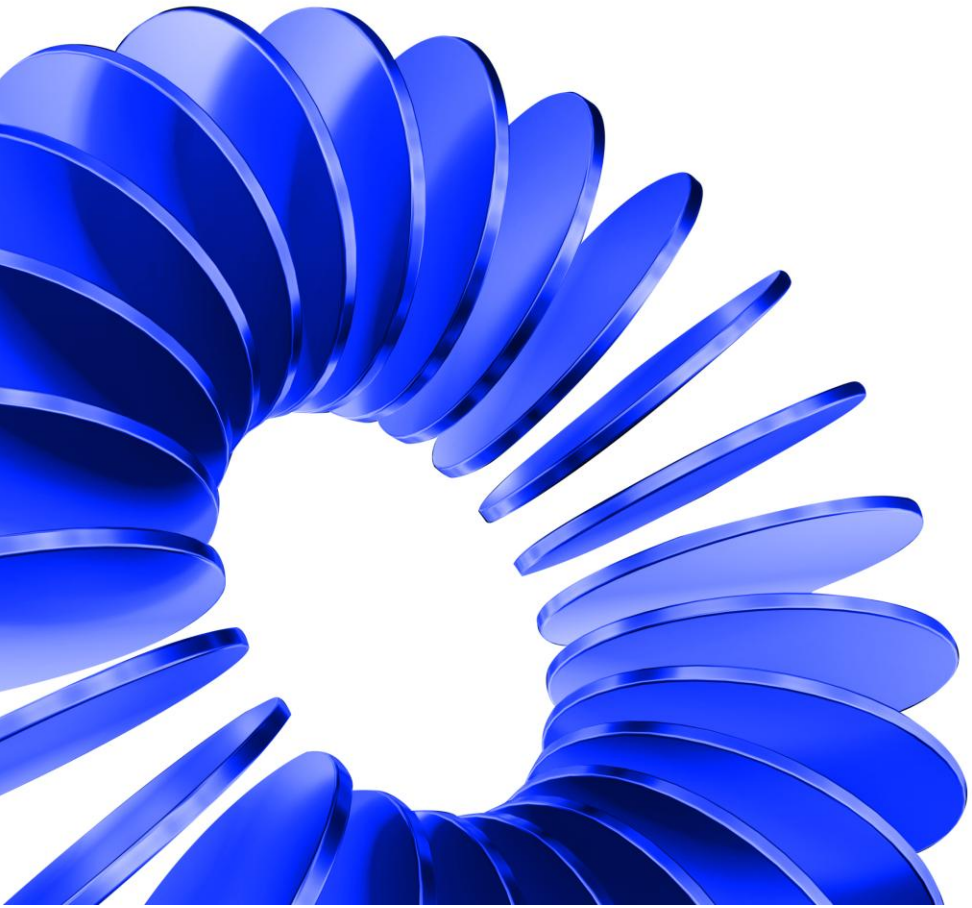
ОБЩЕСТВО С ОГРАНИЧЕННОЙ
ОТВЕТСТВЕННОСТЬЮ «РУСБИТЕХ-АСТРА»

СИСТЕМА ХРАНЕНИЯ ДАННЫХ TROK

РУКОВОДСТВО ПО НАСТРОЙКЕ



Москва, 2025г.

СОДЕРЖАНИЕ

1. ОБЩАЯ ИНФОРМАЦИЯ

Руководство по настройке иллюстрирует процесс создания кластерной системы хранения данных на примере конфигурации, состоящей из трёх узлов. Кластер представляют собой три физических сервера с установленной операционной системой Astra Linux. В кластере рекомендуется использовать не менее трёх серверов для обеспечения устойчивости к сбоям и достижения кворума при принятии решений. Подобная конфигурация минимизирует риск разделения сети и повышает общую отказоустойчивость системы. Все серверы подключены к одной локальной сети (с единой подсетью), обеспечивая сетевое взаимодействие внутри единой инфраструктуры. Для обеспечения избыточности и повышенной отказоустойчивости системы репликация данных осуществляется между тремя узлами.

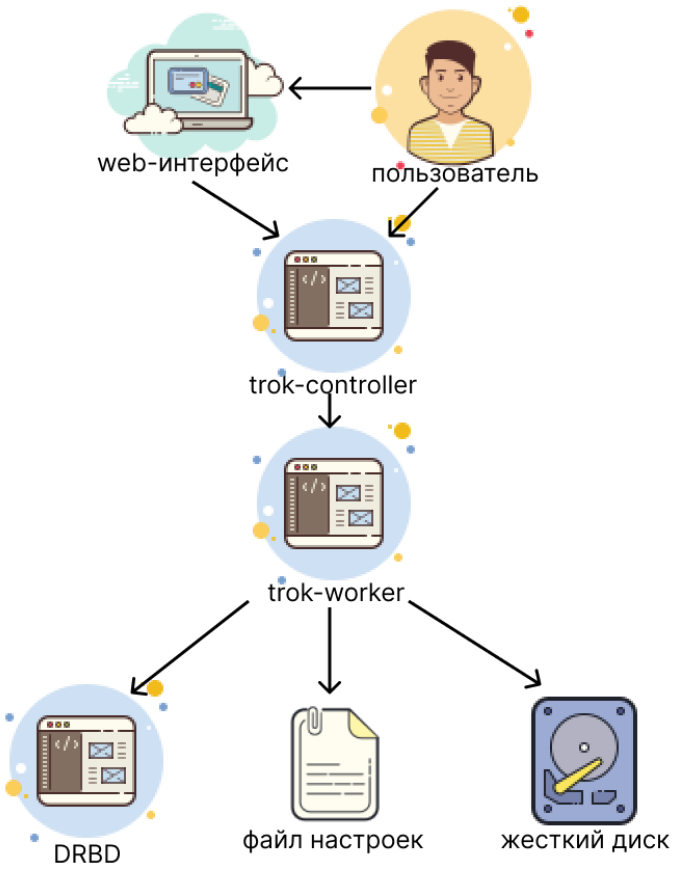


Рисунок 1 – Взаимодействие основных компонентов программного обеспечения

Техническая реализация программного обеспечения включает интеграцию четырех основных компонентов:

– trok-controller – центральный модуль, отвечающий за оркестрацию кластерных ресурсов и управление конфигурацией распределенного хранилища данных.

– trok-worker – служба, которая работает на каждом узле (сервере), где должны быть размещены реплицированные данные. Worker отвечает за управление локальными ресурсами хранения и взаимодействует с trok-controller.

– TROK WEB GUI – веб-интерфейс для интуитивного администрирования функций системы хранения данных.

– trok-harbor – готовый программный модуль, встроенный в контроллер. Он объединяет три ключевых элемента:

– drbd-reactor – событийно-ориентированная служба автоматизации для DRBD, которая в реальном времени отслеживает состояние узлов и ресурсов DRBD и автоматически выполняет заданные действия (сценарии агентов ресурсов) при изменении этих состояний, например, при переключении ролей Primary/Secondary в кластере, обеспечивая автоматическое управление ресурсами.

– resource-agents — это набор готовых сценариев (агентов OCF), которые используются в TROK для управления высокодоступными хранилищами на DRBD. Агенты позволяют автоматизировать контроль состояния ресурсов и обеспечивают отказоустойчивость сервисов доступа.

– сервисы доступа – поддержка протоколов: iSCSI, NFS, SMB.

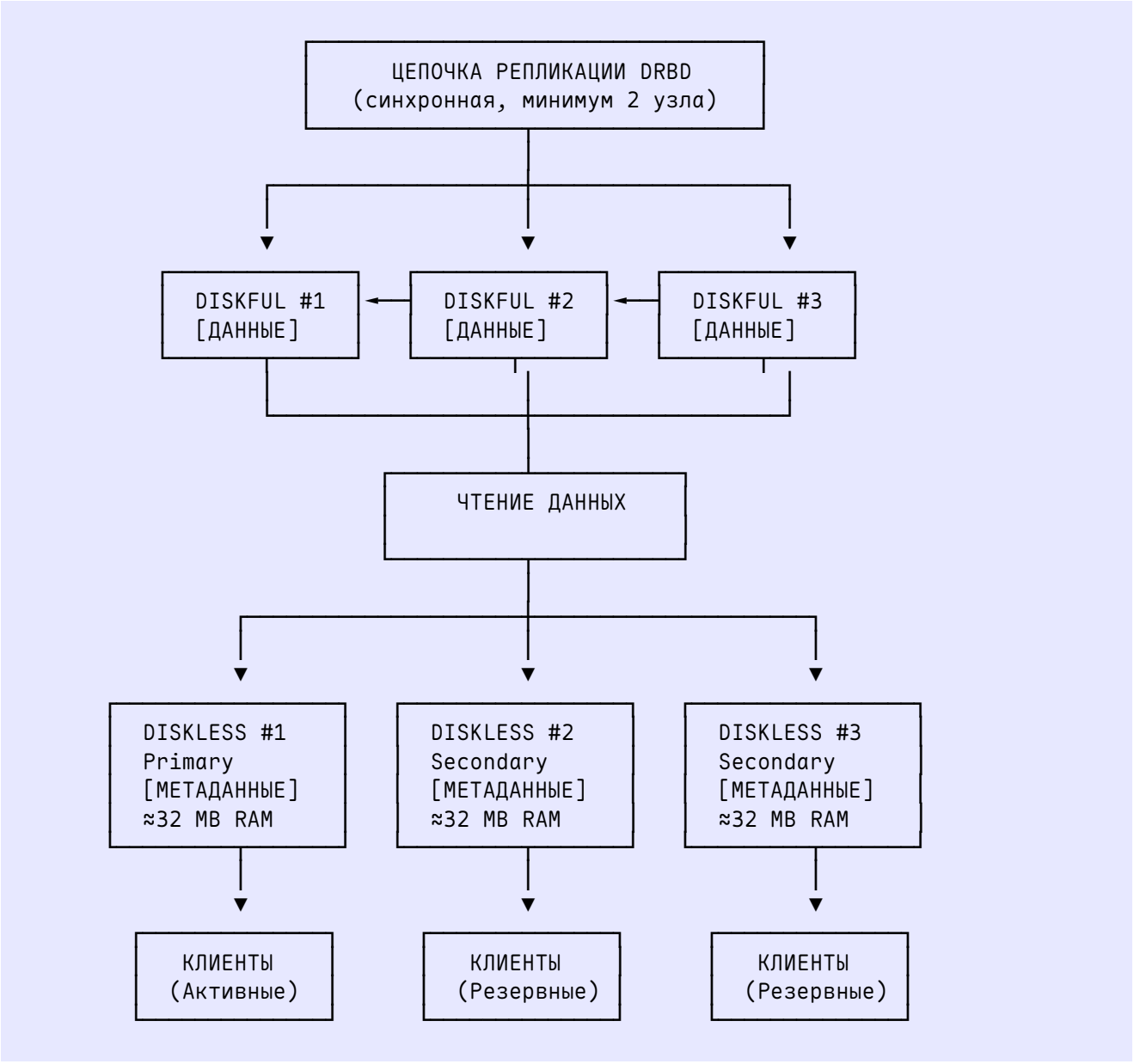
trok-harbor поставляется также и в формате отдельного мета-пакета для установки на узлах типа Worker.

Перед началом конфигурации необходимо обеспечить соблюдение критических условий.

1.1. Бездисковый режим

Diskless (бездисковый) режим – это режим работы DRBD, когда узел не имеет локального дискового хранилища, но участвует в кластере как узел доступа к DRBD-ресурсу.

Архитектура Diskless выглядит таким образом:



Бездисковый узел может выполнять функции арбитра для поддержания кворума в кластере, предотвращая ситуации с разделением ресурсов, когда узлы не могут договориться о том, какие данные являются авторитетными. Он также может выступать в роли клиента, считывающего и записывающего данные с других узлов.

Преимущества режима Diskless:

– не требуется локальный диск на Diskless нодах;

– позволяет размещать Diskless узлы на любой платформе (например, на гипервизорах в конвергентных системах);

– позволяет быстро добавлять и удалять узлы;

– обеспечивает больше возможных точек доступа к данным;

– быстрый failover;

– не теряет связь с ресурсом, если вышел из строя диск на Diskful-узле.

Бездисковый узел имеет следующие особенности:

– нет локального хранения данных - только метаданные в оперативной памяти;

– зависимость от сети - при потере связи данные недоступны;

– требуется минимум 32MB RAM на ресурс для метаданных (32MB на 1 TB DRBD);

– метаданные хранятся в памяти – исчезают после перезагрузки;

– автоматическое восстановление при восстановлении связи.

1.2. Установка и настройка базовых компонентов

Базовые компоненты, необходимые, для работы системы указаны в таблице ниже:

Таблица 1 – Базовые компоненты системы

Компонент	Требования	Проверка выполнения
DRBD + LVM	DRBD версии ≥ 9.0 LVM2 или LVM Thin Provisioning Ядро Linux ≥ 6.1 с поддержкой установки модуля DRBD	sudo lvs --version sudo drbdsetup –version uname -r проверка наличия модуля lsmod | grep drbd

drbd-reactor	Python ≥ 3.6 Конфигурационные файлы в /etc/drbd-reactor/	sudo drbd-reactor --version
WEB-UI	Подключение к API TROK Веб-сервер (Nginx) с HTTPS Порт 80 (или кастомный)	sudo systemctl status nginx
resource-agents	Набор OCF-скриптов для управления ресурсами кластера Поддержка агентов: DRBD, Filesystem, IPaddr2, portblock, nfsserver, exportfs, iSCSITarget, iSCSILogicalUnit	sudo apt list --installed | grep resource-agents dpkg -l | grep resource-agents ls /usr/lib/ocf/resource.d/heartbeat/ # Проверка наличия основных агентов для DRBD и файловых систем ls /usr/lib/ocf/resource.d/heartbeat/ | grep -E "(DRBD|Filesystem|IPaddr2|portblock)" # Проверка наличия основных агентов для сервисов доступа ls /usr/lib/ocf/resource.d/heartbeat/ | grep -E "(nfsserver|exportfs|iSCSI|Samba)"

1.3. Подготовка топологии сети и соблюдение аппаратных требований

Для комфортной работы желательно использовать минимум 3 серверные ноды, имеющие следующие характеристики:

– Процессор: 4+ ядер (x86_64), частота: от 2 ГГц;

– Размер оперативной памяти: 8+ GB;

– Для установки операционной системы: 2 накопителя, каждый из которых имеет емкость 128 ГБ и более (SAS/SATA HDD/SSD);

– Для программного обеспечения TROK: 1 накопитель (и более) с емкостью не менее 128 ГБ (SAS/SATA HDD).

1.4. Сетевые интерфейсы

Рекомендации по скорости соединения:

– Минимум: 1 Гбит/с (1 GbE);

– Рекомендуется: 10 Гбит/с (10 GbE) и выше (особенно для репликации DRBD, чтобы избежать задержек при синхронизации данных между узлами)

Максимальный размер блока данных (в байтах), который может быть передан через сетевой интерфейс без фрагментации (размер MTU):

– Минимум: 1500 байт (стандартный Ethernet);

– Рекомендуется: Jumbo Frames (MTU = 9000) (критично для NVMe-oF, так как увеличивает пропускную способность и снижает накладные расходы при передаче данных).

Для обеспечения стабильной производительности и низкой задержки рекомендуется использовать раздельные сетевые пути для синхронизации данных DRBD и клиентского доступа. Это предотвращает конфликт трафика и повышает надежность системы.

1.5. Сетевые требования

Все ноды должны находиться в одной подсети без NAT (например, 192.168.100.1/24).

Каждое устройство должно иметь статический IP-адрес или использовать резервирование DHCP.

Рекомендуется настроить параметр MTU (Maximum Transmission Unit) на сетевых интерфейсах для оптимизации передачи данных и предотвращения фрагментации пакетов. Для высокоскоростных сетей (10 Гбит/с и выше) рекомендуется использовать значение MTU 9000 (Jumbo Frames) для повышения пропускной способности и снижения нагрузки на процессор. Для корректной настройки параметров обратитесь к официальной документации

вашего дистрибутива, так как конфигурация может зависеть от конкретной реализации и версии ОС.

1.6. Резервное копирование данных

Перед началом любых работ с настройкой системы хранения данных важно сделать резервное копирование. Это поможет избежать проблем, таких как случайное удаление разделов или повреждение данных из-за неправильных команд или неожиданных ошибок.

2. НАСТРОЙКА WEB-UI

Формирование структурных компонентов TROK осуществляется с использованием web-интерфейса, который предоставляет все необходимые функции для контроля и мониторинга ресурсов. Более подробную информацию вы можете найти в документе «TROK v1.0 Руководство по работе с графическим интерфейсом».

Параметры для входа:

username: "admin"

password: "admin".

В следующих разделах будут представлены дополнительные инструкции по работе с командной строкой.

3. ИМЯ УЗЛА И ВОЗМОЖНЫЕ КОНФЛИКТЫ ПРИ СОЗДАНИИ

При выборе уникального имени узла (ноды) рекомендуется учитывать следующие правила:

– имя ноды в системе TROK и имя устройства (хоста) должны совпадать;

– разрешенные базовые символы – латинские буквы: a-z, A-Z; цифры: 0-9; спецсимволы: - (дефис); . (точка).

– начало имени не может начинаться со спецсимвола (например, имя .server недопустимо)

– длина уникального имени должна составлять не более 48 символов.

– регистр букв не учитывается (например, Node1 и node1 считаются идентичными).

– запрещенные символы и форматы – пробелы, кириллица, символы: _, @, #, $, %, ^, &, *, (), {}, [], /, \, :, ;, ", ', ~, !.

– запрещено использовать IP-адреса в качестве имён (например, 192.168.1.1 недопустимо).

Если ваши имя ноды в системе TROK и имя устройства (хоста) отличаются, можно воспользоваться следующими сценариями:

– Изменить имя устройства, используйте команду:

```
sudo hostnamectl set-hostname <имя_узла>
```

Важно. Данная операция может повлечь за собой необходимость корректировки конфигурационных файлов иных программных компонентов, установленных в операционной системе. Поэтому перед выполнением изменения следует тщательно оценить потенциальное влияние на функционирование стороннего программного обеспечения и убедиться в безопасности данной процедуры для текущих процессов.

– Изменить имя ноды в системе TROK. Система не позволяет переименовать узел напрямую, поскольку имя ноды тесно связано с её уникальным идентификатором в базе данных контроллера. Чтобы изменить

имя, необходимо удалить существующую ноду и создать новую с нужным именем на том же устройстве:

– Добавьте новый узел с правильным именем с использованием web-интерфейса системы. Используйте тот же IP-адрес, что у старого узла.

– Удалите старый узел.

Важно. Если узел участвовал в репликации, необходимо выполнить её перенастройку. Рекомендуется применять данную процедуру только в ситуациях, когда узел более не должен входить в состав кластера, перед переустановкой системы или выводом сервера из эксплуатации, а также если узел был добавлен ошибочно.

4. РАЗМЕТКА ДИСКОВ LVM ДЛЯ ПУЛОВ ХРАНЕНИЯ

Разметка дисков производится на нодах, на которых размещена служба trok-worker.

Подготовьте место на диске для дальнейшей работы с помощью утилиты для работы с разделами fdisk:

```
sudo fdisk /dev/<имя_диска>
```

Важно. Все данные на диске будут удалены. Убедитесь, что выбрали правильный диск.

Внутри fdisk введите команды по порядку:

n → создать новый раздел.

p → выбрать тип раздела: основной (primary).

Enter → принять номер раздела по умолчанию.

Enter → принять первый сектор по умолчанию (2048).

Enter → принять последний сектор по умолчанию (весь диск).

w → сохранить изменения и выйти.

Для инициализации раздела под LVM создайте физический том (PV) используя команду:

```
sudo pvcreate /dev/<имя_раздела>
```

pvcreate – команда для инициализации диска/раздела под LVM.

Имя тома (раздела) обычно присваивается по принципу <disk_name> + p + <part_number>, например, nvme0n1p1.

Создайте группу томов (VG):

```
sudo vgcreate <имя_группы_ресурсов> /dev/<имя_раздела>
```

Для проверки созданных объектов используйте команды:

```
sudo pvdisplay # Список физических томов

sudo pvdisplay | grep -B 1 <имя_раздела> # Фильтрует вывод, показывая строки с интересующим именем и дополнительную строку перед ними для контекстного понимания информации.
```

```
   sudo vgdisplay | grep -B 1 <имя_группы_ресурсов> # Аналогичный вывод с фильтрацией по имени тома
```

5. РАЗМЕТКА ДИСКОВ LVM_THIN ДЛЯ ПУЛОВ ХРАНЕНИЯ

Разметка дисков производится на нодах, на которых размещена служба trok-worker.

Подготовьте место на диске для дальнейшей работы с помощью утилиты для работы с разделами fdisk:

```
sudo fdisk /dev/<имя_диска>
```

Важно. Все данные на диске будут удалены. Убедитесь, что выбрали правильный диск.

Внутри fdisk введите команды по порядку:

n → создать новый раздел.

p → выбрать тип раздела: основной (primary).

Enter → принять номер раздела по умолчанию.

Enter → принять первый сектор по умолчанию (2048).

Enter → принять последний сектор по умолчанию (весь диск).

w → сохранить изменения и выйти.

Для создания LVM Thin Pool необходимо выполнить следующую последовательность команд:

Создать физический том (PV):

```
sudo pvcreate /dev/<имя_раздела> # помечаем раздел как «физический том» для LVM
```

pvcreate – команда для инициализации диска/раздела под LVM.

/dev/<имя_раздела> – созданный раздел (после fdisk).

Создать группу томов (VG):

```
sudo vgcreate <имя_группы_ресурсов> /dev/<имя_раздела>
```

vgcreate – создает группу томов (логический контейнер для дисков).

Имя группы (можно выбрать любое, например, data_vg).

Создать «тонкий пул» в группе (Thin Pool):

```
sudo lvcreate -l 100%FREE -T <имя_группы_ресурсов>/<имя_тома_lvm_thin>
```

lvcreate – создает логический том.

-l 100%FREE – использовать всё свободное место в группе.

-T – указание создать Thin Pool (пул с динамическим выделением места).

<имя_группы_ресурсов>/<имя_тома_lvm_thin> – том располагается внутри группы.

Для проверки созданных сущностей воспользуйтесь командой:

```
sudo lvdisplay | grep -E "LV Name|VG Name
```

В случае возникновения необходимости в удалении созданных ресурсов, можно воспользоваться следующими командами:

– удаление группы томов:

```
sudo vgremove <имя_группы_ресурсов>
```

– удаление физического тома:

```
sudo pvremove /dev/<имя_раздела>
```

6. НАСТРОЙКА ДОСТУПА ПО ПРОТОКОЛАМ

6.1. Настройка отказоустойчивого высокодоступного хранилища NFS с использованием drbd-reactor

Первый этап настройки включает в себя подготовку операционной системы.

Обновите кэш пакетов:

```
apt update
```

Установите NFS-сервер и OCF-агенты:

```
apt install -y nfs-kernel-server resource-agents
```

Выполните команды для отключения и остановки стандартного NFS-сервиса:

```
systemctl disable nfs-server
systemctl stop nfs-server
```

Активируйте службу drbd-reactor:

```
systemctl enable drbd-reactor
```

Вторым этапом следует подготовка файловой системы.

Активируйте первичный режим DRBD. На основном узле переведите ресурс в состояние Primary:

```
drbdadm primary <имя_определения_ресурса>
```

Отформатируйте том DRBD в файловую систему ext4 (создайте файловую систему):

```
mkfs -t ext4 /dev/drbd/by-res/<имя_определения_ресурса>/0
```

Создайте структуру каталогов. Подготовьте каталоги для монтирования и служебные директории:

```
mkdir -p /srv/ha/internal/<имя_ресурса>
mkdir -p /srv/ha/internal/<имя_ресурса>/.tickle
mkdir -p /srv/ha/internal/<имя_ресурса>/.nfs
```

Предоставьте полные права доступа на каталог для тестирования:

```
chmod -R 777 /srv/ha/internal/<имя_ресурса>
```

Переведите DRBD-ресурс в Secondary режим:

```
drbdadm secondary <имя_определения_ресурса>
```

Далее следуют шаги для конфигурации службы drbd-reactor.

Создайте конфигурационный файл для promoter'a:

```
touch /etc/drbd-reactor.d/nfs-<имя_ресурса>.toml
```

Откройте файл /etc/drbd-reactor.d/nfs-<имя_ресурса>.toml и добавьте следующую конфигурацию:

```
[[promoter]]

  [promoter.metadata]
    harbor-schema-version = 1

  [promoter.resources]

    [promoter.resources.<имя_определения_ресурса>]
      on-drbd-demote-failure = "reboot-immediate"
      runner = "systemd"
      start = [
        "ocf:heartbeat:portblock    portblock    action=block    ip=192.168.1.130
portno=2049 protocol=tcp",
        "ocf:heartbeat:Filesystem            fs_1            device=/dev/drbd/by-
res/<имя_определения_ресурса>/0 directory=/srv/ha/internal/<имя_ресурса> fstype=ext4
run_fsck=no",
        "ocf:heartbeat:IPaddr2 service_ip cidr_netmask=24 ip=192.168.1.130",
        "ocf:heartbeat:nfsserver          nfsserver          nfs_ip=192.168.1.130
nfs_server_scope=192.168.1.130
nfs_shared_infodir=/srv/ha/internal/<имя_ресурса>/.nfs/nfsinfo",
        "ocf:heartbeat:exportfs        export_1_0        clientspec=192.168.1.0/24
directory=/srv/ha/internal/<имя_ресурса>    fsid=cbc3c45e-594b-52be-bb15-49909a2c2008
options='rw,all_squash,anonuid=0,anongid=0'",
        "ocf:heartbeat:portblock   portunblock   action=unblock   ip=192.168.1.130
portno=2049 protocol=tcp tickle_dir=/srv/ha/internal/<имя_ресурса>/.tickle",
      ]
      stop-services-on-exit = true
      target-as = "BindsTo"
```

Перезапустите службу drbd-reactor для применения конфигурации:

```
systemctl restart drbd-reactor
```

Проверьте статус drbd-reactor с помощью команды:

```
drbd-reactorctl status
```

Убедитесь, что promoter активен и все сервисы запущены.

Для проверки NFS экспорта выполните команду:

```
exportfs
```

Убедитесь, что экспорт настроен для вашей сети.

Протестируйте подключение смонтировав NFS-шару для проверки:

```
mount -t nfs 192.168.1.130:/srv/ha/internal/<имя_ресурса> /mnt
```

6.2. Настройка отказоустойчивого высокодоступного хранилища Samba с использованием drbd-reactor

Первый этап настройки включает в себя установку необходимых пакетов.

Добавьте репозитории Астра Linux в файл /etc/apt/sources.list:

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

deb        https://download.astralinux.ru/astra/stable/1.8_x86-64/repository-main/ 1.8_x86-64 main contrib non-free non-free-firmware
```

Для обновления кэша пакетов выполните команду на всех узлах кластера:

```
sudo apt update
```

Установите Samba, DRBD Reactor и OCF-агенты на всех узлах:

```
sudo apt install samba samba-client drbd-reactor resource-agents
```

Поскольку службой Samba будет управлять drbd-reactor, отключите автостарт Samba:

```
sudo systemctl disable smbd

sudo systemctl stop smbd
```

Следующим шагом выполняется настройка поддержки современных версий SMB.

Отредактируйте файл конфигурации Samba на всех узлах:

```
sudo vim /etc/samba/smb.conf
```

Вставьте следующие параметры в секцию [global] (добавьте секцию, если она отсутствует):

```
[global]
   client min protocol = SMB2_02
   server min protocol = SMB2_02
   client max protocol = SMB3_11
   server max protocol = SMB3_11
```

Подготовьте DRBD ресурса и миграцию данных Samba.

На одной из нод переведите DRBD ресурс в первичный режим:

```
sudo drbdadm primary <имя_определения_ресурса>
```

Отформатируйте устройство DRBD в файловую систему XFS (создайте файловую систему):

```
sudo mkfs.xfs /dev/drbd3650
```

Подготовьте директорию Samba. Создайте резервную копию данных и новую директорию:

```
sudo mv /var/lib/samba /var/lib/samba.bak

sudo mkdir /var/lib/samba
```

Смонтируйте DRBD устройство и восстановите данные:

```
sudo mount /dev/drbd3650 /var/lib/samba

sudo cp -av /var/lib/samba.bak/* /var/lib/samba/
```

Настройте drbd-reactor, выполнив следующую последовательность действий.

На всех узлах создайте файл конфигурации:

```
sudo vim /etc/drbd-reactor.d/<имя_определения_ресурса>.toml
```

Отформатируйте файл, вставив следующий конфигурационный блок:

```
[[promoter]]
id = "<имя_определения_ресурса>"
[promoter.resources.<имя_определения_ресурса>]
on-drbd-demote-failure = "reboot-immediate"
```

```
runner = "systemd"
preferred-nodes = []
start = [
   "ocf:heartbeat:Filesystem fs_1 device=/dev/drbd/by-res/<имя_определения_
pecypca>/0 directory=/var/lib/samba fstype=xfs run_fsck=no",
   "ocf:heartbeat:IPaddr2 service_ip cidr_netmask=24 ip=10.177.161.250",
   "smbd.service",
]
stop-services-on-exit = true
target-as = "BindsTo"
```

Последним шагом будет запуск проверка конфигурации и тестирование подключения.

Перезапустите drbd-reactor. Выполните команду на всех узлах:

```
sudo drbd-reactorctl restart
```

Убедитесь, что конфигурация загружена и ресурс активен:

```
sudo drbd-reactorctl status
```

Ожидаемый вывод:

```
/etc/drbd-reactor.d/<имя_определения_ресурса>.toml:
Promoter: Currently active on this node
● drbd-services@<имя_определения_ресурса>.target
● ├─ drbd-promote@<имя_определения_ресурса>.service
● ├─ ocf.rs@fs_1_<имя_определения_ресурса>.service
● ├─ ocf.rs@service_ip_<имя_определения_ресурса>.service
● └─ smbd.service
```

Проверьте доступность Samba-ресурса, выполнив команду с любого клиента:

```
smbclient -L //10.177.161.250 -m SMB3_11 -U%
```

Ожидаемый результат:

```
        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        IPC$            IPC       IPC Service (Samba 4.21.4-Debian-4.21.4+dfsg-1as
tra4+b2)
        nobody          Disk      Home Directories
SMB1 disabled -- no workgroup available
```

Проверка состояния DRBD ресурсов осуществляется с помощью команды для мониторинга репликации:

```
sudo drbdadm status
```

Состояние сервисов под управлением drbd-reactor можно проверить, выполнив команду:

```
sudo drbd-reactorctl status
```

6.3. Настройки отказоустойчивого высокодоступного iSCSI Target с использованием drbd-reactor

Первый этап настройки включает в себя установку необходимых пакетов.

Выполните следующие команды для установки компонентов iSCSI Target (серверная часть):

```
sudo apt update

sudo apt install targetcli-fb resource-agents -y
```

Для тестирования подключения установите iSCSI инициатор на клиентских узлах:

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

Вторым этапом настройте конфигурации DRBD Reactor для iSCSI, а также конфигурацию iSCSI Initiator (клиентской части).

Создайте или отредактируйте файл конфигурации DRBD Reactor:

```
vim /etc/drbd-reactor.d/iscsi-ssd1.toml
```

Вставьте следующий конфигурационный блок iSCSI в файл:

```
[[promoter]]

[promoter.metadata]
harbor-schema-version = 1

[promoter.resources]

[promoter.resources.ssd1]
on-drbd-demote-failure = "reboot-immediate"
runner = "systemd"
start = [
  "ocf:heartbeat:portblock  pblock0  action=block  ip=185.227.37.224  portno=3260
```

```
protocol=tcp",
     "ocf:heartbeat:IPaddr2 service_ip0 cidr_netmask=24 ip=185.227.37.224",
     "ocf:heartbeat:iSCSITarget              target              allowed_initiators=''
incoming_password=12345678   incoming_username=astra   iqn=iqn.2025-12.ru.astra:ssd1
portals=185.227.37.224:3260",
     "ocf:heartbeat:iSCSILogicalUnit    lu1    lun=1    path=/dev/drbd/by-res/ssd1/0
product_id='TROK iSCSI' scsi_sn=7509fab1 target_iqn=iqn.2025-12.ru.astra:ssd1",
     "ocf:heartbeat:portblock    portunblock0    action=unblock    ip=185.227.37.224
portno=3260 protocol=tcp tickle_dir=/srv/ha/internal/ssd1",
   ]
   stop-services-on-exit = true
   target-as = "Requires"
```

Задайте уникальное имя для идентификации iSCSI инициатора:

```
   echo "InitiatorName=iqn.2025-12.ru.astra:ssd1" > /etc/iscsi/initiatorname.iscsi
```

Отредактируйте файл конфигурации iSCSI и укажите учетные данные для настройки аутентификации CHAP:

```
   vim /etc/iscsi/iscsid.conf
```

Добавьте или измените следующие строки:

```
   node.session.auth.username = astra

   node.session.auth.password = 12345678
```

Выполните команды для обнаружения и подключения к iSCSI Target:

```
   iscsiadm -m discovery -t sendtargets -p 185.227.37.224

   iscsiadm -m node --login
```

Выполните команду для получения уникального SCSI ID:

```
   sudo /usr/lib/udev/scsi_id -g -u /dev/sdd
```

Запишите полученный идентификатор (например: 360014057509fab100000000000000000).

Создайте файл правила для автоматического создания символической ссылки:

```
   cat > /etc/udev/rules.d/30-change-iscsi-name.rules << EOF
   KERNEL=="sd[a-z]",   SUBSYSTEM=="block",   PROGRAM="/usr/lib/udev/scsi_id   -g   -u
/dev/\$name", RESULT=="360014057509fab100000000000000000", SYMLINK+="iscsi/disk_1"
```

```
EOF
```

Перезагрузите правила udev для немедленного применения:

```
udevadm control --reload-rules
```

Отключитесь от iSCSI Target (при необходимости). Для удаления подключения выполните следующие команды:

```
iscsiadm -m node -p 185.227.37.224:3260,0 -T iqn.2025-12.ru.astra:ssd1 --logout
iscsiadm -m node -p 185.227.37.224:3260,0 -T iqn.2025-12.ru.astra:ssd1 -o delete
```

На финальном этапе следует настроить проверку и мониторинг.

Проверьте статус drbd-reactor. Убедитесь, что конфигурация загружена и ресурс активен:

```
sudo drbd-reactorctl status
```

Проверьте доступность iSCSI Target. С клиентского узла убедитесь, что диск доступен по символической ссылке:

```
ls -la /dev/iscsi/disk_1
```

7. МОНИТОРИНГ PROMETHEUS ДЛЯ КОНТРОЛЛЕРОВ

Программное обеспечение предоставляет возможности для интеграции контроллеров с системой мониторинга Prometheus. В случае, если контроллер запущен, он автоматически экспортирует метрики в формате Prometheus, и данные могут быть извлечены посредством запроса к конечной точке http:// controller’s-ip/metrics.

7.1. Метрика trok_info

Метрика trok_info предоставляет базовую информацию о версии и сборке контроллера и содержит метки:

– getid – уникальный идентификатор сборки (хеш коммита Git);

– buildtime – время сборки программного обеспечения;

– version – версия программного обеспечения (семантическое версионирование).

Пример значения:

```
trok_info{getid="372c916b7d97fa10e8ea480b66ea3da665ab5849",buildtime="2025-10-13T14:22:02+03:00",version="1.29.2"} 1
```

Интерпретация:

Значение всегда равно 1 – подтверждает доступность метрик. Используется для проверки работоспособности экспортера и позволяет отслеживать версии в гетерогенных средах.

7.2. Метрика trok_node_state

Метрика trok_node_state отслеживает состояние узлов в кластере и имеет следующие коды состояний:

– 0 = OFFLINE – узел недоступен;

– 1 = ONLINE – узел работает нормально;

– 2 = CONNECTING – выполняется подключение к узлу;

– 3 = EVICTED – узел исключен из кластера.

Метки:

- node – имя узла в кластере;
- nodetype – тип узла (satellite или controller);
- address – IP-адрес узла;
- port – порт для подключения;
- encryption – тип шифрования (PLAIN = без шифрования).

Пример значения:

```
trok_node_state{node="nodeA",nodetype="SATELLITE",address="10.177.161.43",port="50002",encryption="PLAIN"} 1
```

7.3. Метрика trok_storage_pool_capacity_free_bytes

Метрика trok_storage_pool_capacity_free_bytes отображает свободное место в пулах хранения и содержат следующие метки:

- storage_pool – имя пула хранения;
- node – UUID узла, на котором расположен пул;
- driver – тип драйвера хранилища:
  - LVM – классический LVM;
  - LVM_THIN – тонкий LVM;
  - ZFS – файловая система ZFS;
  - DISKLESS – бездисковый пул.

Пример значения:

```
trok_storage_pool_capacity_free_bytes{storage_pool="PoolA",node="0acfd509-93a6-4504-9df9-5f078095f42b",driver="LVM"} 1234000

trok_storage_pool_capacity_free_bytes{storage_pool="PoolB",node="0acfd509-93a6-4504-9df9-5f078095f42b",driver="LVM_THIN"} 1234000
```

7.4. Метрика trok_storage_pool_capacity_total_bytes

Метрика trok_storage_pool_capacity_total_bytes отображает общую емкость пулов хранения и использует набор меток аналогичный пункту 1.3.

Пример значения:

```
   trok_storage_pool_capacity_total_bytes{storage_pool="PoolA",node="0acfd509-
93a6-4504-9df9-5f078095f42b",driver="LVM"} 1236000

   trok_storage_pool_capacity_total_bytes{storage_pool="PoolB",node="0acfd509-
93a6-4504-9df9-5f078095f42b",driver="LVM_THIN"} 1236000
```

7.5. Метрика trok_storage_pool_error_count

Метрика trok_storage_pool_error_count является счетчиком ошибок в пулах хранения и использует метки аналогичные предыдущим метрикам пулов.

Пример значения:

```
   trok_storage_pool_error_count{storage_pool="PoolA",node="0acfd509-93a6-4504-
9df9-5f078095f42b",driver="LVM"} 0

   trok_storage_pool_error_count{storage_pool="PoolB",node="0acfd509-93a6-4504-
9df9-5f078095f42b",driver="LVM_THIN"} 0
```

Интерпретация:

- 0 – ошибок нет;
- >0 – имеются проблемы с пулом.

Используется для уведомлений о проблемах хранилища.

8. МОНИТОРИНГ PROMETHEUS ДЛЯ DRBD REACTOR

Настройка DRBD Reactor для экспорта метрик Prometheus так же проста. Следует разместить следующий файл конфигурации в директории конфигураций DRBD Reactor и перезапустить DRBD Reactor:

```
# cat << EOF > /etc/drbd-reactor.d/prometheus.toml
[[prometheus]]
enums = true
address = "0.0.0.0:9942"
EOF

# systemctl restart drbd-reactor.service
```

После перезапуска вы сможете отправить curl-запрос на конечную точку метрик и просмотреть метрики DRBD, предоставляемые DRBD Reactor.

```
# curl 127.0.0.1:9942
# TYPE drbd_device_alwrites_total counter
# HELP Number of updates of the activity log area of the meta data
drbd_device_alwrites_total{name="trok_db",volume="0",minor="1000"} 1
# TYPE drbd_resource_resources gauge
# HELP Number of resources
drbd_resource_resources 1
…snip…
```

9. ПРОЦЕДУРА ПРОВЕРКИ И УСТАНОВКИ ПАКЕТА TROK-WEBUI

Прежде чем приступить к настройке системы, важно убедиться в том, что пакет trok-webui корректно установлен на целевом сервере. Пакет является компонентом для развертывания веб-интерфейса и включает в себя необходимые зависимости. В случае отсутствия пакета его установка обязательна для обеспечения функциональности системы.

После установки рекомендуется вручную проверить и, при необходимости, скорректировать конфигурацию nginx для соответствия специфическим требованиям среды (например, настройка SSL, прокси-параметров или ограничений доступа).

Для проверки статуса установки пакета trok-webui воспользуйтесь командой управления пакетами:

```
dpkg -l | grep trok-webui
```

Команда фильтрует вывод списка всех установленных пакетов (dpkg -l) по ключевому слову trok-webui, отображая только релевантные записи. Если пакет установлен, вы увидите вывод, подобный следующему примеру:

```
trok-webui   1.0.2~rc6   amd64   TROK Web UI frontend
```

Здесь:

– trok-webui – имя пакета;

– 1.0.2~rc6 – версия пакета (может варьироваться в зависимости от доступной версии в репозиториях);

– amd64 – архитектура (для 64-битных систем);

– Web UI frontend – краткое описание пакета.

Если команда возвращает такой вывод пакет trok-webui успешно установлен, и конфигурация nginx должна быть автоматически применена.

Если вывод команды dpkg -l | grep trok-webui пустой (то есть пакет не найден), необходимо выполнить установку. Для этого используйте менеджер

пакетов apt, который автоматически разрешит зависимости и загрузит пакет из официальных репозиториев.

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

После выполнения этой команды система:

– Скачает пакет trok-webui и его зависимости (включая nginx, если они не установлены ранее).

– Установит их в соответствующие директории.

– Автоматически настроит базовую конфигурацию nginx для работы с trok-webui.

В процессе установки следите за сообщениями в терминале: они могут содержать важную информацию о потенциальных конфликтах или дополнительных шагах (например, перезапуск служб). После завершения установки рекомендуется перезапустить службу nginx для применения изменений:

```
sudo systemctl restart nginx
```

10. НАСТРОЙКА SSL-СЕРТИФИКАТОВ ДЛЯ NGINX

Для обеспечения безопасного соединения с веб-интерфейсом TROK рекомендуется использовать SSL/TLS-сертификаты. Это позволит шифровать трафик между клиентом (браузером) и сервером, предотвращая перехват данных. Без SSL соединение будет происходить по HTTP, что уязвимо для современных угроз.

Для файлов сертификатов и приватных ключей рекомендуется использовать стандартную директорию в структуре Nginx, такую как /etc/nginx/ssl.

Для создания директории выполните команду:

```
sudo mkdir -p /etc/nginx/ssl
```

Флаг -p (parents) обеспечивает создание родительских директорий, если они отсутствуют (хотя в большинстве случаев /etc/nginx уже существует).

Если в вашей системе используется альтернативная структура (например, /etc/ssl/private), адаптируйте путь соответственно.

Установка прав доступа выполняется командой:

```
sudo chmod 700 /etc/nginx/ssl
```

Команда устанавливает права drwx------: только владелец (обычно root) может читать, писать и выполнять. Группа и другие пользователи полностью исключены.

Приватные ключи – это чувствительные данные. Несанкционированный доступ может привести к компрометации сервера. После создания файлов внутри директории вы можете дополнительно настроить владение (chown) для пользователя nginx, если требуется (например, sudo chown -R nginx:nginx /etc/nginx/ssl).

После выполнения этих шагов директория готова. Проверьте её наличие и права командой:

```
ls -ld /etc/nginx/ssl
```

В результате выполнения команды должно появиться сообщение похожее на drwx------ 2 root root 4096 [дата] /etc/nginx/ssl. Теперь перейдем к генерации или установке сертификата.

При генерации или установке SSL-сертификата выбор метода зависит от вашей среды: для разработки подойдет самоподписанный сертификат, для корпоративной сети – внутренний CA, а для продакшена с внешним провайдером – fullchain от Let's Encrypt или аналогичного. Сертификат состоит из публичного файла (.crt или .pem) и приватного ключа (.key). После установки следует перезапустить Nginx (sudo systemctl restart nginx), чтобы изменения применились, а также протестировать соединение.

10.1. Создание самоподписанного (self-signed) сертификата

Самоподписанный сертификат генерируется с помощью OpenSSL и не требует внешнего CA, но браузеры и клиенты будут считать его недоверенным, показывая предупреждение «Небезопасное соединение».

Выполните единую команду для генерации пары ключ + сертификат (замените параметры на свои):

```
sudo openssl req -x509 -nodes -days 365 \
  -newkey rsa:4096 \
  -keyout /etc/nginx/ssl/trok-webui.key \
  -out /etc/nginx/ssl/trok-webui.crt \
  -subj "/C=RU/ST=Region/L=City/O=TROK/OU=WebUI/CN=192.168.1.10"
```

Разбор параметров:

– req -x509: генерирует запрос на сертификат и сразу самоподписывает его (X.509 – стандарт формата).

– -nodes: не шифрует приватный ключ паролем.

– -days 365: срок действия – 1 год (можно увеличить до 825 дней, но не более, чтобы соответствовать рекомендациям).

– -newkey rsa:4096: создает новый RSA-ключ длиной 4096 бит (высокий уровень безопасности; альтернатива – ECDSA для лучшей производительности).

– -keyout и -out: пути к файлам ключа и сертификата в директории.

– -subj: строка с данными субъекта (Subject). Здесь:

– C=RU: страна (Russia).

– ST=Region: регион (замените на свой, например, "Moscow").

– L=City: город (например, "Moscow").

– O= TROK: организация.

– OU=WebUI: подразделение.

– CN=192.168.1.10: Common Name – IP-адрес или домен сервера. Важно: CN должен точно совпадать с адресом, по которому клиенты подключаются (IP или hostname). Если используете доменное имя, замените на CN=trok-webui.local (или реальный домен, например, trok.example.com).

После выполнения проверьте файлы: ls /etc/nginx/ssl/ – должны появиться trok-webui.key и trok-webui.crt.

Установите права на ключ (только чтение для владельца):

```
sudo chmod 600 /etc/nginx/ssl/trok-webui.key
```

10.2. Установка сертификата, выданного внутренним CA

Собственный центр сертификации (внутренний CA) обеспечивает доверие внутри домена без обращения к публичным CA. Процесс включает копирование файлов сервера и настройку цепочки доверия (root CA + intermediate CA), чтобы клиенты автоматически доверяли сертификату.

Для начала создания у вас должны быть: server.crt (сертификат сервера), server.key (приватный ключ), rootCA.crt (корневой CA) и intermediateCA.crt (промежуточный CA, если есть).

Для копирования сертификата и ключа сервера выполните команды:

```
sudo cp server.crt /etc/nginx/ssl/trok-webui.crt

sudo cp server.key /etc/nginx/ssl/trok-webui.key
```

Установите права:

```
sudo chmod 600 /etc/nginx/ssl/trok-webui.*.
```

Данные операции изолируют файлы в директории Nginx и предотвращает случайное удаление оригиналов.

Перед установкой убедитесь, что сертификат сервера правильно подписан CA. Выполните:

```
openssl verify -CAfile rootCA.crt trok-webui.crt
```

Если цепочка полная (intermediate включен), используйте -CAfile intermediateCA.crt -untrusted rootCA.crt для многоуровневой проверки.

Ожидаемый вывод:

```
trok-webui.crt: OK.
```

Если возникает ошибка (например, «unable to get local issuer certificate»), проверьте файлы на валидность (openssl x509 -in trok-webui.crt -text -noout).

Установка CA-сертификатов в системное хранилище доверия:

Разные дистрибутивы имеют свои пути установки CA-сертификатов в системное хранилище доверия.

10.2.1. Установка CA-сертификатов для Debian/Ubuntu/Astra Linux

Выполните следующие команды:

```
sudo cp rootCA.crt /usr/local/share/ca-certificates/rootCA.crt
sudo cp intermediateCA.crt /usr/local/share/ca-certificates/intermediateCA.crt
sudo update-ca-certificates
```

update-ca-certificates обновит хранилище /etc/ssl/certs/ и перезапустит службы.

10.2.2. Установка CA-сертификатов для RHEL/AlmaLinux/Rocky Linux

Выполните:

```
sudo cp rootCA.crt /etc/pki/ca-trust/source/anchors/
```

```
sudo cp intermediateCA.crt /etc/pki/ca-trust/source/anchors/

sudo update-ca-trust
```

Это обновит глобальное хранилище NSS (Network Security Services), используемое Firefox и другими.

10.2.3. Установка CA-сертификатов для Windows-клиентов (если сервер обслуживает Windows-машины)

Выполните следующие шаги:

– Откройте инструмент certmgr.msc (Run → certmgr.msc);

– Импортируйте rootCA.crt в «Trusted Root Certification Authorities» (Доверенные корневые центры сертификации);

– Импортируйте intermediateCA.crt в "Intermediate Certification Authorities" (Промежуточные центры сертификации);

– Для групповой политики: Разверните через GPO или скрипты (certutil.exe).

10.3. Использование сертификата с полной цепочкой (fullchain)

Если ваш CA (внутренний или внешний, например, Let's Encrypt) выдает сертификат в формате fullchain (сертификат сервера + встроенная цепочка CA в одном файле), процесс упрощается – нет нужды вручную устанавливать промежуточные CA на сервере. Это удобно для автоматизированных систем.

У вас есть: trok-webui.fullchain.crt (fullchain-сертификат) и trok-webui.key (ключ).

Скопируйте их, используя команды:

```
sudo cp trok-webui.fullchain.crt /etc/nginx/ssl/trok-webui.fullchain.crt

sudo cp trok-webui.key /etc/nginx/ssl/trok-webui.key

sudo chmod 600 /etc/nginx/ssl/trok-webui.*
```

В файле конфигурации сайта (обычно /etc/nginx/sites-available/default или /etc/nginx/conf.d/trok.conf) в блоке server для HTTPS (порт 443) укажите:

```
ssl_certificate /etc/nginx/ssl/trok-webui.fullchain.crt;

ssl_certificate_key /etc/nginx/ssl/trok-webui.key;
```

Полный пример блока:

```
server {
    listen 443 ssl http2;
    server_name trok-webui.local;
    ssl_certificate /etc/nginx/ssl/trok-webui.fullchain.crt;
    ssl_certificate_key /etc/nginx/ssl/trok-webui.key;
    # Другие настройки: ssl_protocols TLSv1.2 TLSv1.3; etc.
}
```

Проверьте конфигурацию:

```
sudo nginx -t
```

Если ответ OK, перезапустите nginx:

```
sudo systemctl restart nginx
```

11. ВКЛЮЧЕНИЕ HTTPS В ФАЙЛЕ КОНФИГУРАЦИИ

Для получения доступа к файлу конфигурации введите команду:

```
sudo nano /etc/nginx/sites-available/trok-webui-nginx.conf
```

Путь /etc/nginx/sites-available/ – стандартная директория в Nginx, где хранятся конфигурации виртуальных хостов (сайтов). Файлы из этой директории обычно подключаются к активной конфигурации через симлинк в директорию /etc/nginx/sites-enabled/.

Внутри файла найдите закомментированный раздел, который скорее всего помечен как SSL CONFIG (OPTIONAL) или подобным образом. Раскомментируйте следующие критически важные директивы:

```
# Находим и убираем символ '#' в начале каждой нужной строки
listen 443 ssl;
http2 on;
ssl_certificate     /etc/nginx/ssl/trok-webui.crt;
ssl_certificate_key /etc/nginx/ssl/trok-webui.key;
ssl_session_cache   shared:SSL:10m;
ssl_session_timeout 1h;
ssl_protocols       TLSv1.2 TLSv1.3;
ssl_ciphers         HIGH:!aNULL:!MD5;
```

Значение каждой директивы:

– listen 443 ssl – указывает Nginx прослушивать входящие соединения на стандартном HTTPS-порту 443 и обрабатывать их с использованием SSL/TLS. Без этой директивы Nginx будет работать только по HTTP на порту 80, и шифрование применяться не будет.

– http2 on – включает поддержку современного протокола HTTP/2. HTTP/2 значительно увеличивает производительность веб-сайта за счёт мультиплексирования запросов, сжатия заголовков и других оптимизаций. Для его работы обязательно требуется HTTPS.

– ssl_certificate и ssl_certificate_key – указывают пути к файлу вашего SSL-сертификата (публичная часть) и соответствующему ему закрытому ключу. Nginx должен знать, где находятся эти файлы, чтобы установить безопасное

соединение. Убедитесь, что пути указаны абсолютно верно и файлы существуют.

– ssl_session_cache shared:SSL:10m; – настраивает кеш для параметров SSL-сессий размером 10 мегабайт, совместно используемый между всеми рабочими процессами Nginx (shared). При повторном соединении клиент может использовать данные из кеша сессии (resumption), что позволяет избежать полного и ресурсоемкого TLS-рукопожатия. Это значительно снижает нагрузку на процессор и ускоряет установление соединения.

– ssl_session_timeout 1h; – устанавливает время жизни кешированных параметров сессии – в данном случае на 1 час. Определяет, как долго клиент может возобновлять сессию без повторного рукопожатия. Баланс между безопасностью (меньшее время) и производительностью (большее время).

– ssl_protocols TLSv1.2 TLSv1.3; – явно разрешает для использования только современные и безопасные версии протоколов TLS версий 1.2 и 1.3. Старые протоколы (SSLv2, SSLv3, TLSv1.0, TLSv1.1) имеют уязвимости и считаются небезопасными. Эта директива запрещает их использование, повышая безопасность.

– ssl_ciphers HIGH:!aNULL:!MD5; – задает набор шифров, которые можно использовать при соединении. Позволяет использовать только самые сильные и безопасные алгоритмы шифрования, исключая уязвимые варианты.

– HIGH – разрешает только шифры с высокой стойкостью.

– !aNULL – запрещает шифры с анонимной аутентификацией (уязвимые для атак).

– !MD5 – запрещает шифры, использующие устаревший и небезопасный алгоритм хеширования MD5.

После внесения изменений и сохранения файла обязательно выполните следующее:

Проверьте синтаксис конфигурации командой:

```
sudo nginx -t
```

Успешный результат должен выглядеть следующим образом:

```
nginx: configuration file /etc/nginx/nginx.conf test is successful
```

Если проверка прошла успешно, примените конфигурацию – перезагрузите или перезапустите службу Nginx, чтобы изменения вступили в силу.

```
sudo systemctl reload nginx

# Или:

sudo systemctl restart nginx
```

reload предпочтительнее, так как он плавно загружает новую конфигурацию без полной остановки работы сервера.

Теперь веб-интерфейс TROK должен быть доступен по безопасному HTTPS-соединению по адресу https://your-server-ip-or-domain. Вы можете проверить это в браузере или с помощью утилит командной строки, таких как curl или openssl s_client.

12. АКТИВАЦИЯ БЛОКА РЕДИРЕКТА (ОПЦИОНАЛЬНО)

Чтобы обеспечить безопасность и соответствие веб-стандартам, рекомендуется настроить автоматическое перенаправление всех HTTP-запросов на защищенное HTTPS-соединение.

В вашем конфигурационном файле Nginx (/etc/nginx/sites-available/trok-webui-nginx.conf) найдите закомментированный блок сервера, предназначенный для редиректа. Обычно он выглядит следующим образом:

```
# server {
#   listen 80;
#   return 301 https://$host$request_uri;
# }
```

Раскомментируйте этот блок, удалив символы # в начале каждой строки.