ИНСТРУКЦИЯ ПО УСТАНОВКЕ И НАСТРОЙКЕ

ИНФОРМАЦИОННО-УПРАВЛЯЮЩЕЙ СИСТЕМЫ «ПЛАНИРОВАНИЕ РЕЖИМОВ РАБОТЫ ГЭС» (Нептун)

Для системного администратора

Редакция 1.1.

Москва

2024

СОДЕРЖАНИЕ

1. Основные понятия, определения и сокращения

AD	Служба каталогов, являющаяся единым хранилищем данных организации и контролирующая доступ для пользователей на основе политики безопасности каталога.
API	Описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой.
DNS	Компьютерная распределённая система для получения информации о доменах.
IMAPS	Протокол доступа к электронной почте.
HTTP	HyperText Transfer Protocol – протокол прикладного уровня передачи данных.
HTTPS	HyperText Transfer Protocol Secure – расширение протокола HTTP для поддержки шифрования в целях повышения безопасности.
Java	Строго типизированный объектно-ориентированный язык программирования общего назначения.
JavaScript	Прототипно-ориентированный сценарный язык программирования.
JSON	Текстовый формат обмена данными, основанный на JavaScript.
LDAP	Протокол взаимодействия со службой каталогов (AD).
LDAPS	LDAP с поддержкой SSL.
Nexus	Менеджер репозиториев предназначенный для проксирования репозиториев и хранения ПО.
SMTP	Протокол передачи сообщений с компьютера на почтовый сервер для доставки конечному получателю.
REST	Архитектурный стиль взаимодействия компонентов распределённого приложения в сети. REST представляет собой согласованный набор ограничений, учитываемых при проектировании распределённой гипермедиа-системы.
SNMP	Протокол, используемый для управления сетевыми устройствами.
SSL	Криптографический протокол, обеспечивающий защищённую передачу данных между узлами в сети.
SSH	Протокол удаленного управления компьютером с операционной системой Linux.
CPU	Центральный процессор.

RAM	Оперативная память.
HDD	Жесткий диск.
БД	База данных.
ИА	Исполнительный аппарат АО «СО ЕЭС».
ИК	Исходный код.
ИУС	Информационно-управляющая системы.
ИУС СОИ	Информационно-управляющая система «Система обмена информацией».
ПАК	Программно-аппаратный комплекс.
ПАК ЕСМ	ПАК «Единая система мониторинга».
ПАК ИСП	ПАК «Иерархическая система прогнозирования»
ПО	Программное обеспечение.
СУБД	Система управления базами данных.
УЗ	Учётная запись.

2. Назначение руководства

Инструкция описывает действия администратора по установке и настройке ИУС «Нептун (далее по тексту – Система).

Перечисленные в инструкции команды выполняются с использованием SSH-клиента, например – PuTTY.

3. Требования к программным/аппаратным ресурсам

Для установки Системы необходимо подготовить сервера с операционной системой Astra Linux Special Edition в соответствии с данными, указанными в этой главе.

3.1. Требования к аппаратному обеспечению

Рекомендованные характеристики серверов указаны в таблице 1.

Таблица 1 – Рекомендуемая конфигурация серверов Системы

№	Серверы	Кол-во серверов	Рекомендованные характеристики серверов		
			CPU, core	RAM, Gb	HDD, Gb
1	neptune-web	2	2	4	22
2	neptune-backend	2	4	6	24
3	neptune-db	3	4	6	270
	Итого	7	24	38	902

3.2. Требования к программному обеспечению

На серверах **neptune-backend** должно быть установлено следующее ПО:

- Операционная система – Astra Linux Special Edition;
- ПО java liberica jdk версии 17+;
- RabbitMQ версии 3.8.x
- HAProxy версии 2.5+.

На серверах **neptune-web** должно быть установлено следующее ПО:

- Операционная система – Astra Linux Special Edition;
- ПО Nginx версии 1.16.1+;
- ПО Keepalived версии 2.x.x.

На серверах **neptune-db** должно быть установлено следующее ПО:

- Операционная система – Astra Linux Special Edition;
- СУБД – Postgres Pro STD версии 13;

- ПО Patroni 2.1.12+;
- Etcd 3.3.25+.

3.3. Предварительная настройка окружения

Для запуска Системы необходимо:

1. Зарегистрировать DNS имя для frontend сервиса системы (frontend-web).
2. Выпустить SSL сертификаты в PEM[1] формате для сайта Системы.

Если сертификаты предоставлены в формате PFX необходимо произвести конвертацию сертификата в PEM формат. Для конвертации рекомендуется использовать библиотеку *openssl*, документация для ПО доступна по ссылке: https://www.openssl.org/docs/manmaster/man1/openssl.html

Пример конвертации сертификата с именем my.pfx:

```
sudo openssl pkcs12 -in ~/my.pfx -clcerts -nokeys -out /etc/nginx/conf.d/rp-control-web.crt
sudo openssl pkcs12 –in ~/my.pfx/ -nocerts -out ~/my.key
sudo openssl rsa -in ~/my.key -out /etc/nginx/conf.d/rp-control-web.key
```

Таблица 2 содержит список сетевых взаимодействий Системы.

Таблица 2 – Сетевое взаимодействий Системы

Источник	Приёмник	Протокол/Порт
Backend сервер Системы (neptune-backend)		
Компьютер администратора Системы	Сервера приложений (Linux)	TCP-22(SSH) TCP-8080
Сервер ПАК ЕСМ	Сервера приложений (Linux)	TCP-8080 UDP-161
Сервера приложений (Linux)	Сервер ПАК ЕСМ	UDP-162
Сервера приложений (Linux)	Сервер ПАК ИСП	TCP-80 (HTTP) (порт API сервиса может отличаться, необходимо уточнение у администратора)
Сервера приложений (Linux)	Сервера СУБД Системы	TCP-5432, TCP-8008
Сервера приложений (Linux)	Сервер AD (контроллер домена)	TCP-636 (LDAPS)
Сервера приложений (Linux)	Сервер ФПА – хранилище конфигурации (server-git.comm)	TCP-443 (HTTPS)
Сервера приложений (Linux)	Сервер ФПА – хранилище артефактов (server-git.comm)	TCP-443 (HTTPS) TCP-18181

[1] Необходима пара ключей (открытый и закрытый ключ), расширения по умолчанию данной пары -.ctr и .key

Источник	Приёмник	Протокол/Порт
Сервера приложений (Linux)	Почтовый сервер	TCP-993 (IMAPS), TCP-25 (SMTP)
Сервера приложений (Linux)	API ОИК СК-11	TCP-443 (HTTPS) TCP-9443 (HTTPS) (порт API сервиса может отличаться, необходимо уточнение у администратора)
Сервера приложений (Linux)	API ПАК MODES-terminal	TCP-443 (HTTPS) (порт API сервиса может отличаться, необходимо уточнение у администратора)
Сервера приложений (Linux)	Сервер точного времени	UDP-123
Frontend сервера Системы (neptune-web)		
Компьютер администратора Системы	Web-сервера Системы	TCP-22(SSH) TCP-443(HTTPS)
Пользователи Системы	Web-сервера Системы	TCP-443 (HTTPS)
Сервер ПАК ЕСМ	Web-сервера Системы - локальная инсталляция	TCP-443 (HTTPS), UPD-161
Web-сервера Системы	Сервера приложений (Linux) - локальная инсталляция	TCP-8080 UDP-161
Web-сервера Системы	Web-сервера Системы	VRRP
Web-сервера Системы	Сервер ФПА – хранилище артефактов (server-git.comm)	TCP-443 (HTTPS) TCP-18181
Web-сервера Системы	Сервер точного времени	UDP-123
Сервер СУБД Системы (neptune-db)		
Компьютер администратора Системы	Сервера СУБД Системы	TCP-22(SSH) TCP-5432 TCP-8008, TCP-2379, TCP-2380, TCP-7000, TCP-5000,
Сервера приложений (Linux)	Сервера СУБД Системы	TCP-5432, TCP-8008, TCP-2379, TCP-2380, TCP-7000, TCP-5000,

Источник	Приёмник	Протокол/Порт
Сервера СУБД Системы	Сервера СУБД Системы	TCP-5432, TCP-8008, TCP-2379, TCP-2380, TCP-7000, TCP-5000,
Сервера СУБД Системы	Сервер точного времени	UDP-123

4. Установка компонентов системы

4.1. Предварительная настройка серверов Системы

Для интеграции с ПАК ЕСМ необходимо установить пакет snmpd, используя команду:

```
sudo apt update && sudo apt install snmpd
```

4.2. Установка и настройка серверов neptune-backend

4.2.1. Установка Liberica JDK

Для установки Liberica JDK на серверах neptune-backend необходимо подключиться к каждому серверу по SSH и выполнить последовательно следующие команды:

1. Скачиваем deb-пакет bellsoft-jdk17 из реестра проверенного ПО

```
wget -o bellsoft-jdk17.deb -O “https://disk-
cloud.comm/nextcloud/index.php/s/qJQZfmeSFfjBFtW/download?path=%2FLinux%2F
Liberica%20(Java)&files=bellsoft-jdk17.0.5%2B8-linux-amd64.deb”
```

2. Устанавливаем Liberica JDK

```
sudo su
dpkg -i bellsoft-jdk17.deb
```

3. Устанавливаем сертификаты СО в Liberica JDK для работы с СК-11

Для установки сертификата, предполагается наименование «System Operator RSA CP CA.crt», оно может отличаться на момент установки. Для более точного получения сертификата лучше обратиться к СИБ.

После получения и загрузки сертификата, необходимо провести следующую команду проверки:

```
echo ${JAVA_HOME}
```

1. Если ${JAVA_HOME} отображает путь, действуем следующим образом:

```
keytool -importcert -keypass changeit -file "System Operator RSA CP
CA.crt" -keystore ${JAVA_HOME}/lib/security/cacerts -noprompt -storepass
changeit -alias "System Operator RSA CP CA.crt"
```

Выполнение проверки:

```
keytool -list -v -keystore ${JAVA_HOME}/lib/security/cacerts -noprompt -
storepass changeit -alias "System Operator RSA CP CA.crt"
```

2. Если ${JAVA_HOME} не отображает путь, действуем следующим образом в соответствии с тем, что установлен bellsoft-jdk17:

```
export JAVA_HOME=/usr/lib/jvm/bellsoft-java17-amd64
keytool -importcert -keypass changeit -file "System Operator RSA CP
CA.crt" -keystore ${JAVA_HOME}/lib/security/cacerts -noprompt -storepass
changeit -alias "System Operator RSA CP CA.crt"
```

Выполнение проверки:

```
keytool -list -v -keystore /usr/lib/jvm/bellsoft-java17-
amd64/lib/security/cacerts -noprompt -storepass changeit -alias "System
Operator RSA CP CA.crt"
```

4.2.2. Установка RabbitMQ

Для установки RabbitMQ на серверах neptune-backend необходимо подключиться к каждому серверу по SSH и выполнить следующие команды:

```
sudo su
apt-get update
apt-get install rabbitmq-server -y
systemctl enable rabbitmq-server
```

Для подключения RabbitMQ в коастер на серверах neptune-backend необходимо выполнить следующее:

1. Убедиться, что /etc/hosts настроен правильно на всех узлах кластера, чтобы обеспечить корректное разрешение имён. Как пример:

```
127.0.0.1 neptune-backend1
192.168.9.51 neptune-backend2
```

2. Синхронизировать файл .erlang.cookie с узлом, к которому присоединяемся. Этот файл должен быть идентичен на всех узлах кластера и остановить приложение RabbitMQ на присоединяемом узле:

```
scp user@other_node_ip:/var/lib/rabbitmq/.erlang.cookie
/var/lib/rabbitmq/.erlang.cookie

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

service rabbitmq-server restart && rabbitmqctl stop_app
```

3. Сбросить текущее состояние узла, чтобы убрать всю метаинформацию о старом кластере и присоединить узел к кластеру и запустить:

```
rabbitmqctl reset
rabbitmqctl join_cluster rabbit@name_of_the_node_to_join
```

```
rabbitmqctl start_app
```

4. Проверить статус кластера:

```
rabbitmqctl cluster_status
```

4.2.3. Настройка RabbitMQ (НТЦ ЕЭС ИК)

Для настройки RabbitMQ на серверах neptune-backend необходимо подключиться к каждому серверу по SSH и выполнить следующие команды (см. ниже):

Таблица 3 - Параметры для настройки RabbitMQ

Переменные	Пример	Комментарии
$RMQ_USER	Admin_rmq	Учетная запись для сервиса RabbitMQ
$RMQ_PASS	Qwe+1230	Пароль для УЗ $RMQ_USER
$RMQ_DEFAULT_USER	guest	Предустановленные УЗ RabbitMQ

1. Создаем нового пользователя сервиса RabbitMQ

```
sudo su
rabbitmqctl add_user $RMQ_USER
```

Система запросит ввод пароля. Придумываем и вводим его. В случае необходимости - поменять пароль мы можем командой:

```
rabbitmqctl change_password $RMQ_USER
```

2. Даём права созданному пользователю к текущему хосту

```
rabbitmqctl set_permissions -p / $RMQ_USER ".*" ".*" ".*"
```

3. Назначаем созданного пользователя администратором сервиса RabbitMQ

```
rabbitmqctl set_user_tags $RMQ_USER administrator
```

4. В целях безопасности рекомендуется удалить дефолтные учетные записи. Это можно сделать следующей командой:

```
rabbitmqctl delete_user $RMQ_DEFAULT_USER
```

5.Просмотреть список всех пользователей сервиса RabbitMQ можно командой:

```
rabbitmqctl list_users
```

Убеждаемся что созданная УЗ в п.1 является администратором сервиса и единственно настроенная в сервисе RabbitMQ

4.2.4. Установка HAProxy

Для установки haproxy необходимо подключиться к каждой ВМ neptune-backend по SSH и выполнить следующую последовательность действий:

```
sudo apt install haproxy -y
sudo systemctl enable haproxy
```

4.2.5. Настройка HAProxy

1) Настроить конфигурационный файлы HAProxy, через команду:

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

Содержание конфигурационного файла должно быть следующим:

```
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 node1 < IP NODE1>:5432 maxconn 100 check port 6011
    server node2 <IP NODE2>:5432 maxconn 100 check port 6011
    server node3 <IP NODE3>:5432 maxconn 100 check port 6011
```

где

7000 – порт статистики для haproxy;
5000 – порт для подключения к кластеру БД PostgreSQL ИУС Нептун;
<IP NODE{1,2,3}> – ip-адреса всех трех узлов серверовБД;
5432 – порт подключения к PostgreSQL;
6011 – порт restapi patroni.

Пример:

```
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 node1 172.20.11.51:5432 maxconn 100 check port 6011
    server node2 172.20.11.52:5432 maxconn 100 check port 6011
    server node3 172.20.11.53:5432 maxconn 100 check port 6011
```

2) Перезагрузить HAProxy:

```
service haproxy restart
```

3) Проверить корректность работы сервиса HAProxy:

```
service haproxy status
```

Статус сервиса должен соответствовать active (running).

4.3. Установка и настройка web серверов

4.3.1. Установка nginx

1. Для установки nginx необходимо подключиться к каждой ВМ neptune-web по SSH и выполнить следующую последовательность действий:

```
sudo apt install nginx -y
```

2. Добавить сервис nginx в атозапуск и запустить сервис:

```
sudo systemctl start nginx
sudo systemctl enable nginx
```

4.3.2. Настройка nginx

Для настройки nginx необходимо подключиться к каждой ВМ neptune-web по SSH и выполнить следующую последовательность действий:

1. Удалить автоматически созданный файл конфигурации nginx:

```
rm /etc/nginx/sites-available/default
```

2. Очистить директорию www командой

```
rm -r /var/www/*
```

3. Создать директорию веб сайта

```
mkdir /var/www/neptune-front
```

4. Предоставить права УЗ user, в группу которого будут входить все DevOps -инженеры, на директорию с web-приложением neptune-front, используя команду:

```
sudo chown -R user:to-users  /var/www
```

5. Заполнить первичные настройки взаимодействия с сервисами по шаблону ниже, используя команду:

```
sudo nano /etc/nginx/conf.d/upstream.conf
```

Шаблон:

```
upstream neptune-service {
    server neptune-backend ip1:8080;
    server neptune-backend ip2:8080;
}
```

6. Заполнить конфиг-файл веб-сайта по шаблону ниже, используя команду:.

```
sudo nano /etc/nginx/conf.d/upstream.conf
```

Шаблон:

```
server {
    listen 443 ssl;
    ssl_certificate /etc/nginx/conf.d/neptuneserver-name.so.pfx.crt;
    ssl_certificate_key /etc/nginx/conf.d/neptuneserver-name.so.pfx.key;
    server_name neptune-server-name;
    root /var/www/neptune-front;
    gzip on;
```

```
    gzip_types text/css application/javascript application/json
image/svg+xml;
    gzip_comp_level 9;
    etag on;

    location /api {
        proxy_pass http://neptune-service;

        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "Upgrade";
        proxy_set_header Origin http://$host;

        proxy_http_version 1.1;

        proxy_buffering off;

        proxy_connect_timeout 7d;
        proxy_send_timeout 7d;
        proxy_read_timeout 7d;

        proxy_socket_keepalive on;
    }

    location ~ ^/(int-api|ext-api|swagger-ui|v3/api-docs) {
        proxy_pass http://neptune-service;

        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "Upgrade";

        proxy_http_version 1.1;

        proxy_buffering off;

        proxy_connect_timeout 7d;
        proxy_send_timeout 7d;
        proxy_read_timeout 7d;

        proxy_socket_keepalive on;
    }

    client_max_body_size 200m;
```

```
    proxy_connect_timeout 60;
    proxy_send_timeout 60;
    proxy_read_timeout 2400;
    send_timeout 2400;

    location / {
        try_files $uri $uri/ /index.html =404;
    }

    location /index.html {
        add_header Cache-Control 'no-store, no-cache, must-revalidate,
proxy-revalidate, max-age=0';
        if_modified_since off;
        expires off;
        etag off;
    }
}
```

7. Убедимся, что конфигурация nginx настроена правильно командой:

```
nginx -T
```

8. Перезапустим сервис nginx:

```
systemctl restart nginx
```

9. Установка и настройка web серверов закончена. Для проверки работоспособности Nginx необходимо выполнить команду:

```
systemctl status nginx | grep active
```

Ожидаемый ответ:

```
Active: active (running)
```

4.3.3. Установка keepalived для кластера балансировки нагрузки.

ПО keepalived необходимо для организации отказоустойчивого кластера. Для установки keepalivedнеобходимо подключиться к каждой ВМ neptune-web по SSH и выполнить следующую последовательность действий:

```
sudo su
apt-get update
apt-get install keepalived -y
echo net.ipv4.ip_forward=1 >> /etc/sysctl.conf
sysctl -p
touch /etc/keepalived/keepalived.conf
```

Так же необходимо добавить сервис в автозагрузку командой:

```
systemctl enable keepalived
```

4.3.4. Настройка keepalived

Для завершения конфигурации keepalived необходимо отредактировать конфигурационный файл командой sudo nano /etc/keepalived/keepalived.conf, добавив в него нижеприведенную конфигурацию и изменить значение priority в зависимости от роли сервера (основной/резервный).

Переменную <IP> необходимо заменить на ip адрес, выделенный для работы frontend сервиса.

```
lobal_defs {
script_user root
enable_script_security
}
vrrp_script chk_nginx {
    script "ps -C nginx"
    interval 2
}
vrrp_instance NEPTUNE_WEB {
    state MASTER #BACKUP
    interface eth0 #Указываем интерфейс, к которому будет привязан VRRP
instance
    virtual_router_id 200 #Уникальное значение кластера
    #Должен быть одинаков на всех хостах в instance
    #допустимые значения от 1 до 255.
    priority 110 #Для основного узла указываем 110 для резервного 100.
    advert_int 4
    #Настройка аутентификации по паролю
    authentication {
        auth_type PASS
        auth_pass 12345
    }
    #Настройка виртуального сетевого интерфейса
    virtual_ipaddress {
            <IP> dev eth0 label eth0:vip
    }
    track_script {
            chk_nginx
    }
}
```

После чего необходимо перезапустить сервис командой:

```
systemctl restart keepalived
```

Установка и настройка keepalived закончена для проверки установки необходимо выполнить команду:

```
systemctl status keepalived
```

Статус сервиса должен соответствовать active (running).

Для основного сервера в выводе должно содержаться сообщение:

```
VRRP_Instance(NEPTUNE_WEB) Entering MASTER STATE
```

Для резервного сервера в выводе должно содержаться сообщение:

```
VRRP_Instance(NEPTUNE_WEB) Entering BACKUP STATE
```

4.4. Установка и настройка СУБД

4.4.1. Установка сервиса etcd

Для установки etcd необходимо подключиться по ssh на каждый узел кластера БД, выделенный для установки СУБД и выполнить следующие команды:

```
sudo apt-get install etcd
```

4.4.2. Настройка Etcd

Для настройки etcd необходимо подключиться по ssh на каждый узел кластера БД, выделенный для установки СУБД и выполнить следующие команды:

1) Настроить конфигурационный файл согласно шаблону (см. ниже), через команду:

```
nano /etc/default/etcd
```

Переменная	Пример	Комментарий
ETCD_NAME	sqlnode1	hostname текущей машины
ETCD_LISTEN_PEER_URLS	http://127.0.0.0:2380	адрес текущей машины
ETCD_LISTEN_CLIENT_URLS	http://127.0.0.0:2379	адрес текущей машины
ETCD_INITIAL_ADVERTISE_PEER_URLS	http://127.0.0.0:2380	адрес текущей машины
ETCD_INITIAL_CLUSTER	sqlnode1=http://127.0.0.0:2380,sqlnode2=http://127.0.1.0:2380	адреса всех машин в кластере etcd
ETCD_INITIAL_CLUSTER_STATE	new	статус текущего кластера
ETCD_INITIAL_CLUSTER_TOKEN	etcd-cluster	токен кластера

ETCD_ADVERTISE_CLIENT_URLS	http://127.0.0.0:2379	адрес текущей машины

Пример:

```
[member]
ETCD_NAME=sqlnode1ETCD_LISTEN_PEER_URLS="http://192.168.0.143:2380"
ETCD_LISTEN_CLIENT_URLS="http://192.168.0.143:2379"
[cluster]
ETCD_INITIAL_ADVERTISE_PEER_URLS="http://192.168.0.143:2380"
ETCD_INITIAL_CLUSTER="=sqlnode1=http://192.168.0.143:2380,sqlnode2=http://
192.168.0.144:2380,sqlnode13=http://192.168.0.145:2380
ETCD_INITIAL_CLUSTER_STATE="new"
ETCD_INITIAL_CLUSTER_TOKEN="etcd-cluster"
ETCD_ADVERTISE_CLIENT_URLS="http://192.168.0.143:2379"
```

4.4.3. Установка СУБД

Для установки PostgreSQL необходимо подключиться по ssh на каждый узел кластера БД, выделенный для установки СУБД и выполнить следующие команды:

1. **Добавляем репозиторий Postgres PRO Enterprise**

```
sudo su
echo "deb [trusted=yes arch=amd64] http://rep-poz.comm /postgrespro13 main
contrib non-free">> /etc/apt/sources.list.d/pgpro.list
```

2. **Добавляем исключение для прокси**

```
echo 'Acquire::http::PROXY::rep-poz.comm "DIRECT";' >>
/etc/apt/apt.conf.d/noproxy
```

3. **Обновить список пакетов с репозитория**

```
apt-get update
```

4. **Установить пакет Postgres и rsync**

```
apt install -y postgrespro-std-13
```

5. Разрешить **подключение** к PostgreSQL с внешних узлов:

```
echo "host all all 0.0.0.0/0 md5" >> /var/lib/pgpro/std-
13/data/pg_hba.conf
echo "listen_addresses = '*'" >> /var/lib/pgpro/std-
13/data/postgresql.conf
```

3. Запустить СУБД PostgreSQL:

```
sudo systemctl enable postgrespro-std-13
sudo systemctl restart postgrespro-std-13
systemctl status postgrespro-std-13
```

В строке, которая начинается с «Active:» должен быть указан статус «active (running)»

4. Присвоить УЗ postgres пароль командой:

```
sudo passwd postgres
```

На запрос системы необходимо дважды ввести пароль.

4.4.4. Настройка СУБД

Для настройки основного сервера СУБД необходимо создать учетные записи и базы данных для сервисов Системы. Для этого необходимо:

1. Выполнить команды в соответствии с шаблоном (см. ниже).

Таблица 5 содержит описание параметров, указанных в шаблоне.

Таблица 4 – Параметры конфигурации БД

Переменные	Пример	Комментарии
$PG_PSWD	PassWord	Пароль привилегированной учетной записи PostgreSQL
$DB_NAME	neptune-db	Имя БД
$DB_USER	neptune-user	УЗ для доступа к БД
$DB_PASS	Qwe+1230	Пароль для УЗ $DB_USER
$MAIN_DB	10.0.0.100	IP адрес основного сервера СУБД

Шаблон:

```
#Переключиться в консоль привилегированного пользователя СУБД
su postgres
#Войти в консоль СУБД
psql
#Изменить пароль входа в СУБД для пользователя postgres
ALTER USER postgres WITH PASSWORD '$PG_PSWD';
#Создать УЗ для БД
CREATE USER "$DB_USER" WITH PASSWORD '$DB_PASS' LOGIN;
#Создать основную БД
CREATE DATABASE "$DB_NAME";
#Предоставить права к БД для УЗ
GRANT ALL ON DATABASE "$DB_NAME" TO "$DB_USER" WITH GRANT OPTION;
#Выйти из консоли СУБД
\q
#Выйти из консоли пользователя postgres
exit
```

Пример:

```
su postgres
psql
ALTER USER postgres WITH PASSWORD '*****';
CREATE USER "neptune-user" WITH PASSWORD '*****' LOGIN;
CREATE DATABASE "neptune-db";
GRANT ALL ON DATABASE "neptune-user" TO "neptune-db" WITH GRANT OPTION;
\q
exit
```

4.4.5. Установка Patroni

1. Необходимо подключиться по ssh на каждый узел кластера БД, выделенный для установки СУБД и остановить сервис и отключить postgres на всех узлах кластера баз данных и приложений:

```
sudo systemctl stop postgrespro-std-13
sudo systemctl disable postgrespro-std-13
```

2. Установить patroni на каждом из узлов кластера баз данных и приложений с помощью следующих команд:

#Устанавливаем Python

```
sudo apt-get update
sudo apt-get install python3-pip python3-dev python3-requests postgrespro-std-13-dev postgrespro-std-13-libs -y
```

#Создаем файл конфигурации для python

```
sudo cat <<EOF > /etc/pip.conf
[global]
index = https://ups_worker:Vn7Z3g5mQEvHtybm@server-git.comm/repository/pypi-group/pypi
index-url = https://ups_worker:Vn7Z3g5mQEvHtybm@server-git.comm/repository/pypi-group/simple
trusted-host = server-git.comm

EOF
```

Устанавливаем пакеты patroni

```
pip3 install --upgrade pip
export PATH="/opt/pgpro/ent-13/bin/:$PATH"
pip3 install psycopg2
pip3 install patroni[etcd]==3.0.2
pip3 install psycopg2-binary
```

Удаляем оригинальный инстанс СУБД

```
sudo rm -fr /var/lib/pgpro/std-13/data/*
```

Добавляем английскую локаль

```
sed -i "s/# en_US.UTF-8/en_US.UTF-8/" /etc/locale.gen
locale-gen en_US.UTF-8
```

3. Создаем каталоги для хранения БД:

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

4. Создаем юнит файл сервиса patroni по шаблону (см. ниже)

```
sudo cat << EOF > /etc/systemd/system/patroni.service
```

Шаблон:

```
[Unit]
Description=Runners to orchestrate a high-availability PostgreSQL
After=syslog.target network.target

[Service]
Type=simple
 User=postgres
Group=postgres
 ExecStart=/usr/local/bin/patroni /etc/patroni.yaml
 KillMode=process
 TimeoutSec=30
 Restart=no

[Install]
WantedBy=multi-user.target\

EOF
```

4.4.6. Настройка Patroni

Для настройки Patroni необходимо подключиться по ssh на каждый узел кластера БД, выделенный для установки СУБД и выполнить следующие команды:

1. Создаем настроечный файл сервиса patroni после чего корректируем переменные согласно комментариям

Команда редактирования:

```
sudo touch /etc/patroni.yaml
sudo nano /etc/patroni.yaml
```

Содержимое:

```
sudo cat << EOF > /etc/patroni.yaml
scope: pgsql_sepg # должно быть одинаковым на всех нодах
namespace: /cluster_srdk/ # должно быть одинаковым на всех нодах
name: postgres3 # должно быть разным на всех нодах

restapi:
    listen:  sqlnode1_ip:8008 # адрес той ноды, в которой находится этот
файл
    connect_address:  sqlnode1_ip:8008 # адрес той ноды, в которой
находится этот файл

etcd:
```

```
    hosts:  sqlnode1_ip:2379,sqlnode2_ip:2379,sqlnode3_ip:2379 #
перечислите здесь все ваши ноды, в случае если вы устанавливаете etcd на
них же
      username: patroni
    password: пароль_пользователя{}patroni{}созданный при настройке etcd


# this section (bootstrap) will be written into
Etcd:/<namespace>/<scope>/config after initializing new cluster
# and all other cluster members will use it as a `global configuration`
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: 512
                    max_wal_senders: 5
                    max_replication_slots: 5
                    checkpoint_timeout: 30

    initdb:
    - encoding: UTF8
    - data-checksums
    - locale: en_US.UTF8
    # init pg_hba.conf должен содержать адреса ВСЕХ машин, используемых в
кластере
    pg_hba:
    - host replication postgres ::1/128 md5
    - host replication postgres 127.0.0.1/8 md5
    - host replication postgres sqlnode1_ip/24 md5
    - host replication postgres sqlnode2_ip/24 md5
    - host replication postgres sqlnode3_ip/24 md5
    - host all all 0.0.0.0/0 md5

    users:
        admin:
            password: *** #придумать пароль
            options:
                - createrole
                - createdb

postgresql:
```

```
    listen:  sqlnode1_ip:5432 # адрес той ноды, в которой находится этот
файл
    connect_address:  sqlnode1_ip:5432 # адрес той ноды, в которой
находится этот файл
    data_dir: /data/patroni # эту директорию создаст скрипт, описанный
выше и установит нужные права
    bin_dir:  /opt/pgpro/ent-13/bin # укажите путь до вашей директории с
postgresql
    pgpass: /tmp/pgpass
    authentication:
        replication:
            username: postgres
            password: *** #придумать пароль
        superuser:
            username: postgres
            password: *** #придумать пароль
    create_replica_methods:
        basebackup:
            checkpoint: "fast"
    parameters:
        unix_socket_directories: "."

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

2. Используя следующую команду, редактирум файл конфигурации согласно комментариям.

```
nano  /etc/patroni.yaml
```

3. Запускаем сервис Patroni командой:

```
systemctl start patroni
```

4. Проверяем работу сервиса используя команду:

```
patronictl -c /etc/patroni.yaml list
```

5. Ожидаемый результат после запуска сервиса на всех узлах кластера:

```
+ Cluster: pgsql_sepg (7099461315590300498) --+----+-----------+
| Member    | Host        | Role    | State   | TL | Lag in MB |
```

```
+-----------+-------------+---------+---------+----+-----------+
| postgres2 | 10.15.1.85  | Replica | running | 13 |         0 |
| postgres3 | 10.15.1.86  | Replica | running | 13 |         0 |
| postgres4 | 10.15.1.82  | Leader  | running | 13 |           |
+-----------+-------------+---------+---------+----+-----------+
```

4.4.7. Настройка резервного копирования СУБД

Для создания резервных копий баз необходимо настроить сохранения резервных копий и логов транзакций в сетевой каталог.

Хранение резервных копий рекомендуется на сетевом каталоге. Для облегченного доступа к резервным копиям рекомендуется создать сетевую папку на сервере под управлением любой версии Windows, а так же создать учетную запись и предоставить ей права на запись как в файловой системе, так и на уровне сетевого доступа.

Для настройки резервного копирования кластера СУБД Postgres на сетевой диск доступный по протоколу SMB необходимо подключиться к консоли узла через ssh и выполнить следующие действия:

1. Произвести установку cifs-utils;

```
sudo apt update
sudo apt install -y cifs-utils
```

2. Создать файл /root/.smbclient с параметрами доступа к сетевому каталогу Windows:

```
sudo nano /root/.smbclient
```

Заполнить файл, указав логин, пароль, домен:

```
username=<логин>
password=<пароль>
domain=<домен: например, comm>
```

3. Создать каталог на сервере Linux, в который будет монтироваться сетевой каталог Windows:

```
sudo mkdir /srv/backup
```

4. Настроить автоматическое монтирование сетевого диска . Для этого необходимо отредактировать файл /etc/fstab, командой `sudo nano /etc/fstab`, и добавить в данный файл строку:

```
//winserver/Share/ /srv/backup cifs
uid=postgres,gid=postgres,rw,credentials=/root/.smbclient,file_mode=0600,d
ir_mode=0777 0 0
```

где:

- //winserver/Share/ – путь к сетевому каталогу Windows, заменить на нужный путь, при этом меняем «\» на «/»);

- /srv/backup – точка (каталог) монтирования на сервере Linux, созданный на шаге 3 текущего раздела;
- /root/.smbclient – полный путь файла с параметрами доступа к сетевому каталогу Windows, созданному на шаге 2 текущего раздела.

Внимание! Если в пути каталога встречается «пробел» необходимо указывать его через запись «\040».

5. Запустить процесс монтирования каталогов в соответствии с настройками, указанными в файле /etc/fstab:

```
sudo mount -a
```

6. Создать директории для хранения резервных копий СУБД.

```
sudo mkdir /srv/backup/postgres
```

7. Настроить ежедневное создание полной копии СУБД. Для этого на сервере СУБД, используя команду `sudo -u postgres crontab -e` добавляем в cron строку:

```
00 22 * * * PGPASSWORD="$REPLICA_PSWD" pg_basebackup -h MYIP -U
replication -F t -D /srv/backup/postgres/$(date +\%Y\%m\%d) -X stream -z -
p 543
```

MYIP заменить на IP сервера.

$REPLICA_PSWD - пароль пользователя от которого будет производиться бэкап(replication)

В результате каждый день в 22-00 будет создаваться, сжатая архиватором gzip, полная архивная копия СУБД.

8. Настраиваем очистку каталога с резервными копиями СУБД, для этого на сервере СУБД, используя команду `sudo -u postgres crontab -e` добавляем в cron строку:

```
40 23 * * * /usr/bin/find /srv/backup/postgres/ -maxdepth 1 -type d -
mtime +5 -exec rm -rf {} \;
```

В результате ежедневно будет производится очистка резервных копий СУБД, будут удалены архивы старше 5 дней.

9. В случае если `sudo -u postgres crontab -e` завершается с ошибкой, для выполнения пунктов 7 и 8 необходимо добавить в файл /etc/crontab строки

```
00 22 * * * postgres PGPASSWORD="$REPLICA_PSWD" pg_basebackup -h MYIP -U
replication -F t -D /srv/backup/postgres/$(date +\%Y\%m\%d) -X stream -z -
p 5432
40 23 * * * postgres /usr/bin/find /srv/backup/postgres/ -maxdepth 1 -
type d -mtime +5 -exec rm -rf {} \;
```

С аналогичной заменой MYIP на адрес сервера.

10. Настраиваем очистку каталога с журналами СУБД, для этого на каждом сервере СУБД, используя команду sudo -u postgres crontab –e добавляем в cron строку:

```
40 23   * * * find /data/patroni/pg_wal/ -maxdepth 1  -type f -mtime +2 -
delete
```

В результате ежедневно будет производится очистка журналов СУБД, будут удалены архивы старше 2 дней.

5. Передача данных группе КТО

После выполнения установки группе КТО необходимо передать:

1. IP адреса и имена ВМ Системы;
2. Пароли и УЗ для подключения к БД.

6. Лист регистрации изменений

№ п/п	Автор	Редакция	Дата	Описание изменения
1	АО «НТЦ ЕЭС Информационные комплексы»	1.0	30.09.2024	Первая версия инструкции по установке и настройке
2	АО «НТЦ ЕЭС Информационные комплексы»	1.1	08.10.2024	Поправки в инструкции по установке и настройке