завершают её сразу. Не обходите такую ошибку ручной подстановкой физического адреса.

10.5.1. Проверка marker-файлов на всех ha_controllers

Выполните на каждом узле группы `ha_controllers`:

```
sudo test -e /etc/drbd-reactor.d/HA-TROK-DB.toml && echo reactor-marker-present
sudo test -e /etc/systemd/system/trok-controller.service.d/10-ha-db-mount.conf
&& echo controller-marker-present
```

```
sudo test -e /etc/systemd/system/trok-auth.service.d/10-ha-db-mount.conf && echo
auth-marker-present

sudo ls -ld \
  /var/lib/trok-controller \
  /var/lib/trok-controller.orig \
  /var/lib/trok-auth \
  /var/lib/trok-auth.orig 2>&1
```

Отсутствующий файл в выводе ls является диагностическим признаком и не должен автоматически создаваться или удаляться. Состояние markers необходимо сравнить между всеми HA-узлами: наличие хотя бы одного marker означает, что дальнейшие API-проверки допустимы только через VIP.

10.5.2. Проверка DRBD на всех ha_controllers

```
sudo drbdadm role HA-TROK-DB

sudo drbdadm status HA-TROK-DB

sudo drbdsetup status HA-TROK-DB --verbose --statistics

sudo systemctl status drbd-reactor --no-pager

sudo journalctl -u drbd-reactor -n 200 --no-pager
```

В здоровом состоянии ровно один узел имеет локальную роль Primary, остальные – Secondary; все диски должны иметь состояние UpToDate.

10.5.3. Проверка файловых систем и mounts

На каждом HA-узле выполните:

```
sudo findmnt /var/lib/trok-controller

sudo findmnt /var/lib/trok-auth

sudo findmnt -S /dev/drbd/by-res/HA-TROK-DB/0

sudo findmnt -S /dev/drbd/by-res/HA-TROK-DB/1

sudo blkid \
  /dev/drbd/by-res/HA-TROK-DB/0 \
  /dev/drbd/by-res/HA-TROK-DB/1
```

Обе файловые системы могут быть смонтированы только на текущем Primary. На Secondary эти устройства не должны быть смонтированы.

10.5.4. Проверка сервисов и VIP

```
sudo systemctl status \
    trok-controller.service \
    trok-auth.service \
    trok-cp-endpoint.service \
    apache2.service \
    --no-pager

sudo systemctl is-enabled \
    trok-controller.service \
    trok-auth.service \
    trok-cp-endpoint.service \
    apache2.service

ip -br address show
```

В свежем здоровом HA сервисы Controller, Auth, Control Plane Endpoint и Apache, а также VIP активны только на том же узле, который является Primary. Их direct boot должен быть отключён: запуском и остановкой управляет drbd-reactor.

Проверьте параметр controllers в клиентской конфигурации на HA controller и worker-узлах:

```
sudo awk -F= '
    $1 ~ /^[[:space:]]*controllers[[:space:]]*$/ { print $2 }
' /etc/trok/trok-client.conf
```

Ожидается ha_controller_vip:8080 первым значением, затем уникальные физические адреса ha_controllers. Остальные параметры файла не должны изменяться.

Только если предыдущие проверки уже доказали здоровое HA-состояние, проверьте VIP аутентифицированным read-only запросом:

```
sudo python3 - 'http://192.0.2.248:8080' <<'PY'
import sys

from trok import MultiTrok

with MultiTrok([sys.argv[1]]) as client:
    result = client.node_list_raise(limit=1)
    print(result)
PY
```

Обычная проверка открытого порта или HTTP-корня не подтверждает готовность TROK API.

10.5.5. Правило принятия решения

Обычный повторный запуск site.yml допустим только в одном из следующих состояний:

1. Предыдущий первичный HA-деплой полностью завершился, а текущая проверка подтверждает один Primary, UpToDate, mounts, сервисы и VIP на одном владельце.

2. Первичный запуск не дошёл до создания ни одного marker-файла, каталоги .orig отсутствуют, DRBD-устройства HA DB не смонтированы, а исходный non-HA controller полностью работоспособен.

Остановите процедуру и передайте диагностику в поддержку TROK, если выполняется хотя бы одно условие:

- присутствует только часть marker-файлов;
- существует один из каталогов .orig, но состояние второго отличается;
- Primary отсутствует либо Primary больше одного;
- DRBD не UpToDate;
- устройство смонтировано на Secondary или одновременно на нескольких узлах;
- файловая система не определяется либо blkid показывает неожиданный тип;
- сервисы и VIP принадлежат разным узлам;
- Apache отсутствует в актуальной fresh-HA reactor-конфигурации либо в ней остаётся ссылка на nginx;
- параметры trok-client.conf расходятся между HA controller и worker-узлами;
- marker-файлы не найдены, но хотя бы один узел ha_controllers недоступен для проверки;
- состояние не соответствует двум разрешённым вариантам выше.

Ручной rollback частично выполненной HA-миграции не входит в настоящее руководство. Он требует отдельного регламента, резервной копии и решения специалиста, ответственного за данные.

10.6. Проверка корректности поднятия HA

На каждом HA-узле выполните:

```
sudo systemctl status drbd-reactor  
  
sudo drbdadm role HA-TROK-DB
```

Ожидаемый результат:

- drbd-reactor активен (active).
- Ровно один узел в статусе Primary, остальные – Secondary.

Примечание. При повторном запуске site.yml активный узел может остаться тем же или измениться после переключения — это нормально.

На активном HA-узле выполните:

```
sudo systemctl status trok-controller  
  
sudo systemctl status trok-cp-endpoint  
  
sudo systemctl status trok-auth  
  
ip addr show
```

Ожидаемый результат:

- Сервисы trok-controller, trok-cp-endpoint, trok-auth активны только на активном узле.
- Виртуальный IP (ha_controller_vip) присутствует на активном узле.

Важное замечание:

Если после повторного запуска активная роль перешла с controllers[0] на другой HA-узел, это не ошибка. Базовые роли controller и worker не пытаются запускать сервисы локально на пассивных узлах. Управление сервисами контроллера остаётся за drbd-reactor.

Для проверки с клиентской стороны откройте в браузере:

```
http://<ha_controller_vip>
```

Ожидаемый результат: вход в TROK и работа через VIP доступны.

10.7. Возможные ошибки конфигурации

При настройке HA-контроллера некорректные значения переменных или неверный состав групп инвентаря могут привести к сбоям развёртывания. В таблице ниже перечислены основные проблемные сценарии и их влияние на систему.

Таблица 20 – Типовые ошибки конфигурации HA

Проблема	Описание
Избыточный состав группы <code>ha_controllers</code>	Включение в группу узлов, не предусмотренных архитектурой, приводит к попытке размещения HA-ресурса на нецелевых хостах, что нарушает предполагаемую топологию.
Несоответствие <code>ha_pool_name</code> фактическому пулу	Указание имени пула, отсутствующего в системе, влечёт за собой ошибку на этапе создания HA-ресурса, поскольку роль не выполняет создание пула самостоятельно.
Некорректный порядок узлов в группе <code>controllers</code>	Первый узел в списке (<code>controllers[0]</code>) используется как основной для подготовки HA. Неверный выбор этого узла приводит к выполнению критических шагов на нецелевом хосте.

10.8. Дальнейшие шаги после развёртывания HA

Развёртывание HA-контроллера завершает этап обеспечения отказоустойчивости управляющего слоя. Настройка сервисов экспорта данных

(iSCSI, NFS, S3) является следующим, независимым этапом и не входит в состав процедуры установки НА.

Порядок действий для развёртывания сервисов экспорта:

- Убедитесь в успешном завершении развёртывания НА-контроллера.
- Активируйте поддержку шлюзов: `enable_gateway: true`.
- Определите группы узлов-экспортёров в инвентаре: `iscsi_exporters`, `nfs_exporters`, `s3gw_gateways`.
- Запустите соответствующие роли:
 - `astra.trok.iscsi_target`;
 - `astra.trok.nfs_export`;
 - `astra.trok.s3gw`.

Конфигурация сервисов экспорта выполняется с использованием тех же переменных (`iscsi_targets`, `nfs_exports`, `trok_s3_gw_*`), что и в сценарии обычного кластера.

11. ЗАВЕРШЕНИЕ УСТАНОВКИ

После завершения установки необходимо проверить, что компонент DRBD корректно интегрирован в систему. Ниже приведены ключевые команды проверки с краткими пояснениями для каждой строки или блока. Успешное выполнение команд указывает на готовность модуля DRBD к использованию.

Процедура верификации модуля DRBD применяется исключительно на узлах хранения данных (trok-worker, trok-combined). Если на узле размещается только trok-controller, наличие модуля DRBD не обязательно.

Выполните проверку версии ядра ОС (kernel release):

```
sudo uname -r
```

Пример вывода подтверждает загруженную версию ядра (здесь: 6.1.152-1-generic):

```
6.1.152-1-generic
```

DRBD – это модуль ядра Linux. Он должен быть скомпилирован и загружен именно для той версии ядра, которая сейчас активна. Несовпадение версии ядра и версии DRBD может привести к неработоспособности хранения, поэтому следующим шагом проверьте версию DRBD, чтобы сопоставить данные.

Для этого выполните команду, которая отобразит перечень установленных пакетов, применяя фильтр для вывода только тех, в наименовании которых присутствует «drbd»:

```
sudo apt list --installed | grep drbd
```

Пример вывода:

```
drbd-6.1.152-1-generic/stable,stable,now 9.2.12-4 amd64 [установлен] # Установлен
связанный с ядром модуль DRBD версии 9.2.12.
```

```
drbd-reactor/stable,now 1.4.1-1 amd64 [установлен] # Установлен компонент
реактора для управления ресурсами.
```

```
drbd-ueficert/now 9.2.11-1 all [установлен, локальный] # Установлен сертификат
UEFI для DRBD.
```

```
drbd-utils/stable,now 9.28.0-1 amd64 [установлен] # Установлены утилиты DRBD
версии 9.28.0.
```

Анализ представленного вывода подтверждает, что все необходимые компоненты (ядро DRBD, утилиты, интеграционные модули, сертификаты) установлены корректно. В случае отсутствия какого-либо из указанных элементов в списке результатов, требуется их установка. Судя по префиксу «drbd-6.1.152-1-generic...», используется модуль DRBD версии 9.2.12. Он подходит к текущему ядру, потому что его версия совпадает с тем, что показывает команда `sudo uname -r`.

Выполните команду для отображения информации о версии и сборке DRBD Admin utility:

```
sudo drbdadm --version
```

Пример вывода:

```
DRBDADM_BUILDTAG=GIT-hash:\
ba2ce9037989b6141222c7901d1219cf852949f1configure.ac\ build\ by\ root@linux\,\
2025-01-17\ 18:32:01 # Метка сборки с хэшем Git и датой компиляции.

DRBDADM_API_VERSION=2 # Версия API drbdadm (2).

DRBD_KERNEL_VERSION_CODE=0x09020c # Код версии ядра DRBD (9.2.12).

DRBD_KERNEL_VERSION=9.2.12 # Человеко-читаемая версия ядра DRBD.

DRBDADM_VERSION_CODE=0x091c00 # Код версии drbdadm (9.28.0).

DRBDADM_VERSION=9.28.0 # Версия утилиты drbdadm.
```

Вывод показывает, что утилита для управления DRBD (drbdadm) установлена и работает. Любая рассинхронизация или отсутствие – свидетельство неправильной установки и источник будущих проблем кластера.

Если версии не совпадают или утилита не работает – требуется устранить рассогласования в установке.

Проверьте загруженные модули ядра, фильтруя по «drbd»:

```
sudo lsmod | grep -i drbd
```

Пример вывода:

```
drbd      819200   0   # Основной модуль DRBD загружен (размер 819200 байт,
используется 0 процессами).

lru_cache 16384   1 drbd # Зависимый модуль LRU-кэша, используемый DRBD.

libcrc32c 16384   6 nf_conntrack,nf_nat,dm_persistent_data,nf_tables,drbd,sctp
# Модуль CRC32C, включая DRBD как одного из клиентов.
```

DRBD должен быть не просто установлен, а также загружен в ядро ОС как модуль и готов к работе. Без этого система хранения работать не сможет – даже если все пакеты формально установлены.

Если строки `drbd` нет: DRBD не загружен – либо из-за ошибки, либо несовместимости. Нужно выполнить установочную команду `trok-drbd-autoinstall` от имени суперпользователя или разбираться с ошибкой ядра.

Для подтверждения успешной установки компонентов кластера TROK рекомендуется выполнить команду на узлах контроллера:

```
sudo systemctl status trok-auth.service trok-controller.service trok-cp-
endpoint.service trok-worker.service
```

Если на узле контроллера вы не устанавливали компонент `trok-worker`, удалите этот аргумент из запроса:

```
sudo systemctl status trok-auth.service trok-controller.service trok-cp-
endpoint.service
```

Для узлов хранения (типа `worker`) команда будет иметь меньшее количество аргументов:

```
sudo systemctl status trok-worker.service
```

Все основные компоненты (аутентификация, контроллер, `endpoint`, `worker`) должны запускаться как сервисы. Если сервис не стартовал/имел ошибку на запуске – система хранения не будет работать, как ожидается.

Если сервисы имеют статус `active (running)`, нет ошибок типа `failed`, `inactive` и т.п., результат удачный.

Отрицательный сценарий: ошибки загрузки, сервисы в состоянии `failed`. Требуется диагностика: изучите логи, проверьте шаги установки/настройки.

```

root@astra183-gui-qa:~# systemctl status trok-auth.service trok-controller.service trok-cp-endpoint.serv
ice trok-worker.service
● trok-auth.service - trok-auth authentication service for TROK SDS
   Loaded: loaded (/lib/systemd/system/trok-auth.service; enabled; preset: enabled)
   Active: active (running) since Thu 2025-12-18 13:41:44 MSK; 24min ago
     Main PID: 1212 (trok-auth)
        Tasks: 7 (limit: 4613)
      Memory: 14.0M
         CPU: 16ms
       CGroup: /system.slice/trok-auth.service
               └─1212 /usr/sbin/trok-auth -config /etc/trok-auth/config.yaml -log /var/log/trok-auth/trok>
дек 18 13:41:44 astra183-gui-qa systemd[1]: Started trok-auth.service - trok-auth authentication servic>

● trok-controller.service - trok-controller service for TROK SDS
   Loaded: loaded (/lib/systemd/system/trok-controller.service; enabled; preset: enabled)
   Active: active (running) since Thu 2025-12-18 13:41:44 MSK; 24min ago
     Main PID: 1214 (trok-controller)
        Tasks: 9 (limit: 4613)
      Memory: 19.3M
         CPU: 1.038s
       CGroup: /system.slice/trok-controller.service
               └─1214 /usr/sbin/trok-controller -config /etc/trok-controller/config.yaml -log /var/log/tr>
дек 18 13:41:44 astra183-gui-qa systemd[1]: Started trok-controller.service - trok-controller service f>

● trok-cp-endpoint.service - TROK control plane API entryptpoint service
   Loaded: loaded (/lib/systemd/system/trok-cp-endpoint.service; enabled; preset: enabled)
   Active: active (running) since Thu 2025-12-18 13:41:43 MSK; 24min ago
   Process: 1153 ExecStartPre=/usr/bin/install -o trok -g trok -m 0640 /dev/null /var/log/trok-cp-endp>
     Main PID: 1171 (trok-cp-endpoint)
        Tasks: 7 (limit: 4613)
      Memory: 19.7M
         CPU: 24ms
       CGroup: /system.slice/trok-cp-endpoint.service
               └─1171 /usr/sbin/trok-cp-endpoint -config /etc/trok-cp-endpoint/config.yaml -log /var/log/>
дек 18 13:41:43 astra183-gui-qa systemd[1]: Starting trok-cp-endpoint.service - TROK control plane API >
дек 18 13:41:43 astra183-gui-qa systemd[1]: Started trok-cp-endpoint.service - TROK control plane API e>

● trok-worker.service - TROK worker service for DRBD managment
   Loaded: loaded (/lib/systemd/system/trok-worker.service; enabled; preset: enabled)
   Active: active (running) since Thu 2025-12-18 13:41:43 MSK; 24min ago
   Process: 1154 ExecStartPre=/usr/bin/install -o trok -g trok -m 0640 /dev/null /var/log/trok-worker/>
     Main PID: 1173 (trok-worker)
        Tasks: 11 (limit: 4613)
      Memory: 18.1M
         CPU: 942ms
       CGroup: /system.slice/trok-worker.service

```

Рисунок 14 – Результат вывода команды проверки статуса установленных служб
(комбинированный тип узла)

12. ОСОБЕННОСТИ ПЕРЕУСТАНОВКИ КОМПОНЕНТОВ

Если при переустановке пакета `trok-webui` возникают ошибки (например, конфликты конфигураций или зависимостей), следуйте этой последовательности команд для полной очистки и повторной установки:

Удалите пакет с полным очищением:

```
sudo apt remove --purge trok-webui
```

Команда `remove --purge` удаляет пакет и все его конфигурационные файлы, а не только исполняемые файлы. Такой способ удаления предотвращает конфликты при повторной установке.

Удалите ненужные зависимости:

```
sudo apt autoremove
```

Это полезная команда для регулярного обслуживания системы, так как она эффективно очищает неиспользуемые файлы любого программного обеспечения и поддерживает оптимальную производительность.

Переустановите пакет:

```
sudo apt install trok-webui
```

Команды следует выполнять с правами администратора (`sudo`).

13. ЛОГИРОВАНИЕ

Логи сохраняются в директорию `/var/log/trok-installer` (или в указанную через `--log`). Имя файла лога генерируется автоматически на основе текущей даты и времени.

Пример:

`/var/log/trok-installer/install_1750759134.log`

Система осуществляет автоматическую регистрацию событий в соответствии со следующей схемой:

- Хранилище журналов
 - Основной каталог: `/var/log/trok-installer/`
 - Альтернативный путь: определяется параметром `--log` при запуске (подробности в пункте 5)
- Генерация имен файлов

Формируется по шаблону: `install_<timestamp>.log`

где: `<timestamp>` – это количество секунд, прошедших с начала эпохи Unix (01.01.1970). Например, `install_1750759134.log`

14. ПОДГОТОВКА РЕСУРСОВ К ОБНОВЛЕНИЮ ПО

Перед началом обновления рекомендуется убедиться, что все DRBD-ресурсы находятся в актуальном состоянии. Это необходимо для предотвращения потери данных и корректной работы кластерной системы.

Инструкция предназначена для программного обеспечения версии 1.2; обновление возможно с версии 1.0 и выше.

Выполнение всех команд, приведённых в руководстве, требует прав суперпользователя в операционной системе.

Проверьте состояние DRBD-ресурсов на каждом узле:

```
sudo drbdadm status
```

В результате выполнения команды убедитесь, что все используемые ресурсы имеют статус UpToDate: данные на принимающем и передающем узле полностью синхронизированы и готовы к дальнейшей работе.

15. РЕЗЕРВНОЕ КОПИРОВАНИЕ КОНФИГУРАЦИЙ СЕРВИСОВ ПЕРЕД ОБНОВЛЕНИЕМ

Перед началом процедуры обновления рекомендуется создать резервную копию всех важных конфигурационных файлов и рабочих данных сервисов. Данная мера обеспечит возможность оперативного восстановления системы в случае возникновения внештатных ситуаций.

Создайте каталог для хранения резервных копий:

```
sudo mkdir -p /backup
```

Перед началом архивирования рекомендуется убедиться, что все необходимые директории существуют. В противном случае архив может быть создан неполным или с ошибками.

Перед непосредственным запуском процесса резервного копирования рекомендуется проверить, достаточно ли свободного места на целевом диске или разделе для хранения архивных файлов. Для этого используйте команду:

```
df -h /backup
```

Обратите внимание на значение в столбце «Свободно» («Available»). Необходимо удостовериться, что объём свободного пространства превышает ожидаемый размер резервной копии с учётом возможных временных файлов, которые могут быть созданы в процессе копирования.

Выполните архивирование всех необходимых директорий:

```
sudo tar -cvzf /backup/trok-$(date +%Y%m%d-%H%M%S).tar.gz \  
  /etc/trok /etc/trok-auth /etc/trok-controller \  
  /etc/trok-cp-endpoint /etc/trok-worker /etc/drbd.d \  
  /var/lib/trok /var/lib/trok-auth /var/lib/trok-controller \  
  /var/lib/trok-cp-endpoint /var/lib/trok-worker
```

В данном архиве будут сохранены все основные конфигурационные файлы, а также рабочие данные сервисов, задействованных в инфраструктуре.

Проверьте наличие и целостность созданного архивного файла:

```
sudo ls -lh /backup/trok-*.tar.gz
```

Вывод команды должен содержать строку с информацией о свежесозданном архиве, аналогично следующему примеру:

```
-rw-r--r-- 1 root root 16K map 26 16:02 /backup/trok-2026-03-26.tar.gz
```

Убедитесь, что архив успешно создан и имеет ожидаемый размер. В случае возникновения ошибок проверьте права доступа к исходным данным и повторите попытку создания резервной копии.

Примечание. Архив с резервной копией рекомендуется переместить на внешний носитель или в отдельное защищённое хранилище. Это позволит восстановить систему даже при повреждении основного сервера.

16. РЕЗЕРВНОЕ КОПИРОВАНИЕ ДАННЫХ ТОМОВ DRBD-РЕСУРСОВ ПЕРЕД ОБНОВЛЕНИЕМ

Резервное копирование данных томов DRBD-ресурсов – это последовательность действия, включающая подготовку ресурса, создание образа и последующее восстановление.

Перед началом резервного копирования переведите DRBD-ресурс в состояние Primary на том узле, где планируется выполнять создание копии. Выполните команду:

```
sudo drbdadm primary <имя_ресурса>
```

Убедитесь, что ресурс на остальных узлах находится в состоянии Secondary. Проверьте статус с помощью команды:

```
sudo drbdadm status <имя_ресурса>
```

Перед созданием копии рекомендуется остановить все сервисы, использующие данный DRBD-раздел, и предварительно размонтировать файловую систему. Этот шаг позволяет сохранить целостность данных и предотвратить возможные ошибки при копировании.

Выполните команду для побитового копирования содержимого DRBD-устройства:

```
sudo dd if=/dev/drbd/by-res/<имя_ресурса>/<id_тома> of=/путь/к/резервной/копии/backup_drbd<имя_ресурса><id_тома>.img bs=1M status=progress
```

Если в ресурсе используется несколько томов, повторите процедуру для каждого из них, указав соответствующий идентификатор тома.

Внимание:

- Ошибка в параметре of (например, указание пути к существующему устройству или разделу вместо файла резервной копии) приведёт к

необратимой потере данных на целевом устройстве. Перед запуском команды рекомендуется внимательно проверить указанный путь.

- Ориентировочное время выполнения операции будет зависеть от размера тома и производительности используемых дисков. При оценке времени копирования используйте примерную скорость перемещения данных, характерную для вашей инфраструктуры (например, для SATA SSD — до 500 МБ/с, для HDD — 100–200 МБ/с).
- Обратите внимание, что размер резервной копии будет эквивалентен размеру исходного DRBD-тома. Убедитесь, что на целевом диске или разделе достаточно свободного пространства для сохранения образа.
- Во время выполнения команды `dd` возможно временное снижение производительности системы в связи с высокой нагрузкой на дисковую подсистему. Рекомендуется выполнять данный процесс в период наименьшей пользовательской активности.

Сохраните резервную копию на отдельный надёжный носитель или перенесите её на другой узел. Это обеспечит безопасность данных при обновлении текущего узла.

Оставьте DRBD-устройство в состоянии Primary и при возможности размонтируйте том или смонтируйте только для чтения.

Создайте контрольную сумму образа. Выполните команду для создания контрольной суммы резервной копии:

```
sha256sum /путь/к/резервной/копии/backup_drbd_<имя_ресурса><id_тома>.img >
/путь/к/резервной/копии/backup_drbd_<имя_ресурса><id_тома>.img.sha256sum
```

Если после обновления программного обеспечения все копии DRBD-ресурса были утеряны, восстановите состояние томов из резервной копии. Создайте новый DRBD-ресурс с такими же или большими размерами томов, чтобы восстановление прошло успешно.

Проверьте контрольные суммы (например, с помощью `sha256sum`) для подтверждения целостности резервной копии.

Переведите созданный ресурс в состояние Primary:

```
sudo drbdadm primary <новое_имя_ресурса>
```

Восстановите данные из резервной копии с помощью команды:

```
sudo dd if=backup_drbd_<имя_ресурса><id_тома>.img of=/dev/drbd/by-res/<новое_имя_ресурса>/<id_тома> bs=1M status=progress
```

Обратите внимание: восстановление через dd перезапишет все данные на целевом устройстве. Проверьте правильность пути назначения, чтобы не потерять актуальные данные.

17. ОБНОВЛЕНИЕ ПАКЕТОВ ИЗ РЕПОЗИТОРИЯ

Процедура обновления узлов в отказоустойчивом кластере требует строгого соблюдения порядка действий для предотвращения простоев и обеспечения целостности данных. Обновление всех узлов одновременно не допускается: операция выполняется строго поочерёдно, с чётким разграничением ролей между узлами (Primary и Secondary). Нарушение этого порядка может привести к потере данных либо нарушению работоспособности кластера.

Для гарантии получения последних версий программного обеспечения из доверенного источника, перед выполнением обновления программного обеспечения убедитесь, что на сервере корректно подключён актуальный локальный apt-репозиторий TROK.

Проверьте подключение apt-репозитория с помощью команды:

```
sudo cat /etc/apt/sources.list.d/trok-sds.list
```

В выводе команды должна присутствовать строка следующего вида:

```
deb [arch=amd64 signed-by=/etc/apt/keyrings/trok-archive-keyring.gpg]  
file:///opt/trok/sds/apt-repo stable main
```

Запись подтверждает подключение корректного локального источника пакетов и наличие файла ключа для проверки подписей пакетов.

Альтернативная команда для проверки:

```
grep -r "trok" /etc/apt/sources.list.d/
```

Для получения актуальной информации о доступных версиях пакетов в подключённых репозиториях выполните обновление индекса пакетов:

```
sudo apt update
```

Убедитесь в отсутствии ошибок при обновлении списка пакетов. В противном случае проверьте наличие и содержимое репозитория согласно предыдущему пункту.

Для контроля версии и наличия установленных компонентов TROK в системе выполните поиск установленных пакетов с помощью фильтра:

```
sudo apt list --installed | grep -i trok
```

В результате выполнения команды будет предоставлен перечень установленных пакетов TROK и их текущие версии:

```
python3-trok/stable,now 1.0-1 all [установлен]
trok-auth/stable,now 1.2 amd64 [установлен, автоматически]
trok-controller-meta/stable,now 1.1 all [установлен]
trok-controller/stable,now 1.2 amd64 [установлен, автоматически]
trok-cp-endpoint/stable,now 1.2 amd64 [установлен, автоматически]
trok-webui/stable,now 1.2 all [установлен]
trok-worker-meta/stable,now 1.1 all [установлен]
trok-worker/stable,now 1.2 amd64 [установлен, автоматически]
```

На основании полученных данных сформируйте список пакетов, требующих обновления.

Перед началом процедуры обновления выполните следующие шаги:

Проверьте состояние сервиса, чтобы убедиться, что он запущен на ноде и перед обновлением его необходимо будет остановить. Для этого выполните следующую команду:

```
sudo systemctl list-units --type=service --state=running | grep trok
```

Остановите сервисы TROK:

```
sudo systemctl stop trok-worker.service
sudo systemctl stop trok-controller.service
sudo systemctl stop trok-cp-endpoint.service
sudo systemctl stop trok-auth.service
```

Для обновления всех обнаруженных пакетов TROK в системе выполните команду для каждого пакета поочерёдно, либо укажите несколько через пробел:

```
sudo apt install --only-upgrade <имя_пакета>
```

Пример для нескольких пакетов:

```
sudo apt install --only-upgrade trok-auth trok-controller trok-cp-endpoint trok-webui trok-worker
```

Дождитесь завершения процесса установки обновлений и убедитесь в отсутствии ошибок.

После завершения обновления всех компонентов TROK необходимо проверить статус обновленных сервисов:

```
sudo systemctl status trok-<название_сервиса>
```

А также перезапустить сервисы для применения изменений:

```
sudo systemctl restart <название_сервиса>
```

Если сервис остановлен, запустите его:

```
sudo systemctl start <название_сервиса>
```

Проверьте состояние DRBD-ресурсов для подтверждения корректной работы кластера:

```
sudo drbdadm status
```

Убедитесь, что все ресурсы находятся в состоянии UpToDate и продолжают функционировать штатно.

Примечание. После выполнения операций по обновлению рекомендуется провести дополнительную проверку работоспособности всех компонентов системы TROK и сопутствующих сервисов.

18. ОБНОВЛЕНИЕ HA-КЛАСТЕРА TROK

18.1. Перед началом обновления

Инструкция предназначена для работающего HA-кластера TROK с несколькими HA controller-узлами и одним или несколькими workers.

Обновление можно выполнить одним из двух способов:

- через Ansible collection из нового ISO — рекомендуемый способ;
- вручную через APT — резервный способ.

Перед началом убедитесь, что:

- новый ISO доступен на управляющем компьютере;
- кластер работает без ошибок;
- все workers отображаются в WebUI в состоянии ONLINE;
- сохранён inventory, который использовался для установки кластера;
- в inventory присутствуют группы [controllers], [workers] и [ha_controllers];
- в существующих group variables сохранены ha_controller_vip и имя HA-ресурса;
- на всех HA controllers работает drbd-reactor;
- в кластере существует ровно один DRBD Primary;
- только на Primary находятся VIP, обе БД и работающие control-plane-сервисы;
- согласовано окно недоступности;
- коллекция внутри нового ISO содержит правильные настройки источника пакетов;
- новый выпуск использует Apache для работы WebUI.

Инструкция обновляет только прикладные пакеты TROK. Она не обновляет операционную систему, ядро, DRBD, drbd-reactor, gateway и остальные системные пакеты.

Не выполняйте одновременно другие работы с кластером.

Не используйте для обновления TROK команды:

```
apt upgrade
apt full-upgrade
apt dist-upgrade
```

18.1.1. Проверка состояния НА

Узнайте имя НА-ресурса из сохранённого group_vars/all.yml:

```
ha_db_resource_name: "HA-TROK-DB"
```

Дальнейшие примеры используют стандартное имя HA-TROK-DB. Если в inventory указано другое имя, замените его в командах.

На каждом НА controller выполните:

```
systemctl is-active drbd-reactor.service
drbdadm role HA-TROK-DB
```

drbd-reactor.service должен иметь состояние active на всех НА controllers.

Ровно один узел должен вернуть роль Primary. Остальные узлы должны вернуть роль Secondary.

На Primary выполните:

```
findmnt -rn --mountpoint /var/lib/trok-controller
findmnt -rn --mountpoint /var/lib/trok-auth
ip -brief address
systemctl is-active \
  trok-auth.service \
  trok-controller.service \
  trok-cp-endpoint.service

systemctl is-active \
  nginx.service \
  apache2.service 2>/dev/null || true
```

На Primary должны быть:

- оба mount points;
- существующий ha_controller_vip;
- состояние active у controller, auth и endpoint;
- состояние active ровно у одного WebUI-сервиса: nginx или Apache.

Если исходный выпуск ещё использует nginx, вместо apache2.service активным WebUI-сервисом будет nginx.service.

На каждом Secondary выполните:

```
findmnt -rn --mountpoint /var/lib/trok-controller || true
findmnt -rn --mountpoint /var/lib/trok-auth || true
ip -brief address
systemctl is-active \
    trok-auth.service \
    trok-controller.service \
    trok-cp-endpoint.service \
    nginx.service \
    apache2.service 2>/dev/null || true
```

На Secondary:

- обе команды findmnt не должны выводить mount points;
- VIP должен отсутствовать;
- control-plane- и WebUI-сервисы должны иметь состояние inactive.

Если Primary отсутствует, Primary больше одного или владение БД, VIP и сервисами распределено между разными узлами, обновление не начинайте.

18.2. Создание резервной копии

Резервная копия создаётся только на текущем Primary до остановки drbd-reactor.

Подключитесь к Primary и перейдите в режим root:

```
sudo -i
```

Проверьте наличие sqlite3:

```
command -v sqlite3
```

Если команда не найдена, установите пакет:

```
apt-get update
apt-get install --no-install-recommends sqlite3
```

Убедитесь, что каталог предыдущей резервной копии отсутствует:

```
test ! -e /var/backups/trok-before-upgrade
```

Если каталог существует, не удаляйте и не перезаписывайте его. Переместите предыдущую копию во внешнее хранилище и повторите проверку.

Создайте новый каталог:

```
install -d -o root -g root -m 0700 /var/backups/trok-before-upgrade
```

Создайте резервные копии БД controller и auth:

```
sqlite3 /var/lib/trok-controller/data.db \
    ".backup '/var/backups/trok-before-upgrade/controller.db'"

sqlite3 /var/lib/trok-auth/data.db \
    ".backup '/var/backups/trok-before-upgrade/auth.db'"
```

Проверьте созданные копии:

```
sqlite3 /var/backups/trok-before-upgrade/controller.db \
    'PRAGMA integrity_check;'

sqlite3 /var/backups/trok-before-upgrade/auth.db \
    'PRAGMA integrity_check;'
```

Обе команды должны вернуть:

```
ok
```

Скопируйте каталог /var/backups/trok-before-upgrade на внешний носитель или другой сервер.

Если хотя бы одна проверка не вернула ok, обновление не начинайте.

18.3. Вариант А — обновление через Ansible

18.3.1. Установка коллекции из нового ISO

На управляющем компьютере создайте точку монтирования:

```
sudo mkdir -p /mnt/trok-iso
```

Смонтируйте новый ISO, заменив путь на фактический:

```
sudo mount -o loop,ro /путь/к/TROK-new.iso /mnt/trok-iso
```

Проверьте архив коллекции:

```
find /mnt/trok-iso/ansible-collection \
  -maxdepth 1 \
  -type f \
  -name 'astra-trok-*.tar.gz'
```

Команда должна показать ровно один архив.

Установите коллекцию от имени пользователя, под которым запускается Ansible:

```
ansible-galaxy collection install \
  /mnt/trok-iso/ansible-collection/astra-trok-*.tar.gz \
  --force \
  --no-deps
```

Проверьте установленную коллекцию:

```
ansible-galaxy collection list astra.trok
```

18.3.2. Проверка inventory

Используйте существующий inventory.ini от исходного развёртывания.

В нём должны присутствовать три группы:

```
[controllers]
controller-1 ansible_host=<IP-controller-1>

[workers]
controller-1 ansible_host=<IP-controller-1>
controller-2 ansible_host=<IP-controller-2>
controller-3 ansible_host=<IP-controller-3>

[ha_controllers]
controller-1 ansible_host=<IP-controller-1>
controller-2 ansible_host=<IP-controller-2>
controller-3 ansible_host=<IP-controller-3>
```

Состав групп не изменяйте. Один узел может одновременно входить в несколько групп.

Проверьте подключение:

```
ansible -i inventory.ini controllers:workers:ha_controllers -m ping --become
```

Все узлы должны вернуть SUCCESS.

Дополнительные ISO-переменные создавать или изменять не требуется. Встроенная в новый ISO коллекция уже должна содержать правильные настройки выпуска. Существующие `ha_controller_vip` и имя HA-ресурса должны оставаться в `inventory`.

18.3.3. Запуск обновления

Запустите обновление:

```
ansible-playbook -i inventory.ini astra.trok.upgrade
```

Ansible последовательно:

- проверит текущее HA-владение;
- остановит и обновит `workers`;
- остановит HA control plane через `drbd-reactor`;
- обновит `controller`-пакеты на всех HA controllers;
- заменит `nginx` на `Apache` в конфигурации `reactor`;
- снова запустит `drbd-reactor`;
- проверит единственного владельца БД, VIP и сервисов;
- запустит `workers` и дождётся состояния `ONLINE`.

Дождитесь завершения `playbook` без ошибок.

После завершения размонтируйте ISO:

```
sudo umount /mnt/trok-iso
```

Перейдите к разделу 18.5 для проверки результата.

Если `playbook` завершился с ошибкой, не запускайте его повторно. Перейдите к разделу 18.6.

18.4. Вариант Б — ручное обновление через APT

Ручной способ используется только тогда, когда обновление через Ansible выполнить невозможно.

ISO необходимо скопировать на каждый уникальный узел из групп `[workers]` и `[ha_controllers]`, например в `/root/TROK-new.iso`.

Если один узел входит в обе группы, подготовьте репозиторий на нём только один раз.

Все команды этого раздела выполняются от root на соответствующем узле.

18.4.1. Подключение репозитория из ISO

Повторите этот раздел на каждом worker и HA controller.

Проверьте, что каталоги от предыдущей попытки отсутствуют:

```
test ! -e /opt/trok/sds/apt-repo-update
test ! -e /etc/apt/sources.list.d/trok-sds.list.before-upgrade
```

Если один из путей существует, остановитесь. Не удаляйте старые файлы самостоятельно.

Создайте каталоги:

```
mkdir -p /mnt/trok-iso
mkdir -p /opt/trok/sds/apt-repo-update
mkdir -p /etc/apt/keyrings
```

Смонтируйте ISO:

```
mount -o loop,ro /root/TROK-new.iso /mnt/trok-iso
```

Проверьте архив репозитория:

```
find /mnt/trok-iso/repository \
  -maxdepth 1 \
  -type f \
  -name '*.tar.gz'
```

Команда должна показать ровно один архив.

Распакуйте репозиторий:

```
tar -xzf /mnt/trok-iso/repository/*.tar.gz \
  --strip-components=3 \
  -C /opt/trok/sds/apt-repo-update
```

Проверьте результат:

```
test -f /opt/trok/sds/apt-repo-update/dists/stable/Release
```

Установите ключ репозитория:

```
install -o root -g root -m 0644 \
  /mnt/trok-iso/repository/trok-archive-keyring.gpg \
  /etc/apt/keyrings/trok-archive-keyring.gpg
```

Сохраните старый источник пакетов:

```
cp -a \
  /etc/apt/sources.list.d/trok_repo.list \
  /etc/apt/sources.list.d/trok_repo.list.before-upgrade
```

Подключите новый репозиторий:

```
printf 'deb [arch=%s signed-by=/etc/apt/keyrings/trok-archive-keyring.gpg]
file:///opt/trok/sds/apt-repo-update stable main\n' \
  "$(dpkg --print-architecture)" \
  >/etc/apt/sources.list.d/trok_repo.list
```

Обновите список пакетов и размонтируйте ISO:

```
apt-get update && umount /mnt/trok-iso
```

18.4.2. Остановка и обновление workers

Сначала на всех workers остановите сервис и установите runtime mask:

```
systemctl stop trok-worker.service

systemctl mask --runtime trok-worker.service

systemctl is-active trok-worker.service
```

Последняя команда должна вернуть inactive.

Завершите остановку всех workers до обновления пакетов хотя бы на одном из них.

Затем на каждом worker одним блоком сформируйте список пакетов, создайте временный запрет и запустите АРТ:

```
worker_packages=(trok-worker python3-trok)

for package in trok-worker-meta trok-combined-meta; do
  if dpkg-query -W -f='${db:Status-Abbrev}' "$package" 2>/dev/null | grep -q
'^ii'; then
    worker_packages+=("$package")
  fi
done
```

```
done

if [ -e /usr/sbin/policy-rc.d ] || [ -L /usr/sbin/policy-rc.d ]; then
    echo 'Файл /usr/sbin/policy-rc.d уже существует. Обновление остановлено.' >&2
    false
else
    printf '%s\n' '#!/bin/sh' 'exit 101' >/usr/sbin/policy-rc.d
    chmod 0755 /usr/sbin/policy-rc.d
    apt-get install \
        --only-upgrade \
        --no-install-recommends \
        "${worker_packages[@]}" &&
    rm -f /usr/sbin/policy-rc.d
fi
```

Если существующий policy-rc.d обнаружен или APT завершился с ошибкой, не продолжайте ручное обновление. При ошибке APT временный файл останется на месте; удалите его только по разделу 18.6.

Не снимайте runtime masks и не запускайте workers.

Если обновление хотя бы одного worker завершилось с ошибкой, не останавливайте работающий HA control plane. Перейдите к разделу 18.6.

18.4.3. Остановка HA control plane

Непосредственно перед остановкой ещё раз выполните на всех HA controllers:

```
drbdadm role HA-TROK-DB
```

Ровно один узел должен быть Primary. Запишите его имя.

Сначала на каждом Secondary остановите reactor:

```
systemctl stop drbd-reactor.service
systemctl is-active drbd-reactor.service
```

Последняя команда должна вернуть inactive.

Убедитесь, что reactor остановлен на всех Secondary. Только после этого остановите его на записанном Primary:

```
systemctl stop drbd-reactor.service
systemctl is-active drbd-reactor.service
```

Не останавливайте Primary первым: оставшийся Secondary может принять НА-владение во время процедуры.

После остановки reactor на всех HA controllers проверьте сервисы:

```
systemctl is-active \
  trok-auth.service \
  trok-controller.service \
  trok-cp-endpoint.service \
  nginx.service \
  apache2.service 2>/dev/null || true
```

Все сервисы должны иметь состояние inactive.

На каждом HA controller установите runtime masks:

```
systemctl mask --runtime \
  trok-auth.service \
  trok-controller.service \
  trok-cp-endpoint.service \
  nginx.service \
  apache2.service
```

18.4.4. Обновление controller-пакетов

Повторите этот раздел на каждом HA controller. Не запускайте reactor, пока обновление и проверка не будут завершены на всех узлах.

Если nginx установлен, удалите его без purge и autoremove:

```
if dpkg-query -W -f='${db:Status-Abbrev}' nginx 2>/dev/null | grep -q '^ii'; then
  if [ -e /usr/sbin/policy-rc.d ] || [ -L /usr/sbin/policy-rc.d ]; then
    echo 'Файл /usr/sbin/policy-rc.d уже существует. Обновление остановлено.' >&2
    false
  else
    printf '%s\n' '#!/bin/sh' 'exit 101' >/usr/sbin/policy-rc.d
    chmod 0755 /usr/sbin/policy-rc.d
    apt-get remove --no-auto-remove nginx &&
    rm -f /usr/sbin/policy-rc.d
  fi
fi
```

Если АРТ завершился с ошибкой, не продолжайте раздел. Перейдите к разделу 18.6.

Одним блоком сформируйте список controller-пакетов, создайте временный запрет и запустите АРТ:

```

controller_packages=(
    trok-controller
    trok-auth
    trok-cp-endpoint
    trok-webui
    python3-trok
)

for package in trok-controller-meta trok-combined-meta; do
    if dpkg-query -W -f='${db:Status-Abbrev}' "$package" 2>/dev/null | grep -q
'^ii'; then
        controller_packages+=("$package")
    fi
done

if [ -e /usr/sbin/policy-rc.d ] || [ -L /usr/sbin/policy-rc.d ]; then
    echo 'Файл /usr/sbin/policy-rc.d уже существует. Обновление остановлено.' >&2
    false
else
    printf '%s\n' '#!/bin/sh' 'exit 101' >/usr/sbin/policy-rc.d
    chmod 0755 /usr/sbin/policy-rc.d
    apt-get install \
        --only-upgrade \
        --no-install-recommends \
        "${controller_packages[@]}" &&
    rm -f /usr/sbin/policy-rc.d
fi

```

Если существующий policy-rc.d обнаружен или АРТ завершился с ошибкой, не продолжайте раздел. При ошибке АРТ удалите временный файл только по разделу 18.6.

Проверьте конфигурацию Apache:

```
apache2ctl configtest
```

Команда должна вернуть:

```
Syntax OK
```

18.4.5. Переключение HA WebUI на Apache

Повторите этот раздел на каждом HA controller.

Примеры используют стандартный файл /etc/drbd-reactor.d/HA-TROK-DB.toml. Если имя HA-ресурса изменено в inventory, используйте соответствующее имя файла.

Проверьте текущий WebUI-сервис:

```
grep -E '"(nginx|apache2)\.service"' \
/etc/drbd-reactor.d/HA-TROK-DB.toml
```

Команда должна показать ровно одну строку: с nginx.service или с apache2.service.

Замените nginx на Apache:

```
sed -i 's/"nginx\.service"/"apache2.service"/' \
/etc/drbd-reactor.d/HA-TROK-DB.toml
```

Проверьте результат:

```
grep -q '"apache2.service"' \
/etc/drbd-reactor.d/HA-TROK-DB.toml

! grep -q '"nginx.service"' \
/etc/drbd-reactor.d/HA-TROK-DB.toml
```

Если проверка неуспешна, reactor не запускайте.

18.4.6. Запуск HA control plane

Переходите к запуску только после успешного обновления и проверки всех HA controllers.

Сначала снимите runtime masks на каждом HA controller:

```
systemctl unmask --runtime \
trok-auth.service \
trok-controller.service \
trok-cp-endpoint.service \
nginx.service \
apache2.service
```

Затем на каждом HA controller запретите прямой запуск HA-сервисов при загрузке:

```
systemctl disable \
trok-auth.service \
trok-controller.service \
trok-cp-endpoint.service \
apache2.service
```

Не запускайте эти сервисы командами systemctl start. В HA-кластере ими управляет только drbd-reactor.

После снятия masks на всех HA controllers запустите reactor на каждом из них:

```
systemctl enable drbd-reactor.service  
systemctl start drbd-reactor.service  
systemctl is-active drbd-reactor.service
```

Последняя команда должна вернуть active.

Дождитесь появления ровно одного Primary и проверьте WebUI по существующему VIP.

Не изменяйте роль DRBD вручную, не монтируйте БД и не назначайте VIP вручную.

Если единственный согласованный HA-владелец не появился или WebUI не открылся, workers не запускайте. Перейдите к разделу 18.6.

18.4.7. Запуск workers

Только после успешной проверки HA control plane выполните на каждом worker:

```
systemctl unmask --runtime trok-worker.service  
systemctl start trok-worker.service  
systemctl is-active trok-worker.service
```

Последняя команда должна вернуть:

```
active
```

18.5. Проверка результата

18.5.1. Проверка HA controllers

На каждом HA controller выполните:

```
systemctl is-active drbd-reactor.service  
drbdadm role HA-TROK-DB
```

drbd-reactor.service должен быть active на всех узлах. Ровно один узел должен иметь роль Primary, остальные — Secondary.

На Primary выполните:

```
findmnt -rn --mountpoint /var/lib/trok-controller
findmnt -rn --mountpoint /var/lib/trok-auth
ip -brief address
systemctl is-active \
    trok-auth.service \
    trok-controller.service \
    trok-cp-endpoint.service \
    apache2.service
```

На Primary должны одновременно находиться:

- обе смонтированные БД;
- существующий VIP;
- четыре сервиса в состоянии active.

На каждом Secondary выполните:

```
findmnt -rn --mountpoint /var/lib/trok-controller || true
findmnt -rn --mountpoint /var/lib/trok-auth || true
ip -brief address
systemctl is-active \
    trok-auth.service \
    trok-controller.service \
    trok-cp-endpoint.service \
    apache2.service 2>/dev/null || true
```

На Secondary:

- БД не должны быть смонтированы;
- VIP должен отсутствовать;
- четыре сервиса должны иметь состояние inactive.

На всех HA controllers проверьте запрет прямого запуска:

```
systemctl is-enabled \
    trok-auth.service \
    trok-controller.service \
    trok-cp-endpoint.service \
```

```
apache2.service
```

Все четыре сервиса должны иметь состояние disabled.

18.5.2. Проверка WebUI и workers

Откройте WebUI по существующему ha_controller_vip и проверьте:

- главная страница загружается без ошибки;
- вход в систему выполняется успешно;
- все workers отображаются в состоянии ONLINE.

На каждом worker выполните:

```
systemctl is-active trok-worker.service
```

Ожидаемый результат:

```
active
```

Обновление считается завершённым только после успешного выполнения всех перечисленных проверок.

Контролируемое переключение HA-владельца в эту упрощённую инструкцию не входит.

18.6. Действия при ошибке

При любой ошибке прекратите обновление.

Не выполняйте без отдельного решения администратора:

- повторный запуск playbook;
- повторную установку пакетов;
- прямой запуск controller/auth/endpoint/Apache;
- ручное изменение роли DRBD;
- ручное монтирование БД или назначение VIP;
- apt --fix-broken install;
- dpkg --configure -a;
- apt autoremove;
- downgrade пакетов;

- восстановление БД из backup.

Если ошибка произошла во время ручной АРТ-команды, удалите временный файл только в том случае, если он был создан этой инструкцией. Файл, существовавший до начала процедуры, удалять запрещено.

```
rm -f /usr/sbin/policy-rc.d
```

Если ошибка произошла при обновлении worker до остановки HA control plane:

- не останавливайте работающий reactor;
- оставьте workers остановленными и runtime-masked;
- передайте состояние на разбор.

Если reactor уже остановлен:

- не снимайте оставшиеся runtime masks;
- не запускайте reactor или отдельные HA-сервисы;
- передайте состояние на разбор.

Если ошибка произошла после снятия masks при попытке запуска reactor, на всех HA controllers остановите reactor и восстановите runtime masks:

```
systemctl stop drbd-reactor.service

systemctl mask --runtime \
  trok-auth.service \
  trok-controller.service \
  trok-cp-endpoint.service \
  nginx.service \
  apache2.service
```

Workers не запускайте.

Сохраните:

- полный вывод Ansible или АРТ;
- имя нового ISO;
- узел и этап, на котором произошла ошибка;
- вывод `systemctl status drbd-reactor.service`;
- вывод `drbdadm role HA-TROK-DB` на всех HA controllers;
- состояния control-plane-сервисов;

- каталог `/var/backups/trok-before-upgrade`.

19. КРИТИЧЕСКИЕ ПРЕДУПРЕЖДЕНИЯ

Перед началом удаления обратите внимание на следующие особенности:

- Порядок удаления: нарушение последовательности (например, удаление пулов до ресурсов) приведет к ошибкам InUse.
- Тестирование: перед выполнением в продакшене протестируйте процедуру на стенде.
- Служебные данные: не удаляйте директории `/srv/ha/internal/` до полной остановки сервисов – это может вызвать ошибки NFS 1.

20. УДАЛЕНИЕ СТРУКТУРНЫХ КОМПОНЕНТОВ SDS

Удаление структурных компонентов SDS производится с использованием web-интерфейса TROK.

Очередность удаления компонентов SDS:

- 1 шаг – ресурсы;
- 2 шаг – шаблоны томов;
- 3 шаг – шаблоны ресурсов;
- 4 шаг – группы ресурсов;
- 5 шаг – пулы хранения;
- 6 шаг – узлы;

Более подробная информация указана в документе «TROK Руководство по работе с графическим интерфейсом».

21. ПОЛНОЕ УДАЛЕНИЕ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

21.1. Удаление пакетов формата deb

Для удаления установленных пакетов выполните команду:

Внимание! Следующая команда полностью удалит компоненты TROK и веб-сервер **nginx**. Если **nginx** использовался для других целей, не включайте его в команду удаления.

```
sudo apt purge trok-controller-meta trok-worker-meta trok-harbor-meta trok-cp-  
endpoint trok-auth trok-webui nginx
```

Затем удалите неиспользуемые зависимости:

```
sudo apt autoremove
```

Если необходимо удалить пакеты по отдельности, используйте команду:

```
sudo apt purge <имя_пакета>
```

где <имя_пакета> — название конкретного пакета для удаления.

Если необходимо удалить deb-пакеты по отдельности, следует учитывать, какие пакеты устанавливаются вместе с каждым компонентом системы. Полный перечень устанавливаемых пакетов представлен в примечаниях к выпуску (Release Notes) для каждой конкретной версии программного обеспечения.

21.2. Очистка LVM-томов и блочных устройств

Перед удалением LVM-томов и блочных устройств необходимо остановить службу trok-controller командой:

```
sudo systemctl stop trok-controller
```

На узле типа worker удалите описания ресурсов:

```
sudo rm -r /etc/drbd.d/*.res
```

Перезагрузите сервер:

```
sudo reboot
```

Проверьте имеющиеся группы томов:

```
sudo lvs
```

Команда выводит текущее состояние логических томов (LV), их VG, размер и состояние.

Удалите все логические тома в вашей группе томов:

```
sudo lvremove -y /dev/<имя_группы_томов>/*
```

Команда без запроса подтверждения удаляет все логические тома внутри группы томов. Процесс полностью и безвозвратно стирает тома и связанные данные. Может потребоваться предварительно отмонтировать тома и убедиться, что данные не нужны.

Для полной очистки содержимого блочного устройства (например, диска) применяется команда:

```
sudo dd if=/dev/zero of=/dev/sd<буква_тома> bs=1G status=progress
```

где /dev/sd<буква_тома> – имя целевого устройства. Эта операция перезаписывает устройство нулями, что гарантирует удаление всех данных. Используйте с осторожностью.