УТВЕРЖДЕН

RU.46939656.58.29.29.000-02 99 01-ЛУ

Программное средство обработки информации

«Платформа управления документами и бизнес-процессами «ГОССЭД»

Руководство по установке

RU.46939656.58.29.29.000-02 99 01

Листов 39

2024

СОДЕРЖАНИЕ

1. ОБЩИЕ СВЕДЕНИЯ

1.1. Краткое описание целей документа

В настоящем руководстве содержится описание установки программного средства обработки информации «Платформа управления документами и бизнес-процессами «ГОССЭД» (ГОССЭД, система) для операционной системы (ОС) Astra Linux.

1.2. Используемые технологии

Для функционирования ГОССЭД используются следующие технологии:

- Docker - платформа для автоматизации развертывания приложений в контейнерах, что позволяет обеспечить их изоляцию и портативность;
- Docker Swarm - встроенный инструмент для управления кластером Docker, который позволяет объединять несколько Docker-узлов в единый кластер для обеспечения высокой доступности и масштабируемости приложений;
- Manticore Search - высокопроизводительная система полнотекстового поиска, предназначенная для работы с большими объемами данных и обеспечивающая быстрый доступ к информации;
- Redis - система управления базами данных в памяти, используемая для кэширования и хранения данных с высокой производительностью;
- MinIO - высокопроизводительное решение для хранения объектов, совместимое с Amazon S3, которое позволяет разрабатывать облачные приложения;
- RabbitMQ - система управления очередями сообщений, обеспечивающая надежную передачу данных между компонентами приложений;
- PostgreSQL - объектно-реляционная система управления базами данных, известная своей надежностью и расширяемостью;
- HAProxy - программное обеспечение для балансировки нагрузки и проксирования, которое обеспечивает высокую доступность и распределение трафика между серверами.

1.3. Состав поставки

В состав поставки ГОССЭД входят файлы, перечисленные в таблице (Таблица 1).

Таблица 1 - Состав поставки ГОССЭД

Название файла	Описание
haproxy/haproxy.cfg	Файл конфигурации прокси сервера.
swarm/docker-compose.yml	Основной файл решения для запуска в контейнерах docker swarm
images/gossed_app_auth.tar	Файл образа сервиса авторизации
images/gossed_app_root.tar	Файл образа сервиса точки входа в систему
images/gossed_app_v1.tar	Файл образа сервиса legacy back-end для взаимодействия с толстым клиентом
images/gossed_app_v2.tar	Файл образа сервиса actual back-end для взаимодействия с web-клиентом
images/ gossed_auth_service.tar	Файл образа сервиса авторизации
images/gossed_chronos.tar	Файл образа сервисов периодических операций
images/gossed_gateway_service.tar	Файл образа шлюз сервиса
images/gossed_system_notification.tar	Файл образа сервиса системы уведомлений
migration/gossed_foivog.7z	Архив для инициализации миграций в БД

Примечание - В настоящем руководстве в том числе в качестве репозитория будет использоваться диск разработчика, содержащий целевые версии приложений. Источник должен фигурировать в списке /etc/apt/resources.list, пример подключения такого репозитория:

deb cdrom:[OS Astra Linux 1.7.3 1.7_x86-64 DVD]/ 1.7_x86-64 contrib main non-free

Примечание – В состав текущего дистрибутива не включен Р7-Офис. Для использования Р7-Офис, необходимо дополнительно указать целевой IP адрес в docker-compose.yml.

Важно!!! Перед установкой ГосСЭД необходимо получить версию ПО подготовленную РТК=ЦД для конкретной организации.

Необходимо предоставить в РТК-ЦД полное доменное имя (веб адрес по которому будет доступно ПО). Указанное имя должно быть сопоставлено на уровне DNS с IP адресом HAProxy, который будет реализован в процессе установки системы.[1]

[1] Реализация обусловлена используемой микрофронтэнд архитектурой и невозможностью использования компонентов Node.js, не прошедшим тематическое исследование.

2. СИСТЕМНЫЕ ТРЕБОВАНИЯ

2.1. Операционная система

Для установки всех компонентов системы требуется использование дистрибутива ОС Astra Linux. Операционная система должна соответствовать следующим требованиям:

- дистрибутив Astra Linux должен быть установлен и обновлен до версии 1.7 и выше (рассматриваются только 64 битные редакции);
- операционная система должна поддерживать установку и работу с Docker и другими необходимыми компонентами;
- необходимо обеспечить соответствие требованиям по информационной безопасности, установленным для использования в государственных организациях.

Перед началом установки рекомендуется проверить наличие всех необходимых пакетов и обновлений для корректной работы системы.

Примечание – Система может запуститься и работать на других похожих дистрибутивах, но использование их не гарантирует работоспособность, в связи с частными отличиями в процессе установки.

2.2. Система управления базами данных

Для корректной работы системы необходимо использовать следующие базы данных:

- PostgreSQL версия 15.6, рекомендуется использовать последнюю стабильную версию для обеспечения максимальной производительности и безопасности;
- Redis: версия 7.2.5, рекомендуется использовать последнюю стабильную версию для оптимизации работы с кэшированием и очередями сообщений;
- Manticore Search: версия 6.3.6 и старше, рекомендуется использовать последнюю стабильную версию для обеспечения актуальности функционала поиска.

2.3. Требования к версиям остальных компонентов

Для корректной работы системы необходимо использовать следующие компоненты:

- HAProxy: версия 2.2.9;
- MinIO: версия RELEASE.2024-08-03;
- Docker.io: версия 20.10.2;
- RabbitMQ: версия 3.7.8.

2.4. Требования к техническим средствам

Типовой перечень конфигурации серверного оборудования представлен в таблице.

Назначение сервера	Имя	Кол-во ядер	Опер. пямяти	Всего SSD/HDD
Балансировщик	sed-haproxy-01	2	2 GB	42 GB
Manticore	sed-manticore-01	2	4 GB	64 GB
	sed-manticore-02	2	4 GB	64 GB
	sed-manticore-03	2	4 GB	64 GB
Minio	sed-minio-srv01	4	16 GB	96 GB
	sed-minio-srv02	4	16 GB	96 GB
	sed-minio-srv03	4	16 GB	96 GB
	sed-minio-srv04	4	16 GB	96 GB
Сервер СУБД	sed-pgsql-01	2	4 GB	254 GB
	sed-pgsql-node-01	4	6 GB	256 GB
	sed-pgsql-node-02	4	6 GB	256 GB
	sed-pgsql-node-03	4	6 GB	256 GB
Rabbit mq	sed-rabbit-srv01	4	16 GB	56 GB
	sed-rabbit-srv02	4	16 GB	56 GB
	sed-rabbit-srv03	4	16 GB	56 GB
Redis	sed-redis-01	2	4 GB	44 GB
	sed-redis-02	2	4 GB	44 GB
	sed-redis-03	2	4 GB	44 GB
Менеджер Swarm	sed-swarm-man-01	4	8 GB	68 GB
Сервер приложений	sed-swarm-work-01	4	24 GB	84 GB
	sed-swarm-work-02	4	24 GB	84 GB
Сервер службы периодических операций Chronos	sed-swarm-work-03	8	12 GB	72 GB
	sed-swarm-work-04	8	12 GB	72 GB

Важно!!! Перед установкой ГосСЭД необходимо получить версию ПО подготовленную РТК=ЦД для конкретной организации.

Необходимо предоставить в РТК-ЦД полное доменное имя (веб адрес по которому будет доступно ПО). Указанное имя должно быть сопоставлено на уровне DNS с IP адресом HAProxy, который будет реализован в процессе установки системы.[2]

[2] Реализация обусловлена используемой микрофронтэнд архитектурой и невозможностью использования компонентов Node.js, не прошедшим тематическое исследование.

3. УСТАНОВКА DOCKER SWARM

3.1. Установка Docker

3.1.1. Перечень серверов для установки

Установка выполняется на серверах:

sed-swarm-man-01 - 17.20.150.24 (управляющий узел),

sed-swarm-work-01 - 17.20.150.25 (рабочий узел),

sed-swarm-work-02 - 17.20.150.22 (рабочий узел),

sed-swarm-work-03 - 17.20.150.21 (рабочий узел),

sed-swarm-work-04 - 17.20.150.23 (рабочий узел).

3.1.2. Конфигурация системы

Перед началом установки возможно понадобится обновить список доступных репозиториев, для этого в файле `/etc/apt/sources.list` необходимо снять комментарии в указанных строках:

```
repository-base
repository-extended
repository-update
```

После сохранения файла необходимо обновить список пакетов с помощью команды:

```
sudo apt update
```

3.1.3. Порядок установке Docker

Установите пакет docker.io на всех узлах кластера с помощью команды:

```
sudo apt install docker.io
```

Выдайте права текущему пользователю на исполнение команд Docker с помощью команды:

```
sudo usermod -aG docker $USER
```

Для того чтобы изменения вступили в силу, войдите в систему заново с помощью команды:

```
exec su - $USER
```

3.1.4. Проверка установки

Убедитесь, что Docker установлен и работает, выполнив команду:

```
docker --version
```

3.2. Инициализация Docker Swarm

3.2.1. Команда для инициализации Swarm

На управляющем узле (sed-swarm-man-01) выполните команду инициализации:

```
docker swarm init
```

При успешной инициализации будет выведено сообщение с командой для подключения рабочих узлов:ls

```
docker swarm join --token <hash> 17.20.150.24:2377
```

3.2.2. Проверка статуса Swarm

Проверьте статус Swarm, выполнив команду:

```
docker info
```

3.3. Добавление узлов в Swarm

3.3.1. Команды для добавления рабочих узлов

На каждом рабочем узле (sed-swarm-work-01, sed-swarm-work-02, sed-swarm-work-03, sed-swarm-work-04) выполните команду, полученную на управляющем узле:

```
docker swarm join --token <hash> 17.20.150.24:2377
```

3.3.2. Проверка состояния узлов

На управляющем узле выполните команду для проверки состояния узлов:

```
docker node ls
```

4. УСТАНОВКА КЛАСТЕРА MANTICORE SEARCH

4.1. Загрузка и установка Manticore Search

4.1.1. Перечень серверов для установки

Установка выполняется на серверах:

sed-manticore-01 - 17.20.150.30,

sed-manticore-02 - 17.20.150.31,

sed-manticore-03 - 17.20.150.32.

4.1.2. Порядок установки Manticore Search

Для установки Manticore Search необходимо скачать пакеты, выполнив команду:

```
wget
https://repo.manticoresearch.com/repository/manticoresearch_buster/dists/buster/main/
binary-amd64/manticore-server_6.3.6-24080214-593045790_amd64.deb \

https://repo.manticoresearch.com/repository/manticoresearch_buster/dists/buster/main/
binary-amd64/manticore-server-core_6.3.6-24080214-593045790_amd64.deb \

https://repo.manticoresearch.com/repository/manticoresearch_buster/dists/buster/main/
binary-amd64/manticore-columnar-lib_2.3.0-24052206-88a01c3_amd64.deb \

https://repo.manticoresearch.com/repository/manticoresearch_buster/dists/buster/main/
binary-amd64/manticore-backup_1.3.8-24052208-57fc406_all.deb \

https://repo.manticoresearch.com/repository/manticoresearch_buster/dists/buster/main/
binary-amd64/manticore-buddy_2.3.12-24071807-45f6b91_all.deb \

https://repo.manticoresearch.com/repository/manticoresearch_buster/dists/buster/main/
binary-amd64/manticore-executor_1.1.12-24071807-0565a65_amd64.deb \

https://repo.manticoresearch.com/repository/manticoresearch_buster/dists/buster/main/
binary-amd64/manticore-tools_6.3.6-24080214-593045790_amd64.deb \

https://repo.manticoresearch.com/repository/manticoresearch_buster/dists/buster/main/
binary-amd64/manticore-common_6.3.6-24080214-593045790_all.deb \

https://repo.manticoresearch.com/repository/manticoresearch_buster/dists/buster/main/
binary-amd64/manticore_6.3.6-24080214-593045790_amd64.deb \

https://repo.manticoresearch.com/repository/manticoresearch_buster/dists/buster/main/
binary-amd64/manticore-dev_6.3.6-24080214-593045790_all.deb \

https://repo.manticoresearch.com/repository/manticoresearch_buster/dists/buster/main/
binary-amd64/manticore-icudata-65l.deb \

https://repo.manticoresearch.com/repository/manticoresearch_buster/dists/buster/main/
binary-amd64/manticore-galera_3.37_amd64.deb \

https://repo.manticoresearch.com/repository/manticoresearch_buster/dists/buster/main/
binary-amd64/manticore-tzdata_1.0.0-240522-a8aa66e_all.deb
```

Примечание - Если на сервере нет доступа к интернету, необходимо скачать пакеты заранее и передать их на целевой хост.

После загрузки пакетов выполните команду:

```
sudo apt -y update && sudo apt -y install ./*.deb
```

4.1.3. Настройка конфигурации службы

На каждой ноде измените файл конфигурации по пути `/etc/manticoresearch/manticore.conf`, пример конфигурации:

```
searchd {
       listen = <IP_сервера>:9312
       listen = <IP_сервера>:9306:mysql
       listen = <IP_сервера>:9308:http
       listen = <IP_сервера>:9360-9370:replication
       log = /var/log/manticore/searchd.log
       query_log = /var/log/manticore/query.log
       pid_file = /var/run/manticore/searchd.pid
       data_dir = /var/lib/manticore
   }
```

<IP-сервера> - будет различаться в зависимости от хоста, на котором происходит установка.

Выполните последовательность указанных команд на каждом сервере (п. 4.1.1).

4.2. Запуск Manticore Search

4.2.1. Команды для запуска Manticore Search

После внесения изменений в конфигурацию перезапустите сервис Manticore Search, выполнив команду:

```
sudo systemctl restart manticore
```

4.2.2. Проверка работы сервиса

Чтобы проверить статус сервиса, выполните команду:

```
sudo systemctl status manticore
```

4.2.3. Настройка кластера

Реализация кластера построена на репликации баз данных, для этого необходимо подключится любым MySQL-клиентом к любому серверу с Manticore Search через порт 9306 и создать таблицу rt-типа, выполнив команду:

```
CREATE TABLE gossed;
```

Создайте кластер и добавьте к нему таблицу с помощью команды:

```
CREATE CLUSTER gossed;
ALTER CLUSTER gossed add gossed;
```

Поочередно подключайтесь через MySQL-клиент к остальным серверам с Manticore Search и выполняйте команду:

```
JOIN cluster gossed AT '<IP_сервера_где_создавался_кластер>:9312';
```

4.2.4. Проверка статуса кластера

Для проверки статуса кластера выполните запрос через MySQL клиент, выполнив команду:

```
SHOW STATUS LIKE '%cluster%';
```

5. УСТАНОВКА КЛАСТЕРА REDIS

5.1. Загрузка и установка Redis

5.1.1. Перечень серверов для установки

Установка выполняется на серверах:

sed-redis-01 - 17.20.150.37,

sed-redis-02 - 17.20.150.38,

sed-redis-03 - 17.20.150.39.

5.1.2. Конфигурация системы

Перед началом установки необходимо обновить список доступных репозиториев в `/etc/apt/sources.list`, для этого снять комментарии в строках:

```
repository-base
repository-extended
repository-update
```

После сохранения файла обновите список пакетов, выполнив команду:

```
sudo apt update
```

- Для подготовки к установке необходимо установить пакеты для сборки ПО из исходников:

```
sudo apt install make gcc libc6-dev tcl
```

Для корректной работы сервиса необходимо разрешить `overcommit` памяти. В противном случае сервис может дать сбой при недостатке памяти на сервере. Выполните команду:

```
sudo sed -i '1 i\vm.overcommit_memory=1' /etc/sysctl.conf
sudo sysctl -p
```

5.1.3. Порядок установки Redis

Скачать с сайта Redis архив с актуальной версией. В настоящем руководстве используется версия redis-7.2.5. Распакуйте архив с исходниками, выполнив команду:

```
tar xvzf redis-7.2.5.tar.gz
```

Перейдите в директорию и выполните команду для сборки Redis:

```
cd redis-7.2.5
sudo make install
```

После завершения установки проверьте валидность установки с помощью команды:

```
make test
```

При успешном завершении в консоли отображается ответ:

```
\o/ All tests passed without errors!
```

Выполните последовательность указанных команд на каждом сервере (п. 5.1.1).

5.1.4. Настройка конфигурации кластера

Установка кластера выполняется в формате 3 master и 3 slave, на базе 3 серверов, схема установки кластера представлена на рисунке (Рисунок 1). На схеме отображается расположение master и slave на серверах и их зависимость друг от друга.

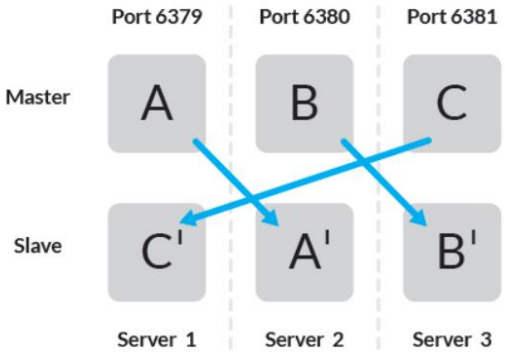

Рисунок 1

Номера используемых портов перечислены в таблице (Таблица 2).

Таблица 2 - Номера используемых портов

Сервер	Master	Slave
sed-redis-01	6379	6381
sed-redis-02	6380	6379
sed-redis-03	6381	6380

Примечание - Здесь и далее работа производится в каталоге redis-7.2.5.

Создайте файлы конфигураций master и slave на всех серверах кластера. Наименование конфигурационных файлов осуществляется в соответствии с таблицей (Таблица 3).

Таблица 3 - Наименование конфигурационных файлов

sed-redis-01
`cp redis.conf c_slave.conf` `mv redis.conf a_master.conf`
sed-redis-02
`cp redis.conf a_slave.conf` `mv redis.conf b_master.conf`
sed-redis-03
`cp redis.conf b_slave.conf` `mv redis.conf c_master.conf`

Внесите изменения в master конфигурации в соответствии с таблицей (Таблица 4).

Таблица 4 - Изменения в master конфигурации

a_master.conf
bind 0.0.0.0 protected-mode no port 6379 pidfile /var/run/redis_6379.pid cluster-enabled yes cluster-config-file nodes-6379.conf cluster-node-timeout 15000
b_master.conf
bind 0.0.0.0 protected-mode no port 6380 pidfile /var/run/redis_6380.pid cluster-enabled yes cluster-config-file nodes-6380.conf cluster-node-timeout 15000
c_master.conf
bind 0.0.0.0 protected-mode no port 6381 pidfile /var/run/redis_6381.pid cluster-enabled yes cluster-config-file nodes-6381.conf cluster-node-timeout 15000

Внесите изменения в slave конфигурации в соответствии с таблицей (Таблица 5).

Таблица 5 - Изменения в slave конфигурации

c_slave.conf
bind 0.0.0.0 protected-mode no port 6381 pidfile /var/run/redis_6381.pid cluster-enabled yes cluster-config-file nodes-6381.conf cluster-node-timeout 15000
a_slave.conf
bind 0.0.0.0 protected-mode no port 6379 pidfile /var/run/redis_6379.pid cluster-enabled yes cluster-config-file nodes-6379.conf cluster-node-timeout 15000
b_slave.conf
bind 0.0.0.0 protected-mode no port 6380 pidfile /var/run/redis_6380.pid cluster-enabled yes cluster-config-file nodes-6380.conf cluster-node-timeout 15000

5.2. Запуск Redis кластере

5.2.1. Команды для запуска Redis в кластере

Запустите все компоненты на всех серверах, где N это префикс, который соответствует схеме конфигурации, описанной ранее, с помощбю команд:

```
sudo redis-server N_master.conf --daemonize yes
sudo redis-server N_slave.conf --daemonize yes
```

Примечание - Ранее сборка кластера производилась через специальную утилиту написанную на Ruby. В актуальных версиях Redis для сборки используется `redis-cli`.

На любой из нод будущего кластера выполните сборку кластера с указанием master серверов, выполнив команду:

```
redis-cli --cluster create <ip redis-01>:6379 <ip redis-02>:6380 <ip redis-
03>:6381
```

Примечание - Указываются порты из master конфигураций

Далее добавьте slave ноды в соответствии со схемой и конфигурациями по шаблону:

```
redis-cli --cluster add-node <SLAVE IP>:<SLAVE PORT> <MASTER IP>:<MASTER
PORT> --cluster-slave
```

Выполните команду три раза для создания трех связей между тремя master и тремя slave:

```
redis-cli --cluster add-node <redis-02 IP>:6379 <redis-01 IP>:6379 --cluster-
slave
redis-cli --cluster add-node <redis-03 IP>:6380 <redis-02 IP>:6380 --cluster-
slave
redis-cli --cluster add-node <redis-01 IP>:6381 <redis-03 IP>:6381 --cluster-
slave
```

Примечание – Обратите внимание, что у master и slave совпадает порт.

После успешного выполнения команд создание кластера завершено.

5.2.2. Проверка работы сервиса

Для проверки кластера подключитесь к любому master серверу с помощью консольной утилиты `redis-cli`:

```
redis-cli -c -h <redis-01 IP> -p 6379
```

Для запроса у redis списка нод кластера выполните команду:

```
CLUSTER NODES
```

В случае если кластер работает корректно, в консоль будет выведен список из шести элементов: три master и три slave сервера.

Для выхода из утилиты `redis-cli` выполните команду `exit`

Примечание - Остановка сервиса. Для удобства работы желательно установить утилиту htop:

```
sudo apt install htop
```

- вызов выполняется через консоль htop,
- найти pid родительского процесса redis (в стандартной цветовой схеме htop он будет белого цвета, дочерние зеленого),

  завершается процесс командой kill -9 <pid процесса>

6. УСТАНОВКА КЛАСТЕРА MINIO

6.1. Загрузка и установка MinIO

6.1.1. Перечень серверов для установки

Установка выполняется на серверах:

sed-minio-01 - 17.20.150.33,

sed-minio-02 - 17.20.150.34,

sed-minio-03 - 17.20.150.35,

sed-minio-04 - 17.20.150.36.

6.1.2. Конфигурация дисков и системы

На каждом сервере дополнительно подключен диск для формирования хранилища данных и сепарации от системного диска. Ниже представлен пример конфигурации диска.

Для начала необходимо отобразить подключеные диски, выполнив команду и получив отображение результата ее выполнения:

```
sudo lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
loop0     7:0    0  89M  1 loop /snap/core/7713
loop1     7:1    0  18M  1 loop /snap/amazon-ssm-agent/1480
xvda    202:0    0   8G  0 disk
└─xvda1 202:1    0   8G  0 part /
xvdb    202:16   0  40G  0 disk
```

Следует смотреть информацию для диска `xvdb`. Создайте в корне файловой системы каталог `/data` и смонтируйте в него диск `xvdb`, предварительно отформатировав в файловую систему XFS (рекомендация вендора), с помощью команд:

```
sudo su
mkdir /data
mkfs.xfs /dev/xvdb
echo "/dev/xvdb /data xfs defaults,noatime,nofail 0 0" >> /etc/fstab
mount -a
```

Убедитесь, что диск смонтирован, выполнив команды:

```
df -h
Filesystem      Size  Used  Avail Use% Mounted on
/dev/xvda1      7.7G  1.3G  6.5G  17%  /
/dev/xvdb       40G   53M   40G   1%   /data
```

Отредактируйте файл `host с помощью команды:`

```
cat > /etc/hosts << EOF
127.0.1.1       sed-minio-01
17.20.150.33    sed-minio-01
17.20.150.34    sed-minio-02
17.20.150.35    sed-minio-03
17.20.150.36    sed-minio-04
```

```
EOF
```

Выполните последовательность описанных действий на каждом сервере (п.6.1.1), первая строчка должна содержать имя самого сервера, на котором проводится настройка.

Примечание – Так как ранее использовалась команда `sudo su`, то далее команды выполняются от учетной записи `root`.

6.1.3. Порядок установки MinIO

Откройте файл `/etc/apt/sources.list` для редактирования, выполнив команду:

```
nano /etc/apt/sources.list
```

В файле `/etc/apt/sources.list` найдите строку:

```
# deb https://download.astralinux.ru/astra/stable/1.7_x86-64/repository-
extended/ 1.7_x86-64 main contrib non-free
```

Снимите комментарий с найденной строки, удалив символ `#` в начале строки.

Сохраните изменения в файле (последовательно нажмите комбинацию клавиш [Ctrl + O], [Enter], и для выхода [Ctrl + X]).

Обновите список пакетов с помощью команды:

```
apt update
```

Установите пакет MinIO, выполнив команду:

```
apt install minio –y
```

6.1.4. Настройка конфигурации кластера

Создайте файл конфигурации для каждого сервера. Например, для `sed-minio-01`:

```
nano /etc/default/minio
```

Вставьте следующие строки, изменяя `MINIO_ACCESS_KEY` и `MINIO_SECRET_KEY` на ваши значения:

```
MINIO_OPTS="http://sed-minio-01:9000/data http://sed-minio-02:9000/data
http://sed-minio-03:9000/data http://sed-minio-04:9000/data"
MINIO_ACCESS_KEY="AKaHEgQ4II0S7BjT6DjAUDA4Fx"
MINIO_SECRET_KEY="SKFzHq5iDoQgF7gyPYRFhzNMYSvY6ZFMHp"
```

Повторите этот шаг для остальных серверов (п. 6.1.1).

Примечание – Строки с ключами придирчивы к сложности, поэтому необходимо соблюдать это условие, чтобы сервисы запустились без ошибок, в иных случаях работа сервиса не гарантируется.

-

6.2. Запуск MinIO

6.2.1. Команды для запуска MinIO в кластере

Для каждого сервера выполните команду:

```
systemctl start minio
```

Настройте MinIO на автоматический запуск при загрузке с помощбю команды:

```
systemctl enable minio
```

6.2.2. Проверка работы сервиса

Проверьте статус MinIO на каждом сервере, выполнив команду:

```
systemctl status minio
```

Проверьте доступность MinIO через браузер, для этого откройте браузер и перейдите по адресу `http://<IP-адрес-сервера>:9000`, например:

- `http://17.20.150.33:9000`,
- `http://17.20.150.34:9000`,
- `http://17.20.150.35:9000`,
- `http://17.20.150.36:9000`.

6.2.3. Создание Bucket и учетной записи

Войдите в интерфейс MinIO с использованием `MINIO_ACCESS_KEY` и `MINIO_SECRET_KEY`, которые указаны в конфигурации.

Нажмите на кнопку `Create Bucket` (Рисунок 2).

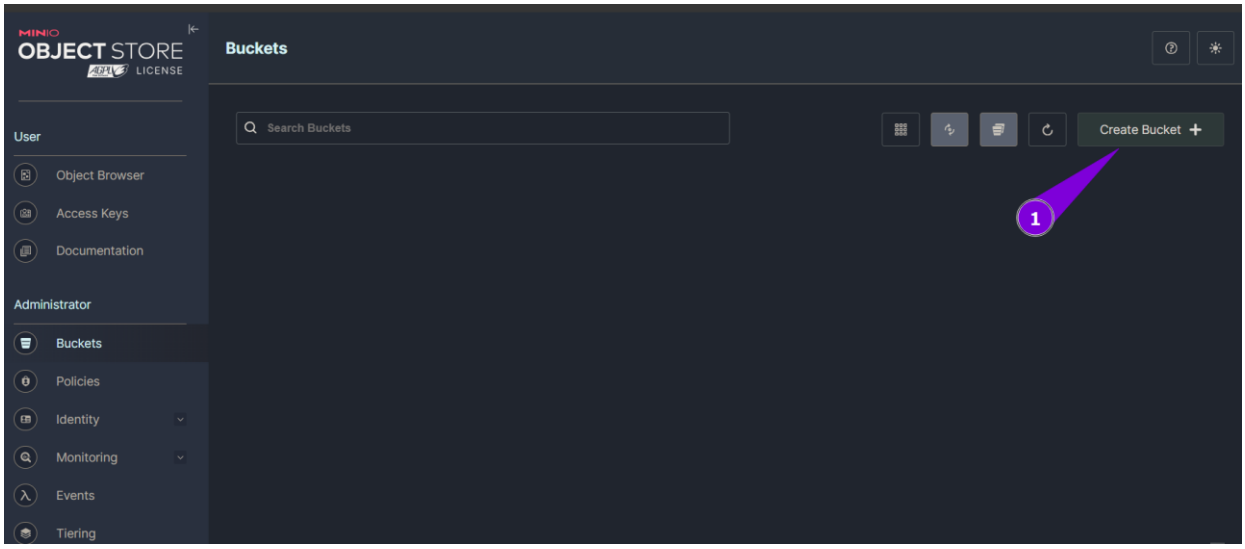

Рисунок 2

Укажите имя создаваемого `bucket`'а, например `sed-bucket`, и нажмите `Create Bucket` (Рисунок 3).

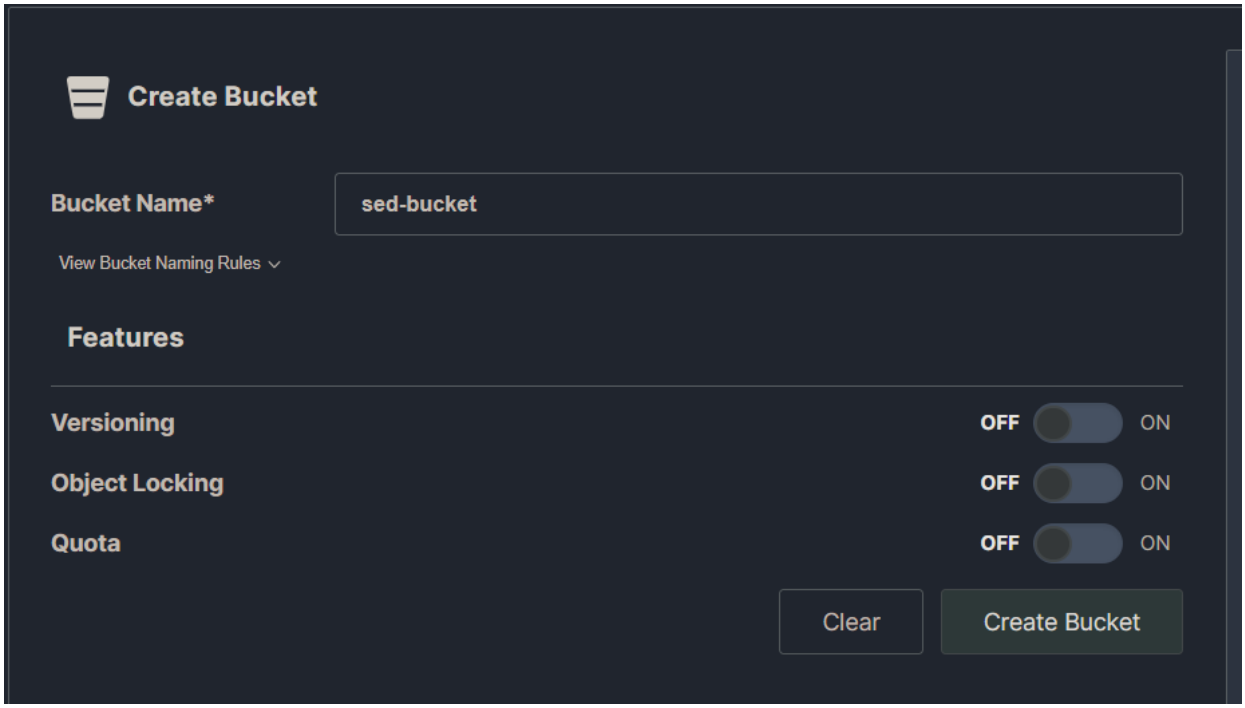

Рисунок 3

Для создания пользователя перейдите в следующий раздел и нажмите `Create User` (Рисунок 4).

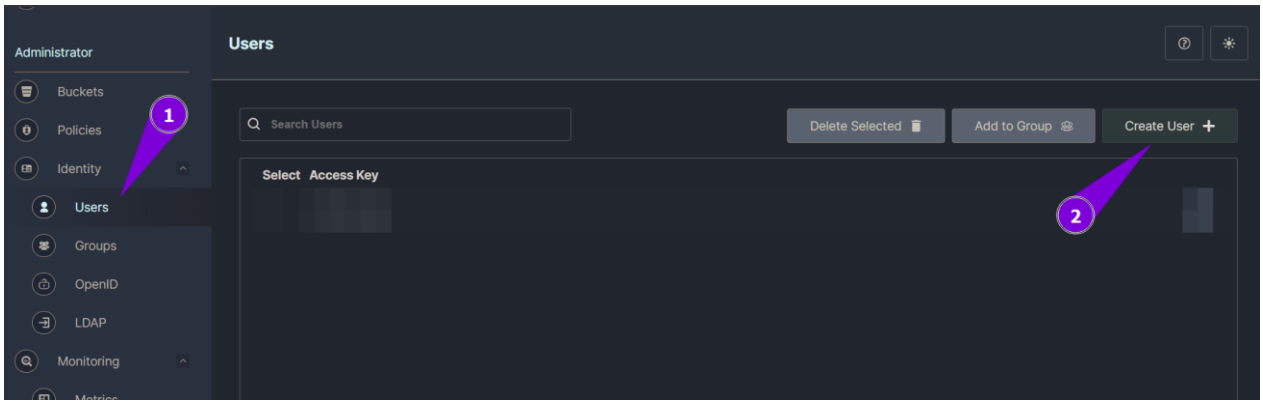

Рисунок 4

В поле `User Name` введите имя пользователя, укажите уровень доступа `readwrite`, сохраните изменения, нажав кнопку `Save` (Рисунок 5).

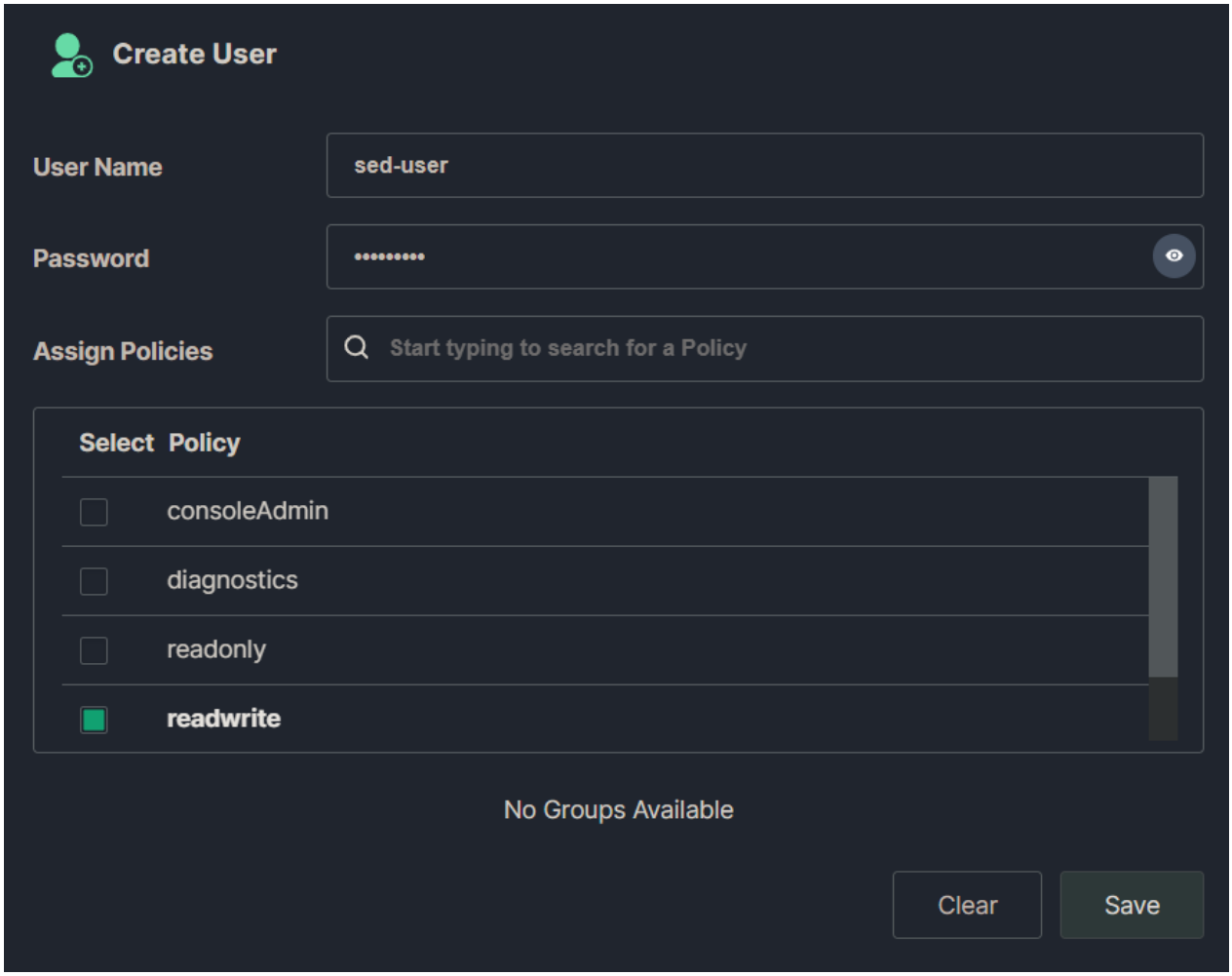

Рисунок 5

После создания откройте свойства пользователя и создайте `Access Key` (Рисунок 6).

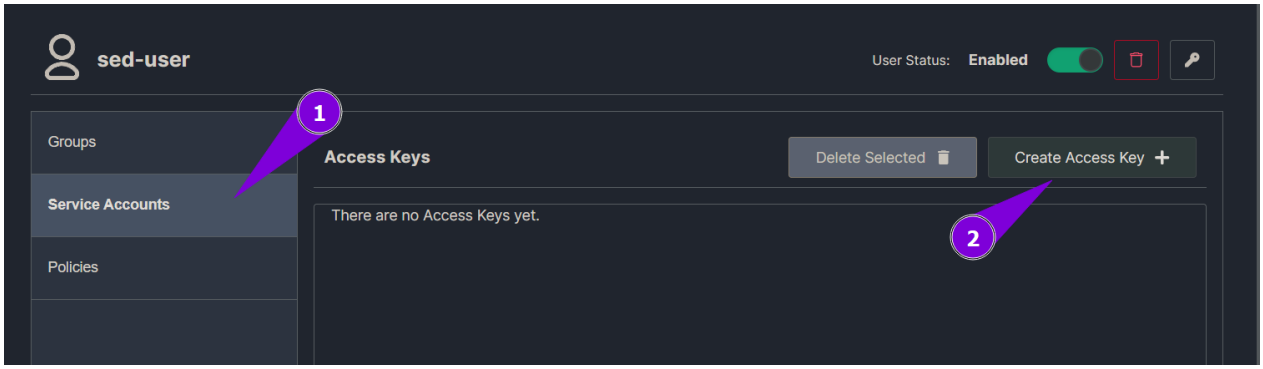

Рисунок 6

В открывшемся окне для сохранения реквизитов учетной записи нажмите `Download for import` (Рисунок 7).

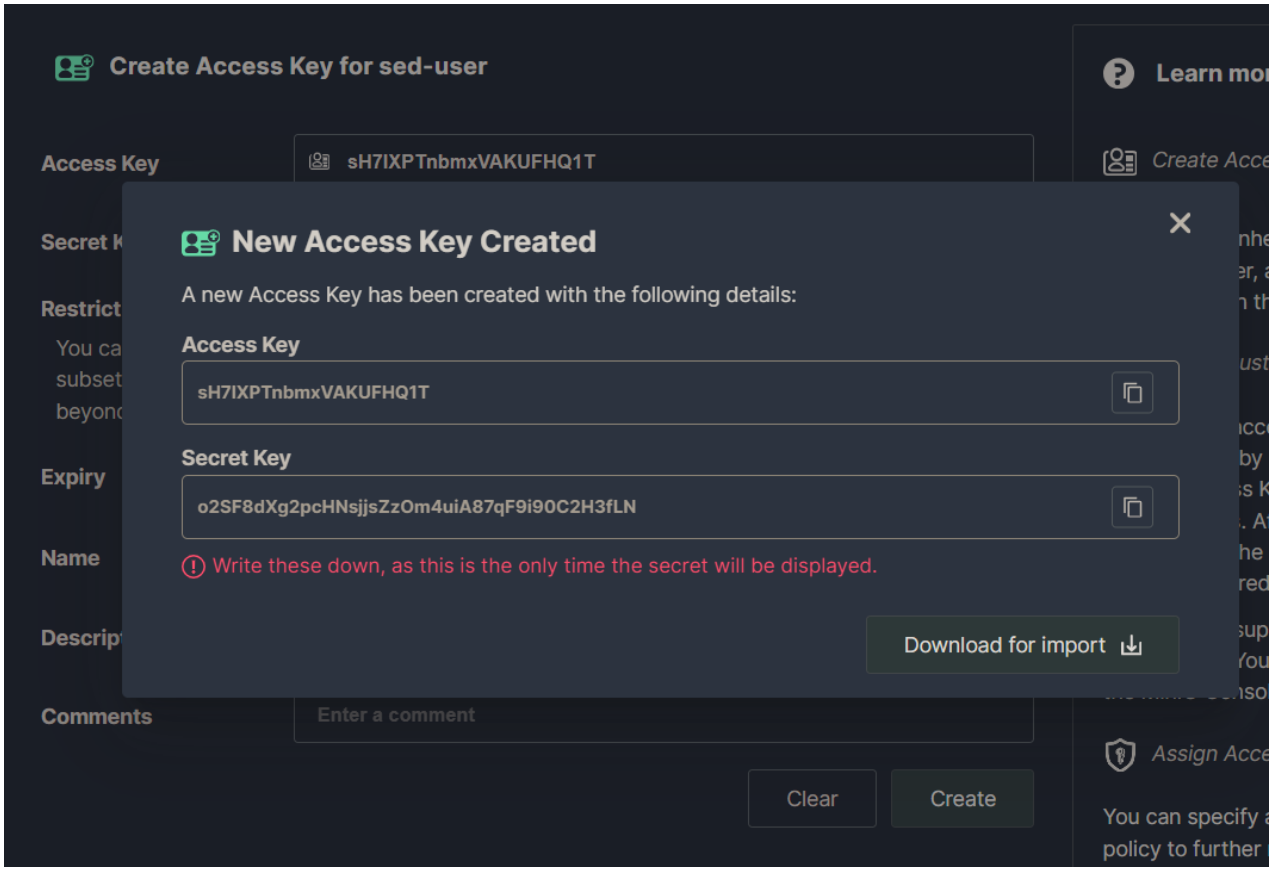
Create Access Key for sed-user
Access Key
sH7IXPTnbmxVAKUFHQ1T
Secret K
Restrict
You ca
subset
beyonc
Expiry
Name
Descrip
Comments
Enter a comment
Clear
Create
Learn mo
Create Acce
Assign Acce
You can specify
policy to further
New Access Key Created
A new Access Key has been created with the following details:
Access Key
sH7IXPTnbmxVAKUFHQ1T
Secret Key
o2SF8dXg2pcHNsjjsZzOm4uiA87qF9i90C2H3fLN
Write these down, as this is the only time the secret will be displayed.
Download for import

Рисунок 7

7. УСТАНОВКА КЛАСТЕРА RABBITMQ

7.1. Загрузка и установка RabbitMQ

7.1.1. Перечень серверов для установки

Установка выполняется на серверах:

sed-rabbit-01 - 17.20.150.26 (master),

sed-rabbit-02 - 17.20.150.27 (slave),

sed-rabbit-03 - 17.20.150.28 (slave).

7.1.2. Порядок установки RabbitMQ

Отредактируйте файл `/etc/hosts`, добавив значения ниже, для каждого сервера первая строчка будет содержать имя самого сервера, на котором проводится настройка:

```
127.0.1.1       sed-rabbit-01
17.20.150.26    sed-rabbit-01
17.20.150.27    sed-rabbit-02
17.20.150.28    sed-rabbit-03
```

Убедитесь, что в файле `/etc/apt/sources.list` присутствует строка для репозитория RabbitMQ. Если необходимо, добавьте репозиторий, выполнив команду:

```
echo "deb cdrom:[OS Astra Linux 1.7.3 1.7_x86-64 DVD ]/ 1.7_x86-64 contrib
main non-free" | sudo tee -a /etc/apt/sources.list
```

Обновите список пакетов с помощью команды:

```
sudo apt-get update
```

Установите RabbitMQ с помощью команды:

```
sudo apt-get install rabbitmq-server=3.7.8
sudo systemctl start rabbitmq-server
sudo systemctl enable rabbitmq-server
```

Убедитесь, что сервис запущен и работает без ошибок, выполнив команду:

```
sudo systemctl status rabbitmq-server
```

7.1.3. Настройка конфигурации кластера

Включите плагины управления с помощью команд:

```
sudo rabbitmq-plugins enable rabbitmq_management
sudo systemctl restart rabbitmq-server
```

Скопируйте `/var/lib/rabbitmq/.erlang.cookie` с master-сервера на slave-сервер (если в процессе переноса изменялись права на файл, необходимо его привести к изначальному виду), выполнив команды:

```
sudo chmod 400 /var/lib/rabbitmq/.erlang.cookie
sudo chown rabbitmq:rabbitmq /var/lib/rabbitmq/.erlang.cookie
```

7.2. Запуск RabbitMQ

7.2.1. Команды для запуска RabbitMQ в кластере

Для присоединения slave-сервера к с master-серверу выполните команды последовательно на slave-серверах:

```
sudo rabbitmqctl stop_app
sudo rabbitmqctl join_cluster
sudo rabbitmqctl start_app
```

7.2.2. Проверка работы сервиса

На всех slave-серверах проверьте вхождение в кластер, выполнив на master-сервере команду:

```
sudo rabbitmqctl cluster_status
```

7.2.3. Добавление администратора

Добавьте пользователя командой, на мастер сервере:

```
sudo rabbitmqctl add_user sedadmin sedslozhno24!
```

Делегируйте права администратора и раздайте разрешения с помощью команд:

```
sudo rabbitmqctl set_user_tags sedadmin administrator
sudo rabbitmqctl set_permissions -p / sedadmin "." "." ".*"
```

Удалите пользователя по умолчанию с помощью команды:

```
sudo rabbitmqctl delete_user guest
```

Добавьте политики зеркалирования «все на все», выполнив команду:

```
sudo rabbitmqctl set_policy ha-all ".*" '{"ha-mode":"all"}'
```

8. УСТАНОВКА КЛАСТЕРА POSTGRESQL

8.1. Загрузка и установка компонентов кластера

8.1.1. Перечень серверов для установки

Установка выполняется на серверах:

sed-pgsql-01 - 17.20.150.41,

sed-pgsql-02 - 17.20.150.42,

sed-pgsql-03 - 17.20.150.43.

8.1.2. Порядок установки PostgreSQL

Включите репозиторий в файле `/etc/apt/sources.list`. Версия, используемая в настоящем руководстве, устанавливается из диска разработчика:

```
deb cdrom:[OS Astra Linux 1.7.3 1.7_x86-64 DVD ]/ 1.7_x86-64 contrib main non-free
```

Обновите список пакетов командой:

```
sudo apt update
```

Установите PostgreSQL с указанием целевой версии с помощью команды:

```
sudo apt install postgresql-15=15.6-astra.se2
```

Проверьте что сервис работает без ошибок, выполнив команду:

```
sudo systemctl status posgresql
```

Отключите автоматический запуск PostgreSQL (управление службой будет выполняться другой утилитой), выполнив команду:

```
sudo systemctl disable --now postgresql
```

Выполните установку и настройку службы PostgreSQL на каждом сервере (п. 8.1.1).

8.1.3. Порядок установки Etcd

Подключите дополнительные репозитории Астры. Откройте файл списка репозиториев командой:

```
sudo nano /etc/apt/sources.list
```

В файле списка репозиториев закомментируйте установочный диск Астры и добавьте следующие репозитории:

```
# Основной репозиторий deb https://dl.astralinux.ru/astra/stable/1.8_x86-64/repository-main/ 1.8_x86-64 main contrib non-free
```

```
# Расширенный репозиторий
deb https://dl.astralinux.ru/astra/stable/1.8_x86-64/repository-extended/
1.8_x86-64 main contrib non-free
```

Обновите список пакетов командой:

```
sudo apt update
```

Установите Etcd, выполнив команду:

```
sudo apt install -y etcd-server
```

8.1.4. Настройка конфигурации Etcd

Настройте Etcd, редактируя конфигурационный файл следующей командой:

```
sudo nano /etcd/default/etcd
```

Вставьте следующий текст, заменив параметры на имя и адрес в соответствии с сервером, на котором производится настройка:

```
ETCD_NAME=<hostname_node>
ETCD_INITIAL_CLUSTER_TOKEN="devops_token"
ETCD_INITIAL_CLUSTER_STATE="new"
ETCD_INITIAL_ADVERTISE_PEER_URLS="http://<IP_node>:2380"
ETCD_DATA_DIR="/var/lib/etcd/postgresql"
ETCD_LISTEN_PEER_URLS="http://<IP_node>:2380"
ETCD_LISTEN_CLIENT_URLS="http:// <IP_node>:2379,http://127.0.0.1:2379"
ETCD_ADVERTISE_CLIENT_URLS="http://<IP_node>:2379,http://127.0.0.1:2379"
ETCD_INITIAL_CLUSTER="node1=http://17.20.150.41:2380,node2=http://17.20.150.4
3:2380,node3=http://17.20.150.42:2380"
```

Перезагрузите службу Etcd, выполнив команду:

```
sudo systemctl restart etcd
```

Выполните установку и настройку на каждом сервере (п. 8.1.1).

Проверьте состояние каждой ноды через веб-интерфейс, указав в адресной строке браузера соответствующий адрес:

```
http://<IP_node>:2379/health
```

Ожидайте ответа:

```
{"health":"true"}
```

8.1.5. Порядок установки Patroni

Установите Patroni командой:

```
sudo apt install -y patroni
```

Проверьте состояние сервиса Patroni, выполнив команду:

```
sudo systemctl status patroni
```

Убедитесь, что он не работает, и что по умолчанию установлен конфигурационный файл `/etc/patroni/config.yml`.

8.1.6. Настройка конфигурации Patroni

Создайте директорию для данных и файла паролей с помощью команд:

```
mkdir -p /data/patroni
chown postgres:postgres /data/patroni
chmod 700 /data/patroni
```

Создайте файл конфигурации командой:

```
nano /etc/patroni/config.yml
```

Вставьте следующую конфигурацию, заменив имя хоста, IP-адреса и пароли, в соответствии с сервером на котором производится установка:

```
scope: sed-pgsql-cluster
   namespace: /cluster/
   name: <hostname_node>

   restapi:
       listen: <IP_node>:8008
       connect_address: <IP_node>:8008

   etcd:
       hosts: 172.20.150.41:2379,172.20.150.43:2379,172.20.150.42:2379

   bootstrap:
       dcs:
           ttl: 100
           loop_wait: 10
           retry_timeout: 10
           maximum_lag_on_failover: 1048576
           postgresql:
               use_pg_rewind: true
               use_slots: true
               parameters:
                       wal_level: replica
                       hot_standby: "on"
                       wal_keep_segments: 5120
                       max_wal_senders: 5
                       max_replication_slots: 5
                       checkpoint_timeout: 30

       initdb:
       - encoding: UTF8
       - data-checksums
       - locale: en_US.UTF8
       pg_hba:
       - host replication postgres ::1/128 md5
       - host replication postgres 127.0.0.1/8 md5
       - host replication postgres 172.20.150.41/24 md5
       - host replication postgres 172.20.150.43/24 md5
       - host replication postgres 172.20.150.42/24 md5
       - host all all 0.0.0.0/0 md5

       users:
           postgres:
```

```
                password: postgres
                options:
                    - createrole
                    - createdb

    postgresql:
        listen: <IP_node>:5432
        connect_address: <IP_node>:5432
        data_dir: /data/patroni
        bin_dir:  /usr/lib/postgresql/15/bin
        pgpass: /tmp/pgpass
        authentication:
            replication:
                username: postgres
                password: postgres
            superuser:
                username: postgres
                password: postgres
        create_replica_methods:
            basebackup:
                checkpoint: 'fast'
        parameters:
            unix_socket_directories: '.'

    tags:
        nofailover: false
        noloadbalance: false
        clonefrom: false
        nosync: false
```

Перезагрузите Patroni командой:

```
sudo systemctl restart patroni
```

Выполните установку и настройку на каждом сервере (п. 8.1.1).

Проверьте состояние кластера через веб-интерфейс, убедитесь, что все ноды имеют один **timeline**, как минимум одна нода имеет статус **role:Leader** и **lag:0**, для этого укажите в адресной строке браузера соответствующие адреса:

```
http://<IP_node>:8008/cluster
```

Примечание - Если вывод состояния кластера не отражает ожидаемых параметров, проверьте статус через 10 минут, есть вероятность, что сервис не завершил процесс синхронизации.

8.1.7. Порядок установки HAProxy

Установите HAProxy, запустив команду:

```
sudo apt install -y haproxy
```

8.1.8. Настройка конфигурации HAProxy

Отредактируйте конфигурационный файл HAProxy, выполните команду:

```
nano /etc/haproxy/haproxy.cfg
```

В конфигурационный файл HAProxy вставьте следующую конфигурацию:

```
global
       maxconn 100

   defaults
       log global
       mode tcp
       retries 2
       timeout client 30m
       timeout connect 4s
       timeout server 30m
       timeout check 5s

   listen stats
       mode http
       bind *:7000
       stats enable
       stats uri /

   listen postgres
       bind *:5000
       option httpchk
       http-check expect status 200
       default-server inter 3s fall 3 rise 2 on-marked-down shutdown-sessions
       server sed-pgsql-01 17.20.150.41:5432 maxconn 100 check port 8008
       server sed-pgsql-02 17.20.150.43:5432 maxconn 100 check port 8008
       server sed-pgsql-03 17.20.150.42:5432 maxconn 100 check port 8008
```

Перезагрузите HAProxy командой:

```
sudo service haproxy restart
```

Выполните установку и настройку на каждом сервере (п. 8.1.1).

8.1.9. Проверка работы сервиса

После настройки HAProxy, появляется единая точка входа в кластер на порту 5000, подключаться к БД нужно именно по нему.

Подключаться можно на любую из нод, HAProxy автоматически перенаправит на лидера кластера. Также на порту 7000 есть веб-интерфейс, с которого можно посмотреть текущего лидера кластера вручную.

9. УСТАНОВКА HAPROXY

9.1. Загрузка и установка HAProxy

9.1.1. Перечень серверов для установки

Установка выполняется на сервере:

sed-haproxy-01 - 17.20.150.29.

9.1.2. Порядок установки HAProxy

Отредактируйте файл /etc/hosts, добавив значения ниже:

```
127.0.1.1       sed-haproxy-01
17.20.150.21    sed-swarm-work-03
17.20.150.22    sed-swarm-work-02
17.20.150.25    sed-swarm-work-01
17.20.150.23    sed-swarm-work-04
17.20.150.24    sed-swarm-man-01
```

В файл `/etc/apt/sources.list` включите репозиторий (версия, используемая в натоящем руководстсве, устанавливается из диска разработчика) командой:

```
deb cdrom:[OS Astra Linux 1.7.3 1.7_x86-64 DVD ]/ 1.7_x86-64 contrib main
non-free
```

Устанавите пакет HAProxy из репозитория Astra Linux командой:

```
sudo apt install haproxy
```

Создайте каталог для SSL сертификатов, выполнив команду:

```
sudo mkdir /etc/haproxy/ssl
```

Скопируйте сертификаты в формате PEM с ключом в созданный каталог, выполнив команду:

```
sudo cp *.pem /etc/haproxy/ssl/
```

9.1.3. Настройка конфигурации

Отредактируйте файл конфигурации HAProxy для маршрутизации нагрузки с помощью команды:

```
sudo nano /etc/haproxy/haproxy.cfg
```

В конфигурационный файл HAProxy вставьте следующую конфигурацию:

```
global
        maxconn 2000000
        log /dev/log    local0
```

```
        log /dev/log    local1 notice
        chroot /var/lib/haproxy
        stats socket /run/haproxy/admin.sock mode 660 level admin
        stats timeout 30s
        user haproxy
        group haproxy
        daemon


defaults
    mode http
    timeout connect 5s
    timeout client 30m
    timeout server 30m

frontend app_root
    bind :80
    bind :443 ssl crt /etc/haproxy/ssl/rtk_crt.pem


    acl fat_client_url path -i -m beg /gossed/
    use_backend fat_client if fat_client_url

    acl web2_client_url path -i -m beg /web2/
    use_backend web2_client if web2_client_url

    default_backend app_root

frontend app_auth
    bind :9002 ssl crt /etc/haproxy/ssl/rtk_crt.pem
    default_backend app_auth


frontend system_notification
    bind :9006 ssl crt /etc/haproxy/ssl/rtk_crt.pem
    default_backend system_notification


frontend design_system
    bind :4000 ssl crt /etc/haproxy/ssl/rtk_crt.pem
    default_backend design_system

frontend auth_service
    bind :7276 ssl crt /etc/haproxy/ssl/rtk_crt.pem
    default_backend auth_service

frontend gateway_service
    bind :7278 ssl crt /etc/haproxy/ssl/rtk_crt.pem
    default_backend gateway_service

frontend stats
    mode http
    bind *:8404
    stats enable
    stats uri /stats
    stats refresh 10s

peers sticktables
    bind :10000
    server sed-haproxy-01
    table sticky-sessions type ip size 1m



backend fat_client
     mode http
     balance roundrobin
     cookie SERVER insert indirect nocache
     server app_root1 17.20.150.22:20100 check weight 50
     server app_root2 17.20.150.25:20100 check weight 50
```

```
backend web2_client
     mode http
     balance roundrobin
     cookie SERVER insert indirect nocache
     server app_root1 17.20.150.22:20200 check weight 50 ssl verify none
     server app_root2 17.20.150.25:20200 check weight 50 ssl verify none

backend app_root
     mode http
     cookie SERVER insert indirect nocache
     balance roundrobin
     server app_root1 17.20.150.22:20301 check weight 50
     server app_root2 17.20.150.25:20301 check weight 50

backend app_auth
     mode http
     balance roundrobin
     server app_auth1 17.20.150.22:20300 check weight 50
     server app_auth2 17.20.150.25:20300 check weight 50

backend system_notification
     mode http
     balance roundrobin
     server system_notification1 17.20.150.22:20303 check weight 50
     server system_notification2 17.20.150.25:20303 check weight 50

backend design_system
     mode http
     balance roundrobin
     server design_system1 17.20.150.22:20302 check weight 50
     server design_system2 17.20.150.25:20302 check weight 50

backend auth_service
     mode http
     balance roundrobin
     cookie SERVER insert indirect nocache
     server auth_service1 17.20.150.22:7276 check weight 50 ssl verify none
     server auth_service2 17.20.150.25:7276 check weight 50 ssl verify none

backend gateway_service
     mode http

     balance roundrobin
     cookie SERVER insert indirect nocache

     server gateway_service1 17.20.150.22:7278 check weight 25 ssl verify none
     server gateway_service2 17.20.150.21:7278 check weight 25 ssl verify none
     server gateway_service3 17.20.150.23:7278 check weight 25 ssl verify none
     server gateway_service4 17.20.150.25:7278 check weight 25 ssl verify none
     server gateway_service_man 17.20.150.24:7278 check weight 25 ssl verify none
```

Примечание – указанная конфигурация содержит настройку sticky-sessions по IP адресу - необходимое условие, чтобы логика приложения штатно выполняла свою работу.

9.2. Запуск HAProxy

9.2.1. Команды для запуска HAProxy

После обновления конфигурации проверьте валидность командой:

```
sudo haproxy -c -f /etc/haproxy/haproxy.cfg
```

Перезагрузите конфигурацию командой:

```
sudo systemctl reload haproxy
```

9.2.2. Проверка работы сервиса

Убедитесь, что сервис HAProxy работает в штатном режиме, командой:

```
sudo systemctl status haproxy
```

10. УСТАНОВКА ПЛАТФОРМЫ

10.1. Загрузка и установка образов

10.1.1. Команды для публикации образов

Загрузите образы из архива, чтобы загрузить образ из локального архива (файла `.tar`), используйте команду на всех серверах swarm группировки:

```
docker load -i /path/to/your/image.tar
```

Примечание - Файлы актуальных образов системы можно будет получить от инженерной поддержки РТК-ЦД.

После загрузки образа можно проверить, что он доступен, с помощью команды:

```
docker images
```

Загруженный образ должен отображаться в списке.

10.2. Запуск установки

Перед запуском установки системы, на сервере sed-swarm-man-01 необходимо разместить основной файл решения docker-compose.yml, через который будут подняты основные контейнеры, обеспечивая взаимодействие компонент и первичную настройку базы данных;

10.2.1. Команды для запуска контейнеров

Перейдите в директорию, в которой размещаются файлы docker-compose.yml, и запустите команду:

```
docker stack up -c docker-compose.yml sed_stack
```

В процессе выполнения запускаются контейнеры ключевых сервисов:

- gossed_app_auth,
- gossed_app_root,
- gossed_app_v1,
- gossed_app_v2,
- gossed_auth_service,
- gossed_chronos,
- gossed_gateway_service,
- gossed_system_notification.

10.2.2. Выполнение миграции

Останавливаем экземпляры chronos, для того чтобы не было активных соединений до БД:

```
docker service update sed_stack_gossed_chronos --replicas 0
```

Разархивируем архив, из состава поставки migration/gossed_foivog.7z, на sed-swarm-man-01, в удобном месте, после нужно заменить в файле Linux/Tools/app.json параметры:

- **GossedDefaultConnection** на ["Host=17.20.150.43; Database=FOIVOG; User ID=postgres; Password=postgres; Port=5000; Pooling=true; MaxPoolSize=100", "Npgsql"]
- **"https://localhost"** на "https://17.20.150.29/FOIVOG/"

Выдаем права файлу на запуск:

```
chmod +x deploy_db.sh
```

После производим запуск миграции:

```
./deploy_db.sh
```

После окончания миграции снова запускаем chronos командой:

```
docker service update sed_stack_gossed_chronos --replicas 1
```

Проверьте доступность системы через веб-интерфейс по адресу HAProxy:

```
https://17.20.150.29/FOIVOG/web
```

10.2.3. Подключение S3 хранилища

Останавливаем экземпляры chronos, для того чтобы не было активных соединений до БД:

```
docker service update sed_stack_gossed_chronos --replicas 0
```

В файле docker-compose.yaml внести значения в следующие параметры environment:

```
GOSSED__Settings__ObjectStorageS3.BucketName=sed-bucket
GOSSED__Settings__ObjectStorageS3.Endpoint=17.20.150.50:9000
GOSSED__Settings__ObjectStorageS3.AccessKey=3254j26736k3246mk373un3
GOSSED__Settings__ObjectStorageS3.SecretKey=36u35k73kkdrksd6ke56
```

Примечание - Заполнить параметры необходимо в нескольких местах в файле docker-compose.yaml! Данные параметры используются сервисами app_v1, app_v2, chronos_gossed

После замены конфигурации в docker-compose.yml необходимо перезапустить стэк:

```
docker stack deploy -c docker-compose.yml sed_stack
```

Далее выполнить запрос в БД:

```
PGPASSWORD= postgres psql -h 17.20.150.43 -p 5000 -U postgres FOIVOG -c
"UPDATE \"FileSources\" SET \"Path\" = 'sed_bucket' WHERE \"ID\" = '4';"
```

После окончания настройки снова запускаем chronos командой:

```
docker service update sed_stack_gossed_chronos --replicas 1
```

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

Изм.	Номера листов (страниц)				Всего листов (страниц) в докум.	№ документа	Входящий № сопроводи-тельного документа	Подпись	Дата
	измененных	замененных	новых	аннулированных					