Программное обеспечение «Базис.Virtual Security». Руководство по установке. Часть 1. Программный модуль «Базис.Virtual Security»

RU.НРФЛ.00002-02 93 01 Ч1

Москва

09/11/2023

Содержание

Общая информация 4

Аннотация 4

Термины и определения 4

Идентификационные данные документа 4

Требования Базис.Virtual Security 5

Требования к оборудованию программного модуля «Базис.Virtual Security» 5

Минимальные требования к аппаратному обеспечению для функционирования программного модуля «Базис.Virtual Security» 5

Установка программного модуля Базис.Virtual Security 8

Простая инсталляция программного модуля Базис.Virtual Security 8

Установка на ALT Linux Базис.Virtual Security 8

Установка на AstraLinux Базис.Virtual Security 13

- Инсталляция с поддержкой высокой доступности программного модуля Базис.Virtual Security 19

Инсталляция с поддержкой высокой доступности AltLinux 19

Инсталляция с поддержкой высокой доступности AstraLinux 31

Обновление программного модуля Базис.Virtual Security 46

Предварительные действия перед обновлением 46

Управляющая часть 46

Обновление пакетов 46

Обновление конфигурации сервисов 46

Запуск сервисов 47

Вычислительная часть 47

Обновление конфигурации сервисов 47

Запуск сервисов 47

Приложения к руководству по установке программного модуля Базис.Virtual Security 48

Приложение 1. Настройка модуля PostgreSQL PAM для программного модуля Базис.Virtual Security 48

Введение 48

Установка модуля 48

Настройка модуля 49

Настройка модуля для иных OpenID-провайдеров 50

Проверка установки и настройки 52

Приложение 2. Интеграция программного модуля Базис.Virtual Security с Базис.vCore 52

Введение 52

Управляющая часть 52

Вычислительная часть 53

Приложение 3. Интеграция программного модуля Базис.Virtual Security с кластерной инсталляцией программного модуля Базис.vCore 54

Подготовка кластера vCore 54

Настройка Pacemaker 55
Настройка портов 56
Установка Virtual Security 57
Настройка кластера PostgreSQL 60
Настройка кластера Pacemaker для PostgreSQL 63
Приложение 4. Список портов сервисов программного модуля Базис.Virtual Security 65

Общая информация

Аннотация

Настоящее руководство предназначено для технического администратора ПО и содержит подробное описание работ, необходимых для установки ПО.

Термины и определения

Термин, сокращение	Определение
Apache	Apache является кроссплатформенным ПО, поддерживает операционные системы Linux, BSD, macOS, Microsoft Windows, Novell NetWare, BeOS. Основными достоинствами Apache считаются надёжность и гибкость конфигурации
DNS	(англ. Domain Name System «система доменных имён») — компьютерная распределённая система для получения информации о доменах. Чаще всего используется для получения IP-адреса по имени хоста (компьютера или устройства), получения информации о маршрутизации почты и/или обслуживающих узлах для протоколов в домене
NgInx	Веб-сервер и почтовый прокси-сервер, работающий на Unix-подобных операционных системах
PostgreSQL	Свободная объектно-реляционная система управления базами данных (СУБД). Существует в реализациях для множества UNIX-подобных платформ, включая AIX, различные BSD-системы, HP-UX, IRIX, Linux, macOS и т.д.
RAM	Форма компьютерной памяти, которая может быть прочитана и изменена в любом порядке, обычно используемая для хранения рабочих данных и машинного кода
БД	База данных
ОС	Операционная система

Идентификационные данные документа

Идентификационные данные ПО	Программное обеспечение «Базис.Virtual Security»
Название документа	Программное обеспечение «Базис.Virtual Security». Руководство по установке. Часть 1. Программный модуль «Базис.Virtual Security»
Версия документа	2.0
Обозначение документа	RU.НРФЛ.00002-02 93 01 Ч1
Автор документа	ООО «БАЗИС»

Требования Базис.Virtual Security

Требования к оборудованию программного модуля «Базис.Virtual Security»

Компоненты «Базис.Virtual Security»:

- сервис-агент;
- сервис аутентификации и авторизации;
- контроллер управления;
- модуль аутентификации в PostgreSQL;
- модуль управления безопасностью API.

Минимальные требования к аппаратному обеспечению для функционирования программного модуля «Базис.Virtual Security»

К аппаратному и программному обеспечению, которые используются для функционирования программного модуля «Базис.Virtual Security», предъявляются требования, изложенные в таблице:

Элемент	**Параметр**
Сервис аутентификации и авторизации, контроллер управления	
Сервис-агент в среде виртуализации KVM	
Операционная система	Astra Linux Special Edition сертификат № 2557 (выдан ФСТЭК России 27.01.2012, действителен до 27.01.2026); Альт 8 СП СЕРВЕР сертификат № 3866 (выдан ФСТЭК России 10.08.2018, действителен до 10.08.2028)
Библиотеки	Libvirt из состава ОС; qemu из состава ОС
Процессор	В соответствии с требованиями ОС, установленной в среде виртуализации.
Оперативная память	В соответствии с требованиями ОС, установленной в среде Виртуализации и 512 МБ дополнительно.
Жесткий диск	1 ГБ
Сетевой адаптер	Ethernet 1 Гбит/сек
Операционная система	Astra Linux Special Edition сертификат № 2557 (выдан ФСТЭК России 27.01.2012, действителен до 27.01.2026); Альт 8 СП СЕРВЕР сертификат № 3866 (выдан ФСТЭК России 10.08.2018, действителен до 10.08.2028)
СУБД	Система управления базами данных «Postgres Pro» сертификат № 3637 (выдан ФСТЭК России 05.10.2016, действителен до 05.10.2024); Система управления базами данных Postgres Pro Enterprise сертификат № 4063 (выдан ФСТЭК России 16.01.2019, действителен до 16.01.2029);

Элемент	Параметр
Сервис аутентификации и авторизации, контроллер управления	
Процессор	Процессор 4 ядра серверного класса с поддержкой виртуализации и тактовой частотой от 2.0 ГГц.
Оперативная память	16 ГБ
Жесткий диск	10 ГБ
Сетевой адаптер	Ethernet 1 Гбит/сек
Модуль аутентификации в PostgreSQL	
СУБД	Система управления базами данных «Postgres Pro» сертификат № 3637 (выдан ФСТЭК России 05.10.2016, действителен до 05.10.2024); Система управления базами данных Postgres Pro Enterprise сертификат № 4063 (выдан ФСТЭК России 16.01.2019, действителен до 16.01.2029);
Операционная система	Astra Linux Special Edition сертификат № 2557 (выдан ФСТЭК России 27.01.2012, действителен до 27.01.2026); Альт 8 СП СЕРВЕР сертификат № 3866 (выдан ФСТЭК России 10.08.2018, действителен до 10.08.2028);
Процессор	x86_64
Оперативная память	4 ГБ
Жесткий диск	1 ГБ
Клиент	
Браузер	Из состава ОС
Сетевой адаптер	Ethernet 100 Мбит/сек
Монитор	Диагональ от 17"; разрешение от 1280x1024 (4:3), от 1440x900 (16:9).
Периферийное оборудование	Клавиатура, манипулятор типа мышь.
Модуль управления безопасностью API	
Операционная система	Альт 8 СП СЕРВЕР сертификат № 3866 (выдан ФСТЭК России 10.08.2018, действителен до 10.08.2028)
Процессор	Процессор 4 ядра серверного класса с поддержкой виртуализации и тактовой частотой от 2.0 ГГц.
Оперативная память	8 ГБ

Элемент	Параметр
Сервис аутентификации и авторизации, контроллер управления	
Жесткий диск	10 ГБ
Сетевой адаптер	Ethernet 1 Гбит/сек

Установка программного модуля Базис.Virtual Security

Простая инсталляция программного модуля Базис.Virtual Security

Установка на ALT Linux Базис.Virtual Security

- Введение (см. стр. 8)
- Установка управляющих компонентов (см. стр. 8)
  - Предварительная настройка (см. стр. 8)
    - Настройка клиента точного времени (см. стр. 8)
    - Добавление репозиториев VS (см. стр. 9)
- Настройка управляющей части (см. стр. 9)
  - Установка PostgreSQL (см. стр. 9)
  - Установка пакетов Virtual Security (см. стр. 10)
    - Auth (см. стр. 10)
    - Admin (см. стр. 10)
    - Web (см. стр. 11)
    - Broker (см. стр. 12)
    - vbalancer (см. стр. 12)
- Настройка агента вычислительного узла (см. стр. 12)
  - Установка пакетов и настройка сервиса (см. стр. 13)

Важно

Для корректной работы Базис.Virtual Security рекомендуется использовать доменную адресацию на узлах.

Введение

Данная статья описывает простую установку Virtual Security (VS) на Alt Linux, при котором компоненты управления будут запущены на единственном узле. Документация покрывает установку на следующие редакции Alt Linux:

- СП 8, релиз 9 (ветка c9f2);
- Десятая редакция (ветка p10).

Установка поделена на две основные части:

- Установка управляющей части VS;
- Установка агентов VS на вычислительных узлах.

Установка управляющих компонентов

Предварительная настройка

Для корректной работы основных компонентов VS необходимо настроить отдельные части операционной системы.

Настройка клиента точного времени

Для нормального взаимодействия между частями платформы важно, чтобы часы во всех узлах были засинхронизированы через единый NTP-сервер. Рекомендуется использование локального NTP-сервера, однако допускается использование публичных NTP-пулов. В качестве клиента NTP рекомендуется использовать chrony[1].

Установите клиент chrony:

```
apt-get install chrony
```

apt-get может предложить удалить другой сервис точного времени, соглашайтесь на удаление.

1 https://chrony.tuxfamily.org/

Добавление репозиториев VS

Для получения пакетов VS нужно добавить официальный репозиторий для всех узлов VS (управляющих и вычислительных). Для этого в файле /etc/apt/sources.list.d/bvs.list укажите адреса репозиториев (имя репозитория p10 формальное, пакеты совместимы со всеми релизами ALT):

```
rpm https://bvs-public:bvs-public@pkgs.basistech.ru/artifactory/bvs-alt-rpm/p10/x86_64
classic
rpm https://bvs-public:bvs-public@pkgs.basistech.ru/artifactory/bvs-alt-rpm/p10/noarch
classic
```

Обновите данные репозиториев:

```
apt-get update
```

Настройка управляющей части

> Для корректной работы сервисов необходимо наличие двух сетевых интерфейсов. В данной инструкции используются: bvs-test и bvs-stand.

Установка PostgreSQL

В качестве СУБД для хранения данных VS использует СУБД PostgreSQL.

Установите пакет `postgresql` версии 15:

```
apt-get install postgresql15-server
```

Запустите инициализацию СУБД:

```
/etc/init.d/postgresql initdb
```

Откройте файл `/var/lib/pgsql/data/postgresql.conf` и измените следующую строку:

```
listen_addresses = 'bvs-test'
```

Откройте файл `/var/lib/pgsql/data/pg_hba.conf` и в конец добавьте следующее:

```
host   vs   vs   bvs-test      md5
```

Запустите СУБД:

```
systemctl start postgresql
```

Добавьте СУБД в автозагрузку:

```
systemctl enable postgresql
```

Добавьте, по необходимости рабочий shell учетной записи postgres:

```
usermod postgres -s /bin/bash
```

Зайдите в окружение postgres:

```
su - postgres
```

Запустите терминальный клиент для работы с СУБД:

```
psql
```

Создайте все объекты внутри СУБД:

```
create user vs with password 'vs';
create database vs owner vs;
\c vs;
create schema vsm authorization vs;
create schema idp authorization vs;
exit;
```

Выйдите из окружения postgres:

```
exit
```

Проверьте вход в базу vs пользователем vs через указанный адрес (пароль будет запрошен интерактивно):

```
psql -h bvs-test -U vs
```

Если вы смогли войти в окружение пользователя vs внутри psql, значит, аутентификация была настроена верно.

Установка пакетов Virtual Security

Auth

Установите сервис:

```
apt-get -y install virtual-security-auth
```

Настройте конфигурационный файл - `/opt/bvs/auth/lib/config/idp.conf`:

- Укажите базовый URL для сервиса:

  ```
  hostname-url=http://bvs-stand:8080
  hostname-admin-url=http://bvs-stand:8080
  ```

- Укажите адрес т порт прослушивания:

  ```
  http-host=bvs-test
  http-port=8082
  ```

- Укажите адрес подключения к postgresql:

  ```
  db-url-host=bvs-test
  ```

Настройте конфигурационный файл - `/opt/bvs/auth/lib/config/application.properties`:

```
quarkus.mod-cluster-client.enabled=true
```

Запустите сервис:

```
systemctl start virtual-security-auth
```

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-auth
```

Admin

Установите сервис:

```
apt-get -y install virtual-security-admin
```

Настройте конфигурационный файл - `/opt/bvs/admin/config/application.yaml`:

```
client:
```

```
    base-uri: http://bvs-stand:8080
    realm: master
    client-id: security-admin-console
  security-proxy:
    base-uri: http://bvs-stand:8080
  provider:
    messaging:
      count: 2
      broker-uri: tcp://bvs-test:61616?
clientFailureCheckPeriod=5000&retryInterval=1000&reconnectAttempts=5&callTimeout=3000

...

http:
    host: bvs-test
    port: 8081
    root-path: /api
    cors:
      ~: true
      origins: http://bvs-stand:8080

...

datasource:
    db-kind: postgresql
    username: vs
    password: vs
    jdbc:
      url: jdbc:postgresql://bvs-test:5432/vs
      max-size: 10

...

 mod-cluster-client:
    enabled: true
```

Запустите сервис:

```
systemctl start virtual-security-admin
```

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-admin
```

Web

Установите сервис:

```
apt-get -y install virtual-security-web
```

Настройте конфигурационный файл - `/opt/bvs/web/config/application.properties`:

```
quarkus.http.host=bvs-test
...
quarkus.oidc.auth-server-url=http://bvs-stand:8082/realms/master
...
quarkus.mod-cluster-client.enabled=true
...
quarkus.mod-cluster-client.services.api.host=bvs-test
quarkus.mod-cluster-client.services.api.port=8080
quarkus.mod-cluster-client.services.api.contexts=/webadmin
```

Запустите сервис:

```
systemctl start virtual-security-web
```

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-web
```

Broker

Установите сервис:

```
apt-get -y install vcore-broker
```

Настройте конфигурационный файл - `/opt/vcore/broker/config/application.properties`:

```
quarkus.http.host=bvs-test
quarkus.http.port=8071
...
quarkus.mod-cluster-client.enabled=true
quarkus.mod-cluster-client.services.api.host=bvs-test
...
provider.transport.use-multicast-discovery=true
...
provider.transport.multicast-interface-address=bvs-test
provider.transport.multicast-group=231.7.7.7
...
provider.transport.bind-address=bvs-test
```

Запустите сервис:

```
systemctl start vcore-broker
```

Добавьте сервис в автозагрузку:

```
systemctl enable vcore-broker
```

vbalancer

Установите сервис:

```
apt-get -y install basis-vbalancer
```

Настройте конфигурационный файл - /etc/sysconfig/basis-vbalancer:

```
BALANCER_BIND_ADDR="bvs-stand"
BALANCER_PRIVATE_BIND_ADDR="bvs-test
```

Запустите сервис:

```
systemctl start basis-vbalancer
```

Добавьте сервис в автозагрузку:

```
systemctl enable basis-vbalancer
```

Настройка агента вычислительного узла

В каждом вычислительном узле должен быть агент Virtual Security, который контролирует выполнение всех операций безопасности.

> **Примечание**
>
> В окружениях Alt Linux агент VS совместим только с "чистой" установкой libvirt и гипервизора. Не используйте агент VS с другими агентами управления виртуализацией. vCore это не касается, так как он использует собственную ОС.

Установка пакетов и настройка сервиса

Установите следующие пакеты:

```
apt-get -y install basis-openjdk virtual-security-agent
```

Настройте конфигурационный файл - `/opt/bvs/agent/config/application.properties`:

```
quarkus.http.host=bvs-stand
...
agent.provider.jms.broker-uris=tcp://bvs-stand:61616?
clientFailureCheckPeriod=5000&retryInterval=1000&reconnectAttempts=5&callTimeout=3000
```

Запустите сервис:

```
systemctl start virtual-security-agent
```

После этого агент Virtual Security будет добавлен в управляющую часть.

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-agent
```

Установка на AstraLinux Базис.Virtual Security

- Введение (см. стр. 13)
- Установка управляющих компонентов (см. стр. 13)
  - Предварительная настройка (см. стр. 13)
    - Добавление репозиториев VS (см. стр. 14)
    - Настройка клиента точного времени (см. стр. 14)
- Настройка управляющей части (см. стр. 14)
  - Установка PostgreSQL (см. стр. 14)
  - Установка Network Manager (см. стр. 15)
  - Установка пакетов Virtual Security (см. стр. 15)
    - Auth (см. стр. 15)
    - Broker (см. стр. 16)
    - Admin (см. стр. 16)
    - Web (см. стр. 17)
    - vbalancer (см. стр. 18)
    - Proxy (см. стр. 18)
- Настройка агента вычислительного узла (см. стр. 18)
  - Установка пакетов и настройка сервиса (см. стр. 18)

Важно

Для корректной работы Базис.Virtual Security рекомендуется использовать доменную адресацию на узлах.

Введение

Документация актуальна для кластера Virtual Security на базе узлов с ОС Astra Linux Special Edition 1.7.4 (Smolensk).

Предоставленные в документации команды следует копировать полностью и, если не указано иное, выполнять от имени пользователя root.

Установка поделена на две основные части:

- Установка управляющей части VS;
- Установка агентов VS на вычислительных узлах.

Установка управляющих компонентов

Предварительная настройка

Для корректной работы основных компонентов VS необходимо настроить отдельные части операционной системы.

Добавление репозиториев VS

В качестве источника пакетов настройте следующие репозитории:

```
cat<<EOF> /etc/apt/sources.list
# Основной репозиторий
deb https://dl.astralinux.ru/astra/stable/1.7_x86-64/repository-main/     1.7_x86-64
main contrib non-free

# Оперативные обновления основного репозитория
deb https://dl.astralinux.ru/astra/stable/1.7_x86-64/repository-update/   1.7_x86-64
main contrib non-free

# Базовый репозиторий
deb https://dl.astralinux.ru/astra/stable/1.7_x86-64/repository-base/     1.7_x86-64
main contrib non-free

# Последнее срочное обновление (если доступно):
deb https://dl.astralinux.ru/astra/stable/1.7_x86-64/last/repository-update/ 1.7_x86-64
main contrib non-free

# Репозиторий пакетов Vitrual Security
deb [trusted=yes] https://pkgs.basistech.ru/artifactory/tvs-deb astra-1.7 main

EOF
```

Добавьте параметры аутентификации для репозитория пакетов Vitrual Security:

```
cat <<EOF > /etc/apt/auth.conf.d/bvs.conf
machine pkgs.basistech.ru
login vcore-public
password vcore-public
EOF
```

Обновите данные репозиториев:

```
apt update
```

Настройка клиента точного времени

Для нормального взаимодействия между частями платформы важно, чтобы часы во всех узлах были засинхронизированы через единый NTP-сервер. Рекомендуется использование локального NTP-сервера, однако допускается использование публичных NTP-пулов. В качестве клиента NTP рекомендуется использовать chrony.

Установите клиент chrony:

```
apt install chrony
```

apt-get может предложить удалить другой сервис точного времени, соглашайтесь на удаление.

Настройка управляющей части

> Для корректной работы сервисов необходимо наличие двух сетевых интерфейсов. В данной инструкции используются: bvs-test и bvs-stand.

Установка PostgreSQL

В качестве СУБД для хранения данных VS использует СУБД PostgreSQL.

Установите пакет `postgresql-server`:

```
apt install postgresql postgresql-11
```

Откройте файл `/etc/postgresql/11/main/postgresql.conf` и измените следующую строку:

```
listen_addresses = 'bvs-test,bvs-stand'
```

Откройте файл `/etc/postgresql/11/main/pg_hba.conf` и в конец добавьте следующее:

```
host    vs              vs              bvs-stand          md5
host    vs              vs              bvs-test           md5
```

Запустите СУБД:

```
systemctl start postgresql
```

Добавьте СУБД в автозагрузку:

```
systemctl enable postgresql
```

Зайдите в окружение postgres:

```
su - postgres
```

Создайте все объекты внутри СУБД:

```
create user vs with password 'vs';
create database vs owner vs;
\c vs;
create schema vsm authorization vs;
create schema idp authorization vs;
exit;
```

Выйдите из окружения postgres:

```
exit
```

Проверьте вход в базу vs пользователем vs через указанный адрес (пароль будет запрошен интерактивно):

```
psql -h bvs-test -U vs
```

Если вы смогли войти в окружение пользователя vs внутри psql, значит, аутентификация была настроена верно.

Установка Network Manager

Установите сервис:

```
apt install network-manager
```

Установка пакетов Virtual Security

Auth

Установите сервис:

```
apt install virtual-security-auth
```

Настройте конфигурационный файл - `/opt/bvs/auth/config/idp.conf`:

- Укажите базовый URL для сервиса:

  ```
  hostname-url=http://bvs-stand:8082
  hostname-admin-url=http://bvs-stand:8080
  ```

- Укажите адрес и порт прослушивания:

```
http-host=bvs-test
http-port=8082
```

- Укажите адрес подключения к postgresql:

```
db-url-host=bvs-test
```

Настройте конфигурационный файл - `/opt/bvs/auth/config/application.properties`:

```
quarkus.mod-cluster-client.enabled=true
```

Запустите сервис:

```
systemctl start virtual-security-auth
```

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-auth
```

Broker

Установите сервис:

```
apt install vcore-broker
```

Настройте конфигурационный файл - /opt/vcore/broker/config/application.properties:

```
quarkus.http.host=bvs-test
quarkus.http.port=8071
quarkus.mod-cluster-client.enabled=true
quarkus.mod-cluster-client.services.api.host=bvs-test
quarkus.mod-cluster-client.services.api.port=8071
 ...
provider.transport.use-multicast-discovery=true
...
provider.transport.multicast-interface-address=bvs-test
provider.transport.multicast-group=231.7.7.7
provider.transport.multicast-port=9876
provider.transport.static-discovery-addresses=tcp://bvs-test:61617
provider.transport.bind-address=bvs-test
```

Запустите сервис:

```
systemctl start vcore-broker
```

Добавьте сервис в автозагрузку:

```
systemctl enable vcore-broker
```

Admin

Установите сервис:

```
apt install virtual-security-admin
```

Настройте конфигурационный файл - `/opt/bvs/admin/config/application.yaml`:

```
client:
    base-uri: http://bvs-stand:8080
```

```
    realm: master
    client-id: security-admin-console
  security-proxy:
    base-uri: http://bvs-stand:8080
  provider:
    messaging:
      count: 2
      broker-uri: tcp://bvs-test:61616?
clientFailureCheckPeriod=5000&retryInterval=1000&reconnectAttempts=5&callTimeout=3000

...

http:
    host: bvs-test
    port: 8081
    root-path: /api
    cors:
      ~: true
      origins: http://bvs-stand:8080

...

datasource:
    db-kind: postgresql
    username: vs
    password: vs
    jdbc:
      url: jdbc:postgresql://bvs-test:5432/vs
      max-size: 10

...

 mod-cluster-client:
    enabled: true
```

Запустите сервис:

```
systemctl start virtual-security-admin
```

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-admin
```

Web

Установите сервис:

```
apt install virtual-security-web
```

Настройте конфигурационный файл - `/opt/bvs/web/config/application.properties`:

```
quarkus.http.host=bvs-test
...
quarkus.oidc.auth-server-url=http://bvs-stand:8080/realms/master
...
quarkus.mod-cluster-client.enabled=true
...
quarkus.mod-cluster-client.services.api.host=bvs-test
quarkus.mod-cluster-client.services.api.port=8080
quarkus.mod-cluster-client.services.api.contexts=/webadmin
```

Запустите сервис:

```
systemctl start virtual-security-web
```

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-web
```

vbalancer

Установите сервис:

```
apt install basis-vbalancer
```

Настройте конфигурационный файл - /etc/default/basis-vbalancer:

```
BALANCER_BIND_ADDR="bvs-stand"
BALANCER_PRIVATE_BIND_ADDR="bvs-test"
```

Запустите сервис:

```
systemctl start basis-vbalancer
```

Добавьте сервис в автозагрузку:

```
systemctl enable basis-vbalancer
```

Proxy

Установите сервис:

```
apt install virtual-security-proxy
```

Настройте конфигурационный файл - /opt/bvs/proxy/config/application.properties:

```
quarkus.http.port=8099
...
quarkus.http.host=bvs-test
...
quarkus.mod-cluster-client.enabled=true
...
proxy.admin.url=http://bvs-stand:8080
proxy.security.url=http://bvs-stand:8080
```

Запустите сервис:

```
systemctl start virtual-security-proxy
```

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-proxy
```

Настройка агента вычислительного узла

В каждом вычислительном узле должен быть агент Virtual Security, который контролирует выполнение всех операций безопасности.

> **Примечание**
>
> В окружениях Alt Linux агент VS совместим только с "чистой" установкой libvirt и гипервизора. Не используйте агент VS с другими агентами управления виртуализацией. vCore это не касается, так как он использует собственную ОС.

Установка пакетов и настройка сервиса

Установите следующие пакеты:

```
apt install basis-openjdk virtual-security-agent
```

Настройте конфигурационный файл - `/opt/bvs/agent/config/application.properties` :

```
quarkus.http.host=bvs-stand
...
agent.provider.jms.broker-uris=tcp://bvs-stand:61616?
clientFailureCheckPeriod=5000&retryInterval=1000&reconnectAttempts=5&callTimeout=3000
```

Запустите сервис:

```
systemctl start virtual-security-agent
```

После этого агент Virtual Security будет добавлен в управляющую часть.

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-agent
```

▪ Инсталляция с поддержкой высокой доступности программного модуля Базис.Virtual Security

Инсталляция с поддержкой высокой доступности AltLinux

- Управляющая часть (см. стр. 20)
  - Предварительная настройка узлов (см. стр. 20)
    - Настройка параметров ОС (см. стр. 20)
    - Настройка точного времени (см. стр. 20)
    - Установка сервисов (см. стр. 20)
    - Добавление репозиториев VS (см. стр. 20)
    - Настройка доменных имен узлов кластера (см. стр. 21)
    - Установка и настройка Pacemaker (см. стр. 21)
    - Настройка аутентификации и запуск компонентов кластера (см. стр. 21)
    - Перезапуск узлов в кластере (см. стр. 22)
    - Включение функций фенсинга узлов (см. стр. 22)
    - Создание VIP-ресурсов для сервисов (см. стр. 23)
  - Установка PostgreSQL (см. стр. 24)
    - Установка пакетов (см. стр. 24)
  - Настройка кластера PostgreSQL (см. стр. 24)
    - Настройка slave-узлов (см. стр. 25)
    - Настройка кластера Pacemaker (см. стр. 26)
    - Создание БД для сервисов Virtual Security (см. стр. 27)
    - Установка Virtual Security (см. стр. 28)
    - Auth (см. стр. 28)
    - Admin (см. стр. 29)
    - Web (см. стр. 29)
    - Broker (см. стр. 30)
    - vbalancer (см. стр. 30)
  - Настройка кластера Pacemaker для Virtual Security (см. стр. 31)
- Вычислительная часть (см. стр. 31)

Важно

Для корректной работы Базис.Virtual Security рекомендуется использовать доменную адресацию на узлах.

Примечание

Документация актуальна для кластера Virtual Security на базе узлов с ОС Alt linux 10.1
Предоставленные в документации команды следует копировать полностью и, если не указано иное, выполняются от имени пользователя root.

Управляющая часть

Примечание

Все процессы описанные ниже производятся на управляющих узлах кластера.

Предварительная настройка узлов

Настройка параметров ОС

Настройка системных параметров.

Для работы VIP-адресов включите возможность привязки сервисов к отсутствующему на данный момент IP-адресу в ОС:

```
echo net.ipv4.ip_nonlocal_bind=1 > /etc/sysctl.d/99-cluster.conf; \
systemctl restart systemd-sysctl; \
sysctl net.ipv4.ip_nonlocal_bind
```

Перезагрузите узел командой:

```
reboot
```

Настройка точного времени

Во всех узлах кластера требуется настроить NTP-клиент с синхронизацией через один и тот же сервер или пул серверов.

Обновите пакеты до последних версий:

```
apt-get update; apt-get dist-upgrade
```

Установка сервисов

Установите сервис chrony:

```
apt-get install chrony
```

apt-get может предложить удалить другой сервис точного времени, соглашайтесь на удаление.

Установите, если отсутствует, пакет для работы с https репозиториями:

```
apt-get install apt-https
```

В случае использования Alterator WWW frontend server, остановите сервис или настройте его на другой порт:

```
systemctl stop ahttpd
systemctl disable ahttpd
```

Добавление репозиториев VS

Для получения пакетов Virtual Security нужно добавить официальный репозиторий для всех узлов (управляющих и вычислительных). Для этого в файле `/etc/apt/sources.list.d/bvs.list` укажите адреса репозиториев (имя репозитория p10 формальное, пакеты совместимы со всеми релизами ALT):

```
#Altlinux P10 BVS repo
rpm https://bvs-public:bvs-public@pkgs.basistech.ru/artifactory/bvs-alt-rpm p10/x86_64
classic
rpm https://bvs-public:bvs-public@pkgs.basistech.ru/artifactory/bvs-alt-rpm p10/noarch
classic
```

Обновите данные репозиториев:

```
apt-get update
```

Настройка доменных имен узлов кластера

> **Важно**
>
> Для обеспечения работы кластера сервисов Virtual Security рекомендуется использовать имена узлов вместо IP-адресов. Желательно настроить отдельный DNS-сервер для этой цели. Однако, если это не является возможным, то допускается добавить соответствующие записи в файл `/etc/hosts`.

Для примера примем следующие данные узлов, которые будут использоваться в дальнейшем:

```
первый узел кластера:
имя узла: control-1
IP-адрес: 10.18.0.253
второй узел кластера:
имя узла: control-2
IP-адрес: 10.18.0.239
третий узел кластера:
имя узла: control-3
IP-адрес: 10.18.0.243
```

Далее установите имена для двух VIP-адресов: один для PostgreSQL и другой для сервисов Virtual Security. Для наглядности рассмотрим следующий пример:

```
VIP-адрес для PostgreSQL:
имя для адреса: pgsql-vip
IP-адрес: 10.18.0.236
VIP-адрес для сервисов Virtual Security:
имя для адреса: bvs-vip
IP-адрес: 10.18.0.226
```

В соответствии с приведенными данными, файл `/etc/hosts` будет иметь следующий вид:

```
10.18.0.253   control-1
10.18.0.239   control-2
10.18.0.243   control-3
10.18.0.236   pgsql-vip
10.18.0.226   bvs-vip
```

Установка и настройка Pacemaker

Для установки Pacemaker необходимо выполнить ряд операций.

Установите пакеты Pacemaker:

```
apt-get install pacemaker pcs
```

Запустите сервис pcsd и добавьте его в автозапуск:

```
systemctl start pcsd.service; \
systemctl enable pcsd.service
```

Настройка аутентификации и запуск компонентов кластера

Для создания кластера Pacemaker необходимо предварительно настроить аутентификацию. Установите общий пароль для пользователя hacluster, который будет использован во всех узлах кластера:

```
passwd hacluster
```

Для выполнения команд pcs достаточно выполнить их только на одном узле, так как изменения будут автоматически синхронизированы на остальные узлы.

Выполните проверку подлинности узлов кластера. При вводе команды укажите учетные данные пользователя и пароль, которые были настроены на предыдущем шаге:

```
pcs host auth control-1 control-2 control-3
```

Создайте кластер:

```
pcs cluster setup cluster control-1 control-2 control-3
```

> Сообщения `'No addresses specified for host...'` не являются ошибками, их можно игнорировать.

Запустите кластер:

```
pcs cluster start --all
```

Проверьте доступность кластера:

```
pcs status
```

Перезапуск узлов в кластере

> **Важно**
>
> Рекомендуется не включать автозапуск для сервисов Pacemaker. При нормальной перезагрузке узла кластера, необходимо выполнить некоторые простые команды. Кроме того, узлы следует перезапускать по одному.

Остановите узел кластера, который вы хотите перезапустить:

```
pcs cluster stop control-1
```

Дождитесь переезда всех ресурсов с этого узла на другие через команду:

```
pcs status
```

После этого штатно перезапустите узел. Когда узел будет доступен по сети, снова запустите сервисы Pacemaker в нём:

```
pcs cluster start control-1
```

Включение функций фенсинга узлов

Включение STONITH и определение политики при отсутствии кворума

Технология STONITH в Pacemaker используется для исключения из эксплуатации нестабильных или неработающих узлов. Она основана на использовании независимого канала, например, протокола IPMI[2] для BMC[3], для перезапуска или выключения проблемного узла. Для правильной работы STONITH необходимо наличие как минимум трех узлов и активация политики отсутствия кворума. Процесс исключения узла с последующей миграцией ресурсов называется фенсингом (fencing).

Для включения кворума и STONITH укажите параметры сервиса:

```
pcs property set stonith-enabled=true
```

Кворум в Pacemaker проверяет количество работающих узлов в кластере. Если количество работающих узлов превышает 50%, то Pacemaker считает, что кластер находится в рабочем состоянии и имеет кворум. Однако, если это условие не выполняется, Pacemaker применяет одну из политик отсутствия кворума, например остановка всех узлов (stop), запрет на перемещение ресурсов без остановки ресурсов (freeze), выключение всех узлов (suicide) или продолжение работы всех ресурсов (ignore). Для сервисов Virtual Security рекомендуется использовать политику остановки (stop):

2 https://ru.wikipedia.org/wiki/Intelligent_Platform_Management_Interface
3 https://www.servethehome.com/explaining-the-baseboard-management-controller-or-bmc-in-servers/

```
pcs property set no-quorum-policy=stop
```

В таком варианте безопасно временно терять один узел из трёх, при потере двух узлов Pacemaker выключит все ресурсы.

Дополнительно рекомендуется настроить STONITH так, чтобы по умолчанию он выключал вышедшие из строя узлы кластера, вместо попыток их перезапуска:

```
pcs property set stonith-action=poweroff
```

Добавление данных IPMI для узлов в кластере

В качестве стандартного метода фенсинга рекомендуется использовать протокол IPMI.

На всех узлах, где запущены компоненты Pacemaker, установите пакет фенсинга для IPMI:

```
apt-get install fence-agents-ipmilan
```

Для включения ресурсов для фенсинга через IPMI необходимо использовать команду `pcs stonith`. Для каждого узла нужен свой ресурс stonith. Предлагаемые настройки для ресурса:

```
pcs stonith create ipmi_control-1 fence_ipmilan pcmk_host_list="control-1" ip=10.16.10.3
 lanplus=true username=ADMIN password=Rp56Yhre auth=MD5
```

Нужно обратить внимание на следующее:

- Так как для каждого узла требуется создать ресурс stonith, после create в имени ресурса укажите, к какому узлу относится ресурс и его укажите в параметре `pcmk_host_list`;
- В ip укажите адрес IPMI-сервера. Если используется нестандартный порт, то вместо ip укажите параметр ipport и укажите порт через двоеточие после адреса (напр. 10.23.10.3:6623);
- По умолчанию предлагается использование IPMI 2.0 через параметр lanplus;
- Остальные параметры определяют данные аутентификации со стандартным методом хэширования пароля MD5.

По умолчанию возможна ситуация, когда ресурс stonith может запуститься на том же узле, BMC которого контролируется этим ресурсом. Для запрета такой ситуации добавьте соответствующее ограничение для каждого из ресурсов stonith:

```
pcs constraint location ipmi_control-1 avoids control-1=INFINITY
```

Проверьте корректность запуска ресурса:

```
pcs status

...
* ipmi_control-1    (stonith:fence_ipmilan):     Started control-2

...
```

Создание VIP-ресурсов для сервисов

Для единой точки входа в СУБД необходимо настроить ресурсы Pacemaker с VIP-адресами. Выделенный узел для VIP определит начальный master-узел PostgreSQL далее.

Добавьте VIP-ресурс PGSQL в Pacemaker:

```
pcs resource create pgsql-vip ocf:heartbeat:IPaddr2 ip=10.18.0.236 cidr_netmask=32 op
monitor interval=10s
```

Создайте VIP-ресурс для Virtual Security:

```
pcs resource create bvs-pri-ip ocf:heartbeat:IPaddr2 ip=10.18.0.226 cidr_netmask=32 op
monitor interval=10s --group bvs
```

Получите статус ресурсов и определите, на каком узле запустился VIP-адрес:

```
pcs status
...
 * pgsql-vip    (ocf::heartbeat:IPaddr2):    Started control-1
  * Resource Group: bvs:
    * bvs-pri-ip    (ocf::heartbeat:IPaddr2):    Started control-1
```

Если vip ресурсы запустились на разных узлах, переместите их на один следующей командой:

```
pcs resource move pgsql-vip pcs control-1
```

Установка PostgreSQL

Установка пакетов

В качестве СУБД для хранения данных VS использует СУБД PostgreSQL. На текущий момент поддерживается версия 12. Установите эту версию СУБД:

```
apt-get install postgresql12 postgresql12-server resource-agents-paf
```

Настройка кластера PostgreSQL

Для настройки кластера PostgreSQL необходимо выполнить ряд операций.

Настройка master-узла

Важно

Следующие операции выполняются на master-узле кластера. Master-узел определяется по тому, в каком узле поднялся VIP-ресурс pgsql-vip в Pacemaker. Далее считается, что master-узлом является control-1.

Инициализируйте БД:

```
/etc/init.d/postgresql initdb
```

Зайдите под пользователем postgres:

```
su - postgres
```

Добавьте, по необходимости, рабочий shell учетной записи postgres:

```
usermod postgres -s /bin/bash
```

Укажите имя узла и имя VIP-адреса PostgreSQL, в примере pgsql-vip, в `~postgres/listen.conf`:

```
cat <<EOF> ~postgres/listen.conf
listen_addresses = 'pgsql-vip, control-1'
EOF
```

Укажите имя узла и имя VIP-адреса PostgreSQL в `~postgres/standby.conf`:

```
cat <<EOF > ~postgres/standby.conf
primary_conninfo = 'host=pgsql-vip application_name=control-1'
EOF
```

Добавьте следующие параметры в `~postgres/data/postgresql.conf`:

```
cat <<EOF>> ~postgres/data/postgresql.conf
hba_file = '/var/lib/pgsql/pg_hba.conf'
wal_keep_segments = 32
include = '../standby.conf'
include = '../listen.conf'
```

```
EOF
```

Сформируйте конфигурацию `pg_hba.conf`, вписав точное имя узла и имя vip адреса pgsql, в примере pgsql-vip:

```
cat <<EOF > ~postgres/pg_hba.conf
local all         all                     trust
host  all         all      127.0.0.0/8    trust
host  all         all      ::1/128        trust
# forbid self-replication from VIP
host  replication postgres pgsql-vip      reject
# forbid self-replication its own IP
local replication all                     reject
host  replication all      control-1      reject
host  replication all      127.0.0.0/8    reject
host  replication all      ::1/128        reject
# allow any standby connection
host  replication postgres 10.18.0.0/24   trust
host vs vs 10.18.0.0/24               md5
EOF
```

Здесь нужно:

- В строке "host replication postgres pgsql-vip reject" указать VIP-адрес или имя домена VIP-адреса;
- В строке "host replication all control-1 reject" указать фактический адрес узла или имя домена для этого адреса;
- В строке "host vs vs 10.18..0/24 md5" нужно указать подсеть, через которую клиенты (в частности, сервисы Virtual Security) могут подключиться к СУБД. Возможно указание адресов сервисов по строке явно через указание подсети /32.

Осуществите выход из под пользователя postgres:

```
exit
```

Запустите PostgreSQL и проверьте статус сервиса:

```
systemctl start postgresql

systemctl list-units |grep postgre
postgresql.service                                          loaded active running
PostgreSQL database server
```

Настройка slave-узлов

> **Важно**
>
> Следующие операции выполняются на slave-узлах.

Зайдите под пользователем postgres:

```
su - postgres
```

Добавьте, по необходимости рабочий shell учетной записи:

```
usermod postgres -s /bin/bash
```

Удалите содержимое кластера СУБД:

```
rm -rf ~postgres/data/*
```

Укажите имя узла и имя VIP-адреса PostgreSQL, в примере pgsql-vip, в `~postgres/listen.conf`:

```
cat <<EOF> ~postgres/listen.conf
listen_addresses = 'pgsql-vip, control-2'
EOF
```

Укажите имя узла и имя VIP-адреса PostgreSQL, в примере pgsql-vip, в `~postgres/standby.conf`:

```
cat <<EOF > ~postgres/standby.conf
primary_conninfo = 'host=pgsql-vip application_name=control-2'
EOF
```

Сформируйте конфигурацию `pg_hba.conf`, вписав точное имя узла и имя vip адреса pgsql:

```
cat <<EOF > ~postgres/pg_hba.conf
local all         all                     trust
host  all         all      127.0.0.0/8    trust
host  all         all      ::1/128        trust
# forbid self-replication from VIP
host  replication postgres pgsql-vip      reject
# forbid self-replication its own IP
local replication all                     reject
host  replication all      control-2      reject
host  replication all      127.0.0.0/8    reject
host  replication all      ::1/128        reject
# allow any standby connection
host  replication postgres 10.18.0.0/24   trust
host vs vs 10.18.0.0/24                 md5
EOF
```

Здесь нужно:

- В строке "host replication postgres pgsql-vip reject" указать VIP-адрес или имя домена VIP-адреса;
- В строке "host replication all control-2 reject" указать фактический адрес узла или имя домена для этого адреса;
- В строке "host vs vs 10.18.0.0/24 md5" нужно указать подсеть, через которую клиенты (в частности, сервисы Virtual Security) могут подключиться к СУБД. Возможно указание адресов сервисов по строке явно через указание подсети /32.

Создайте репликацию из главного (master) узла в текущий (slave) узел:

```
pg_basebackup -h pgsql-vip -D ~postgres/data/ -X stream -P
```

Пометьте узел как slave:

```
touch ~postgres/standby.signal
```

Выйдите из пользователя postgres:

```
exit
```

Запустите PostgreSQL:

```
systemctl start postgresql
```

Проверьте статус сервиса:

```
systemctl list-units |grep postgre
postgresql.service                                          loaded active running
PostgreSQL database server
```

Повторите операции на следующем slave-узле.

Настройка кластера Pacemaker

Настройте автоматическое переключение узлов с помощью Pacemaker.

Отключите PostgreSQL. Сначала на slave-узлах, затем на master:

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

Создайте ресурс postgresql:

```
pcs resource create pgsqld ocf:heartbeat:pgsqlms bindir=/usr/bin pgdata=/var/lib/pgsql/
data op start timeout=60s op stop timeout=60s op promote timeout=30s op demote
timeout=120s op monitor interval=15s timeout=10s role="Master" op monitor interval=16s
timeout=10s role="Slave" op notify timeout=60s promotable notify=true
```

Добавьте метадату количества master-узлов в кластере:

```
pcs resource meta pgsqld master-max=1
```

Настройте зависимости ресурсов:

```
pcs constraint colocation add pgsql-vip with master pgsqld-clone INFINITY
pcs constraint order promote pgsqld-clone then start pgsql-vip symmetrical=false
 kind=Mandatory
pcs constraint order demote pgsqld-clone then stop pgsql-vip symmetrical=false
 kind=Mandatory
```

Проверьте статус кластера postgresql в pcs. На одном из узлов должен быть Master-узел:

```
pcs status
Cluster name: cluster
Status of pacemakerd: 'Pacemaker is running' (last updated 2023-09-15 16:10:22 +03:00)
Cluster Summary:
  * Stack: corosync
  * Current DC: control-1 (version 2.1.5-alt1-a3f44794f) - partition with quorum
  * Last updated: Fri Sep 15 16:10:23 2023
  * Last change:  Fri Sep 15 15:42:39 2023 by hacluster via crmd on control-1
  * 3 nodes configured
  * 15 resource instances configured

Node List:
  * Online: [ control-1 control-2 control-3 ]

Full List of Resources:
  * pgsql-vip   (ocf:heartbeat:IPaddr2):     Started control-1
  * Resource Group: bvs:
    * bvs-pri-ip    (ocf:heartbeat:IPaddr2):     Started control-1
    * tvs-balancer  (systemd:tvs-balancer):    Started control-1
  * Clone Set: pgsqld-clone [pgsqld] (promotable):
    * Promoted: [ control-1 ]
    * Unpromoted: [ control-2 control-3 ]
```

Создание БД для сервисов Virtual Security

В выводе команды `pcs status` определяем, что роль Master у узла control-1:

```
 Promoted: [ control-1 ]
```

Настройки выполняются на узле с ролью Master.

Зайдите под пользователем postgres:

```
su - postgres
```

Запустите терминальный клиент для работы с БД:

```
psql
```

Внутри сессии psql создайте пользователя vs с указанным паролем:

```
create user vs with password 'УКАЖИТЕПАРОЛЬ';
```

Создайте БД vs:

```
create database vs owner vs;
```

Выйдите из консольного клиента:

```
exit
```

Проверьте подключение пользователя vs к созданной БД:

```
psql -h pgsql-vip -U vs
```

Создайте схемы vsm и ids:

```
create schema vsm AUTHORIZATION vs;
create schema idp AUTHORIZATION vs;
```

Выйдите из терминального клиента БД:

```
\q
```

Осуществите выход из под пользователя postgres:

```
exit
```

Установка Virtual Security

Установите пакеты Virtual Security на каждом узле:

```
apt-get install basis-openjdk basis-vbalancer virtual-security-auth virtual-security-
admin virtual-security-web
```

Запустите конфигурирование модулей Virtual Security на каждом узле, в качестве адреса БД укажите имя vip адреса PostgreSQL, в качестве сервера балансировки и URL-системы, укажите VIP адрес для сервисов Virtual Security.

Настройка производится на узле с активным ресурсом bvs-pri-ip. Для его вычисления запустите команду:

```
pcs status
...
Resource Group: bvs
     bvs-pri-ip    (ocf::heartbeat:IPaddr2):    Started control-1
```

В нашем случае это узел control-1.

Настройте сервисы Virtual Security.

Auth

Настройте конфигурационный файл - `/opt/bvs/auth/lib/config/idp.conf`:

- Укажите базовый URL для сервиса:

  ```
  hostname-url=http://bvs-vip:8080
  hostname-admin-url=http://bvs-vip:8080
  ```

- Укажите адрес и порт прослушивания:

  ```
  http-host=bvs-test
  http-port=8082
  ```

- Укажите адрес подключения к postgresql:

  ```
  db-url-host=bvs-test
  ```

Включите регистрацию сервиса в сервисе балансировки (vbalancer) путем включения параметра в `/opt/bvs/auth/lib/config/application.properties`:

```
quarkus.mod-cluster-client.enabled=true
```

Запустите сервис:

```
systemctl start virtual-security-auth
```

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-auth
```

Admin

Настройте конфигурационный файл - `/opt/bvs/admin/config/application.properties`:

```
client:
    base-uri: http://bvs-vip:8080
    realm: master
    client-id: security-admin-console
  security-proxy:
    base-uri: http://bvs-vip:8080/proxy
  provider:
    messaging:
      count: 2
      broker-uri: tcp://bvs-test:61616?
clientFailureCheckPeriod=5000&retryInterval=1000&reconnectAttempts=5&callTimeout=3000

...

http:
    host: bvs-test
    port: 8081
    root-path: /api
    cors:
      ~: true
      origins: http://bvs-vip:8080

...

datasource:
    db-kind: postgresql
    username: vs
    password: vs
    jdbc:
      url: jdbc:postgresql://bvs-test:5432/vs
      max-size: 10

...

 mod-cluster-client:
    enabled: true
```

Запустите сервис:

```
systemctl start virtual-security-admin
```

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-admin
```

Web

Настройте конфигурационный файл - `/opt/bvs/web/config/application.properties`:

```
quarkus.http.host=bvs-test
```

```
...
quarkus.oidc.auth-server-url=http://bvs-vip:8080/realms/master
...
quarkus.mod-cluster-client.enabled=true
...
quarkus.mod-cluster-client.services.api.host=bvs-test
quarkus.mod-cluster-client.services.api.port=8080
quarkus.mod-cluster-client.services.api.contexts=/webadmin
```

Запустите сервис:

```
systemctl start virtual-security-web
```

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-web
```

Broker

Настройте конфигурационный файл - `/opt/vcore/broker/config/application.properties`:

```
quarkus.mod-cluster-client.enabled=true
quarkus.mod-cluster-client.services.api.host=bvs-test
provider.transport.use-multicast-discovery=true
provider.transport.multicast-interface-address=bvs-test
provider.transport.multicast-group=231.7.7.7
provider.transport.multicast-port=9876
provider.transport.static-discovery-addresses=tcp://bvs-test:61617
provider.transport.bind-address=bvs-test
```

Запустите сервис:

```
systemctl start vcore-broker
```

Добавьте сервис в автозагрузку:

```
systemctl enable vcore-broker
```

vbalancer

Настройте конфигурационный файл - /etc/sysconfig/basis-vbalancer:

```
BALANCER_BIND_ADDR="bvs-vip"
BALANCER_PRIVATE_BIND_ADDR="bvs-test"
```

Запустите сервис:

```
systemctl start basis-vbalancer
```

Добавьте сервис в автозагрузку:

```
systemctl enable basis-vbalancer
```

Остановите сервисы Virtual Security:

```
systemctl disable --now virtual-security-auth virtual-security-admin virtual-security-
web basis-vbalancer
```

Переместить ресурс bvs-pri-ip на настраиваемый узел:

```
pcs resource move bvs-pri-ip control-2
```

Проверьте корректность перемещения ресурса:

```
pcs status

...
Resource Group: bvs
     bvs-pri-ip    (ocf::heartbeat:IPaddr2):    Started control-2
...
```

Настройка кластера Pacemaker для Virtual Security

Отключите зависимости vs-systemd на всех узлах. Скомментируйте строку `#Wants=`:

```
systemctl edit --full basis-vbalancer
systemctl edit --full virtual-security-auth
systemctl edit --full virtual-security-admin
systemctl edit --full virtual-security-web
```

Отключите контроль systemd над Virtual Security:

```
systemctl disable --now basis-vbalancer virtual-security-auth virtual-security-admin
virtual-security-web
```

Создайте ресурсы для bvs:

```
pcs resource create basis-vbalancer systemd:basis-vbalancer --group bvs

pcs resource create virtual-security-admin systemd:virtual-security-admin clone
pcs resource create virtual-security-auth systemd:virtual-security-auth clone
pcs resource create virtual-security-web systemd:virtual-security-web clone
#
pcs constraint order virtual-security-auth-clone then virtual-security-admin-clone
pcs constraint order virtual-security-auth-clone then virtual-security-web-clone
pcs constraint order virtual-security-admin-clone then virtual-security-web-clone
pcs constraint order virtual-security-web-clone then basis-vbalancer
#
pcs constraint colocation add bvs-pri-ip with basis-vbalancer INFINITY
pcs constraint order pgsqld-clone then virtual-security-auth-clone
```

Проверьте статус кластера:

```
pcs status
```

Вычислительная часть

> **Примечание**
>
> Все процессы описанные ниже производятся на вычислительных узлах кластера.

Процесс установки и настройки вычислительной части не отличается от простой инсталляции. Данная процедура подробно описана в разделе - "Настройка агента вычислительного узла (см. стр. 12).

Инсталляция с поддержкой высокой доступности AstraLinux

- Управляющая часть (см. стр. 32)
  - Предварительная настройка (см. стр. 32)
    - Настройка системных параметров (см. стр. 32)
    - Настройка клиента точного времени: (см. стр. 33)
    - Настройка доменных имен узлов кластера (см. стр. 33)
  - Установка и настройка Pacemaker (см. стр. 34)
    - Установка пакетов Pacemaker (см. стр. 34)
    - Настройка аутентификации и запуск компонентов кластера (см. стр. 34)
    - Перезапуск узлов в кластере (см. стр. 35)
    - Включение функций фенсинга узлов (см. стр. 35)
    - Создание VIP-ресурсов для сервисов (см. стр. 36)
  - Установка PostgreSQL (см. стр. 37)

- Установка пакетов (см. стр. 37)
- Настройка кластера PostgreSQL (см. стр. 37)
  - Настройка master-узла (см. стр. 37)
  - Настройка slave-узлов (см. стр. 39)
  - Настройка кластера Pacemaker (см. стр. 39)
  - Создание БД для сервисов Virtual Security (см. стр. 40)
- Настройка сервисов Virtual Security (см. стр. 41)
  - Auth (см. стр. 41)
  - Admin (см. стр. 42)
  - Web (см. стр. 42)
  - Broker (см. стр. 43)
  - vbalancer (см. стр. 43)
- Настройка кластера Pacemaker для Virtual Security (см. стр. 44)
- Вычислительная часть (см. стр. 45)

Важно

Для корректной работы Базис.Virtual Security рекомендуется использовать доменную адресацию на узлах.

Примечание

Документация актуальна для кластера Virtual Security на базе узлов с ОС Astra Linux Special Edition 1.7.4 (Smolensk).
Предоставленные в документации команды следует копировать полностью и, если не указано иное, выполнять от имени пользователя root.

Управляющая часть

Примечание

Все процессы описанные ниже производятся на управляющих узлах кластера.

Предварительная настройка

Для корректной работы Virtual Security требуются некоторые изменения в настройках ОС. Все указанные в этом разделе команды должны быть выполнены во всех узлах кластера.

Настройка системных параметров

Для работы VIP-адресов включите возможность привязки сервисов к отсутствующему на данный момент IP-адресу в ОС:

```
echo net.ipv4.ip_nonlocal_bind=1 > /etc/sysctl.d/99-cluster.conf; \
systemctl restart systemd-sysctl; \
sysctl net.ipv4.ip_nonlocal_bind
```

Настройка репозиториев

Для работы с https репозиториями, установите пакет `ca-certificates`:

```
apt-get install ca-certificates
```

Примечание

По умолчанию, в системе используется источник пакетов, в виде cdrom диска.

В качестве источника пакетов настройте следующие репозитории:

```
cat<<EOF> /etc/apt/sources.list
# Основной репозиторий
deb https://dl.astralinux.ru/astra/stable/1.7_x86-64/repository-main/     1.7_x86-64
main contrib non-free
```

```
# Оперативные обновления основного репозитория
deb https://dl.astralinux.ru/astra/stable/1.7_x86-64/repository-update/   1.7_x86-64
main contrib non-free

# Базовый репозиторий
deb https://dl.astralinux.ru/astra/stable/1.7_x86-64/repository-base/     1.7_x86-64
main contrib non-free

# Последнее срочное обновление (если доступно):
deb https://dl.astralinux.ru/astra/stable/1.7_x86-64/uu/last/repository-update/
1.7_x86-64 main contrib non-free

# Репозиторий пакетов Vitrual Security
deb [trusted=yes] https://maven.tionix.ru/artifactory/tvs-deb astra-1.7 main

EOF
```

Добавьте параметры аутентификации для репозитория пакетов Vitrual Security:

```
cat <<EOF > /etc/apt/auth.conf.d/bvs.conf
machine maven.tionix.ru
login vcore-public
password vcore-public
EOF
```

Настройка клиента точного времени:

Для нормального взаимодействия между частями платформы важно, чтобы часы во всех узлах были синхронизированы через единый NTP-сервер. Рекомендуется использование локального NTP-сервера, однако допускается использование публичных NTP-пулов. В качестве клиента NTP рекомендуется использовать chrony.

Установите клиент chrony:

```
apt update
apt-get install chrony
```

Установщик может предложить удалить другой сервис точного времени, соглашайтесь на удаление.

В качестве дополнения проверьте часовой пояс, указанный в ОС:

```
timedatectl | grep zone
```

Если указанный пояс не совпадает с вашим, то нужно указать корректный. Например, для Москвы (GMT+3):

```
timedatectl set-timezone Europe/Moscow
```

Убедитесь, что пояс успешно изменился с помощью команды из предыдущего шага.

Настройка доменных имен узлов кластера

Для кластера сервисов Virtual Security необходимо использовать имена узлов вместо IP-адресов. Рекомендуется использовать отдельный DNS-сервер, но если это технически невозможно, то допустимо внести соответствующие записи в файл `/etc/hosts`.

Вначале, для примера, примем следующие данные узлов, которые будут использованы далее в этой статье:

- первый узел кластера:
  имя узла: control-1
  IP-адрес: 10.24.24.141
- второй узел кластера:
  имя узла: control-2
  IP-адрес: 10.24.24.53
- третий узел кластера:
  имя узла: control-3
  IP-адрес: 10.24.24.75

Также нужно создать имена для двух VIP-адресов: для PostgreSQL и сервисов Virtual Security. Для примера примем следующее:

- VIP-адрес для PostgreSQL:
  имя для адреса: pgsql-vip
  IP-адрес: 10.24.24.102
- VIP-адрес для сервисов Virtual Security:
  имя для адреса: bvs-vip
  IP-адрес: 10.24.24.200

Добавьте записи в файл `/etc/hosts:`

```
cat<<EOF>> /etc/hosts

10.24.24.141  control-1
10.24.24.53  control-2
10.24.24.75    control-3
10.24.24.102    pgsql-vip
10.24.24.200   bvs-vip
EOF
```

Примечание

Для корректного распознавания указанных в `/etc/hosts` имён узлов убедитесь, что они вначале ищутся в файлах hosts и только потом - DNS. Для этого откройте файл `/etc/nsswitch.conf` и проверьте поиск имён для hosts:

```
hosts:      files dns myhostname
```

files должно находиться на первом месте.

Установка и настройка Pacemaker

Для установки Pacemaker необходимо выполнить ряд операций.

Установка пакетов Pacemaker

Установите пакеты Pacemaker:

```
apt-get install pacemaker pcs
```

Запустите сервис pcsd:

```
systemctl start pcsd.service; \
systemctl enable pcsd.service
```

Настройка аутентификации и запуск компонентов кластера

Для объединения узлов в кластер Pacemaker вначале необходимо настроить аутентификацию кластера.

Укажите пароль для пользователя hacluster. Пароль для пользователя hacluster для всех узлов должен совпадать:

```
passwd hacluster
```

Далее команды pcs нужно выполнять только на одном узле, изменения будут засинхронизированы автоматически.

Произведите аутентификацию узлов кластера. При выполнении команды нужно указать пользователя и пароль, который был настроен в прошлом шаге:

```
pcs host auth -u hacluster control-1 control-2 control-3
```

Создайте кластер:

```
pcs cluster setup cluster control-1 control-2 control-3
```

Примечание

Сообщения `'No addresses specified for host...'` не являются ошибками, их можно игнорировать.

Запустите кластер:

```
pcs cluster start --all
```

Проверьте доступность кластера:

```
pcs status
```

Перезапуск узлов в кластере

Для сервисов Pacemaker рекомендуется не включать автозапуск сервисов Pacemaker. Поэтому при штатной перезагрузке узла кластера требуется выполнить ряд несложных команд. Также отметим, что узлы нужно перезапускать только по одному.

Остановите узел кластера, который вы хотите перезапустить:

```
pcs cluster stop control-1
```

Дождитесь переезда всех ресурсов с этого узла на другие через команду:

```
pcs status
```

После этого штатно перезапустите узел. Когда узел будет доступен по сети, снова запустите сервисы Pacemaker в нём:

```
pcs cluster start control-1
```

Включение функций фенсинга узлов

Включение STONITH и определение политики при отсутствии кворума

Для вывода из эксплуатации нестабильных или вышедших из строя узлов в Pacemaker используется технология STONITH. Суть технологии в том, чтобы по независимому каналу, в частности, по протоколу IPMI[4] для BMC[5], в случае возникновения проблемы с узлом можно было перезапустить или выключить узел. Для корректной работы STONITH требуется как минимум три узла и активация политики отсутствия кворума. Сам процесс вывода с последующей миграцией ресурсов называется фенсингом (fencing).

Примечание

В случае развёртывания кластера в тестовой или иной не продуктивной среде, при отсутствии возможности использования фенсинга, его необходимо отключить.

```
pcs property set no-quorum-policy=ignore
pcs property set stonith-enabled=false
```

Дальнейшую настройку фенсинга не проводить.

Для включения кворума и STONITH укажите параметры сервиса:

```
pcs property set stonith-enabled=true
```

Кворум в Pacemaker проверяет, сколько узлов функционирует в кластере. При наличии строго больше 50% работающих узлов Pacemaker считает, что кластер находится в работоспособном состоянии (то есть имеется кворум). Однако если это правило не выполняется, Pacemaker применяет одну из политик отсутствия кворума: остановка всех узлов (stop), freeze (запрет на перемещение ресурсов без

4 https://ru.wikipedia.org/wiki/Intelligent_Platform_Management_Interface
5 https://www.servethehome.com/explaining-the-baseboard-management-controller-or-bmc-in-servers/

остановки ресурсов), suicide (выключение всех узлов) и ignore (продолжение работы всех ресурсов). Для сервисов Virtual Security предлагается использовать политику stop:

```
pcs property set no-quorum-policy=stop
```

В таком варианте безопасно временно терять один узел из трёх, при потере двух узлов Pacemaker выключит все ресурсы.

Дополнительно рекомендуем, чтобы STONITH по умолчанию выключал вышедшие из строя узлы кластера (по умолчанию STONITH их пытается перезапустить):

```
pcs property set stonith-action=poweroff
```

Добавление данных IPMI для узлов в кластере

В качестве стандартного метода фенсинга рекомендуется использовать протокол IPMI.

На всех узлах, где запущены компоненты Pacemaker, установите пакет фенсинга для IPMI:

```
apt-get install fence-agents
```

Для включения ресурсов для фенсинга через IPMI необходимо использовать команду pcs stonith. Для каждого узла нужен свой ресурс stonith. Предлагаемые настройки для ресурса:

```
pcs stonith create ipmi_control-1 fence_ipmilan pcmk_host_list="control-1" ip=10.16.10.3
 lanplus=true username=ADMIN password=Rp56Yhre auth=MD5
```

Нужно обратить внимание на следующее:

- Так как для каждого узла требуется создать ресурс stonith, после create в имени ресурса укажите, к какому узлу относится ресурс и его укажите в параметре pcmk_host_list;
- В ip укажите адрес IPMI-сервера. Если используется нестандартный порт, то вместо ip укажите параметр ipport и укажите порт через двоеточие после адреса (напр. 10.23.10.3:6623);
- По умолчанию предлагается использование IPMI 2.0 через параметр lanplus;
- Остальные параметры определяют данные аутентификации со стандартным методом хэширования пароля MD5.

По умолчанию возможна ситуация, когда ресурс stonith может запуститься на том же узле, BMC которого контролируется этим ресурсом. Для запрета такой ситуации требуется добавить соответствующее ограничение для каждого из ресурсов stonith:

```
pcs constraint location ipmi_control-1 avoids control-1=INFINITY
```

Создание VIP-ресурсов для сервисов

Для единой точки входа в СУБД необходимо настроить ресурс Pacemaker с VIP-адресом. Выделенный узел для VIP определит начальный master-узел PostgreSQL далее.

Добавьте VIP-ресурс в Pacemaker:

```
pcs resource create pgsql-vip ocf:heartbeat:IPaddr2 ip=10.24.24.102 cidr_netmask=32 op
monitor interval=10s
```

Создайте VIP-ресурс для доступа на веб-интерфейс Virtual Security:

```
pcs resource create bvs-pri-ip ocf:heartbeat:IPaddr2 ip=10.24.24.200 cidr_netmask=32 op
monitor interval=10s --group bvs
```

Получите статус ресурсов и определите, на каком узле запустился VIP-адрес:

```
pcs status
...
 * pgsql-vip    (ocf::heartbeat:IPaddr2):    Started control-1
  * Resource Group: bvs:
    * bvs-pri-ip    (ocf::heartbeat:IPaddr2):    Started control-1
```

Если vip ресурсы запустились на разных узлах, переместите их на один следующей командой:

```
pcs resource move pgsql-vip control-1
```

Установка PostgreSQL

Установка пакетов

В качестве СУБД для хранения данных Virtual Security использует PostgreSQL. На текущий момент поддерживается версия 11.

Установите пакет, необходимый для корректной установки Virtual Security:

```
apt-get install dialog
```

Установите пакеты, необходимые для работы СУБД:

```
apt-get install postgresql postgresql-11 astra-resource-agents
```

Создание пользователя для работы с СУБД

Для корректной работы Virtual Security с СУБД, при включенном мандатном доступе (МД) необходимо следующее:

Создайте системного пользователя с тем же именем что и пользователь базы данных:

```
useradd vs
```

Назначьте созданному пользователю мандатный уровень:

```
pdpl-user -l 0:0 vs
```

Добавьте созданного пользователя в базу данных мандатного доступа и разрешите пользователю postgres читать из базы МД этого пользователя:

```
setfacl -m u:postgres:r /etc/parsec/macdb/$(id -u vs)
```

Настройка кластера PostgreSQL

Для настройки кластера PostgreSQL необходимо выполнить ряд операций.

Настройка master-узла

Важно

Следующие операции выполняются на master-узле кластера. Master-узел определяется по тому, в каком узле поднялся VIP-ресурс pgsql-vip в Pacemaker. Далее считается, что master-узлом является control-1.

Примечание

Статья настройки в wiki Astra, для работы с МРД (мандатное управление доступом).[6]

Зайдите под пользователем postgres:

```
su - postgres
```

Добавьте, по необходимости рабочий shell учетной записи postgres:

```
usermod postgres -s /bin/bash
```

Укажите имя узла и имя VIP-адреса PostgreSQL, в примере pgsql-vip, в `/etc/postgresql/11/listen.conf`:

6 https://wiki.astralinux.ru/pages/viewpage.action?pageId=23875043 4

```
cat <<EOF > /etc/postgresql/11/listen.conf
listen_addresses = 'pgsql-vip, control-1'
EOF
```

Добавьте параметры в `postgresql.conf`:

```
cat <<EOF>> /etc/postgresql/11/main/postgresql.conf
hba_file = '/etc/postgresql/11/main/pg_hba.conf'
wal_keep_segments = 32
wal_level = replica
include = '../listen.conf'
EOF
```

Удалите конфигурацию `pg_hba.conf`:

```
rm /etc/postgresql/11/main/pg_hba.conf
```

Сформируйте конфигурацию `pg_hba.conf`, вписав точное имя хоста и имя vip адреса pgsql, в примере pgsql-vip:

```
cat <<EOF > /etc/postgresql/11/main/pg_hba.conf
local all         all                     trust
host  all         all      127.0.0.0/8    trust
host  all         all      ::1/128        trust
# forbid self-replication from VIP
host  replication postgres pgsql-vip      reject
# forbid self-replication its own IP
local replication all                     reject
host  replication all      control-1      reject
host  replication all      127.0.0.0/8    reject
host  replication all      ::1/128        reject
# allow any standby connection
host  replication postgres 10.24.24.0/24  trust
host  vs       vs    10.24.24.0/24       md5
EOF
```

Здесь нужно:

- В строке "host replication postgres pgsql-vip reject" указать VIP-адрес или имя домена VIP-адреса;
- В строке "host replication all control-1 reject" указать фактический адрес узла или имя домена для этого адреса;
- В строке "host vs vs 10.24.24.0/24 md5" нужно указать подсеть, через которую клиенты (в частности, сервисы Virtual Security) могут подключиться к СУБД. Возможно указание адресов сервисов по строке явно через указание подсети /32.

Осуществите выход из под пользователя postgres:

```
exit
```

Запустите сервис PostgreSQL:

```
systemctl start postgresql
```

Проверьте статус PostgreSQL:

```
systemctl list-units |grep postgre
postgresql.service                                                    loaded active
exited    PostgreSQL RDBMS
postgresql@11-main.service                                            loaded active
running   PostgreSQL Cluster 11-main
system-postgresql.slice                                               loaded active
active    system-postgresql.slice
```

Настройка slave-узлов

> **Важно**
>
> Следующие операции выполняем на slave-узлах.

Зайдите под пользователем postgres:

```
su - postgres
```

Добавьте, по необходимости рабочий shell учетной записи postgres:

```
usermod postgres -s /bin/bash
```

Удалите содержимое кластера СУБД:

```
rm -rf /var/lib/postgresql/*/main/*
```

Укажите имя узла и имя VIP-адреса PostgreSQL, в примере pgsql-vip, в `/etc/postgresql/11/listen.conf`:

```
cat <<EOF > /etc/postgresql/11/listen.conf
listen_addresses = 'pgsql-vip, control-2'
EOF
```

Сформируйте конфигурацию `pg_hba.conf`, вписав точное имя узла и имя VIP-адреса pgsql, в примере pgsql-vip:

```
cat <<EOF > /etc/postgresql/11/main/pg_hba.conf
local all         all                     trust
host  all         all      127.0.0.0/8    trust
host  all         all      ::1/128        trust
# forbid self-replication from VIP
host  replication postgres pgsql-vip      reject
# forbid self-replication its own IP
local replication all                     reject
host  replication all      control-2      reject
host  replication all      127.0.0.0/8    reject
host  replication all      ::1/128        reject
# allow any standby connection
host  replication postgres 10.24.24.0/24  trust
host  vs       vs    10.24.24.0/24       md5
EOF
```

Здесь нужно:

- В строке "host replication postgres pgsql-vip reject" указать VIP-адрес или имя домена VIP-адреса;
- В строке "host replication all control-2 reject" указать фактический адрес узла или имя домена для этого адреса;
- В строке "host vs vs 10.24.24.0/24 md5" нужно указать подсеть, через которую клиенты (в частности, сервисы Virtual Security) могут подключиться к СУБД. Возможно указание адресов сервисов по строке явно через указание подсети /32.

Создайте репликацию из главного (master) узла в текущий (slave) узел:

```
pg_basebackup -h pgsql-vip -D /var/lib/postgresql/11/main/ -P
```

Выйдите из пользователя postgres:

```
exit
```

Повторите операции на следующем slave-узле.

Настройка кластера Pacemaker

Теперь необходимо настроить автоматическое переключение с помощью Pacemaker.
Отключите PostgreSQL. Сначала на slave-узлах, затем на master:

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

Создайте ресурс postgresql:

```
pcs resource create pgsql ocf:astra:pgsql \
pgversion="11" \
rep_mode="sync" \
master_ip="pgsql-vip" \
node_list="control-1 control-2 control-3"
```

Объявите ресурс pgsql "продвигаемым" (promotable) и создайте вторичные ресурсы. Такие ресурсы могут переключаться между двумя режимами работы - master и slave:

```
pcs resource promotable pgsql promoted-max=1 promoted-node-max=1 clone-max=3 clone-node-
max=1 notify=true
```

Настройте зависимости ресурсов:

```
pcs constraint colocation add pgsql-vip with master pgsql-clone INFINITY
pcs constraint order promote pgsql-clone then start pgsql-vip symmetrical=false
 kind=Mandatory
pcs constraint order demote pgsql-clone then stop pgsql-vip symmetrical=false
 kind=Mandatory
```

Проверяем статус кластера postgresql в pcs. На одном из узлов должна быть роль master:

```
pcs status

Online: [ control-1 control-2 control-3 ]

Full list of resources:

 pgsql-vip    (ocf::heartbeat:IPaddr2):    Started control-1
 Resource Group: bvs
     bvs-pri-ip    (ocf::heartbeat:IPaddr2):    Started control-1
 Clone Set: pgsql-clone [pgsql] (promotable)
     Masters: [ control-1 ]
     Slaves: [ control-2 control-3 ]
```

Создание БД для сервисов Virtual Security

Проверьте что в выводе команды `pcs status` узел control-1 имеет роль master. Настройки выполняются на этом узле.

Зайдите под пользователем postgres:

```
sudo -i -u postgres
```

Запустите терминальный клиент для работы с СУБД:

```
psql
```

Внутри сессии psql создайте пользователя vs с указанием пароля:

```
create user vs with password 'УКАЖИТЕПАРОЛЬ';
```

Создайте БД vs:

```
create database vs owner vs;
```

Выйдите из терминального клиента БД:

```
\q
```

Проверьте подключение пользователя vs к созданной БД:

```
psql -h pgsql-vip -U vs
```

При корректном подключении появится следующая строка:

```
vs=>
```

Создайте схемы vsm и ids

```
create schema vsm AUTHORIZATION vs;
create schema idp AUTHORIZATION vs;
```

Выйдите из терминального клиента БД:

```
\q
```

Осуществите выход из под пользователя postgres:

```
exit
```

Настройка сервисов Virtual Security

Установите пакеты Virtual Security на всех узлах:

```
apt install basis-openjdk basis-vbalancer virtual-security-auth virtual-security-admin
virtual-security-web
```

Запустите конфигурирование модулей Virtual Security на каждом узле, в качестве адреса БД укажите имя VIP-адреса PGSQL, в качестве сервера балансировки и URL системы укажите VIP-адрес для сервисов Virtual Security. Настройка производится на узле с активным ресурсом bvs-pri-ip. Для его вычисления запустите команду `pcs status`:

```
pcs status
...
Resource Group: bvs
     bvs-pri-ip    (ocf::heartbeat:IPaddr2):    Started control-1
...
```

В нашем случае это узел control-1.

Настройте сервисы Virtual Security.

Auth

Настройте конфигурационный файл - `/opt/bvs/auth/lib/config/idp.conf`:

- Укажите базовый URL для сервиса:

  ```
  hostname-url=http://bvs-vip:8080
  hostname-admin-url=http://bvs-vip:8080
  ```

- Укажите адрес и порт прослушивания:

  ```
  http-host=bvs-test
  http-port=8082
  ```

- Укажите адрес подключения к postgresql:

  ```
  db-url-host=bvs-test
  ```

Включите регистрацию сервиса в сервисе балансировки (vbalancer) путем включения параметра в `/opt/bvs/auth/lib/config/application.properties`:

```
quarkus.mod-cluster-client.enabled=true
```

Запустите сервис:

```
systemctl start virtual-security-auth
```

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-auth
```

Admin

Настройте конфигурационный файл - `/opt/bvs/admin/config/application.properties`:

```
client:
    base-uri: http://bvs-vip:8080
    realm: master
    client-id: security-admin-console
  security-proxy:
    base-uri: http://bvs-vip:8080/proxy
  provider:
    messaging:
      count: 2
      broker-uri: tcp://bvs-test:61616?
clientFailureCheckPeriod=5000&retryInterval=1000&reconnectAttempts=5&callTimeout=3000

...

http:
    host: bvs-test
    port: 8081
    root-path: /api
    cors:
      ~: true
      origins: http://bvs-vip:8080

...

datasource:
    db-kind: postgresql
    username: vs
    password: vs
    jdbc:
      url: jdbc:postgresql://bvs-test:5432/vs
      max-size: 10

...

 mod-cluster-client:
    enabled: true
```

Запустите сервис:

```
systemctl start virtual-security-admin
```

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-admin
```

Web

Настройте конфигурационный файл - `/opt/bvs/web/config/application.properties`:

```
quarkus.http.host=bvs-test
...
quarkus.oidc.auth-server-url=http://bvs-vip:8080/realms/master
...
quarkus.mod-cluster-client.enabled=true
...
```

```
quarkus.mod-cluster-client.services.api.host=bvs-test
quarkus.mod-cluster-client.services.api.port=8080
quarkus.mod-cluster-client.services.api.contexts=/webadmin
```

Запустите сервис:

```
systemctl start virtual-security-web
```

Добавьте сервис в автозагрузку:

```
systemctl enable virtual-security-web
```

Broker

Настройте конфигурационный файл - `/opt/vcore/broker/config/application.properties`:

```
quarkus.mod-cluster-client.enabled=true
quarkus.mod-cluster-client.services.api.host=bvs-test
provider.transport.use-multicast-discovery=true
provider.transport.multicast-interface-address=bvs-test
provider.transport.multicast-group=231.7.7.7
provider.transport.multicast-port=9876
provider.transport.static-discovery-addresses=tcp://bvs-test:61617
provider.transport.bind-address=bvs-test
```

Запустите сервис:

```
systemctl start vcore-broker
```

Добавьте сервис в автозагрузку:

```
systemctl enable vcore-broker
```

vbalancer

Настройте конфигурационный файл - /etc/sysconfig/basis-vbalancer:

```
BALANCER_BIND_ADDR="bvs-vip"
BALANCER_PRIVATE_BIND_ADDR="bvs-test"
```

Запустите сервис:

```
systemctl start basis-vbalancer
```

Добавьте сервис в автозагрузку:

```
systemctl enable basis-vbalancer
```

Остановите сервисы Virtual Security:

```
systemctl stop virtual-security-* basis-vbalancer
```

Перед настройкой сервисов Virtual Security на следующем узле, переместите ресурс bvs-pri-ip на настраиваемый узел, в примере control-2:

```
pcs resource move  bvs-pri-ip control-2
```

Проверьте корректность перемещения ресурса:

```
pcs status

...
Resource Group: bvs
     bvs-pri-ip    (ocf::heartbeat:IPaddr2):    Started control-2
```

```
...
```

Настройка кластера Pacemaker для Virtual Security

Отключите зависимости vs-systemd на всех узлах. Скомментируйте строку `#Wants=:`. Это необходимо для корректной работы кластера Pacemaker:

```
systemctl edit --full basis-vbalancer
systemctl edit --full virtual-security-auth
systemctl edit --full virtual-security-admin
systemctl edit --full virtual-security-web
```

Отключите автозапуск сервисов Virtual Security в systemd:

```
systemctl disable --now basis-vbalancer virtual-security-auth virtual-security-admin
virtual-security-web
```

Создайте ресурсы Virtual Security и определите зависимости и очерёдность их запуска:

```
pcs resource create basis-vbalancer systemd:basis-vbalancer --group bvs

pcs resource create virtual-security-admin systemd:virtual-security-admin clone
pcs resource create virtual-security-auth systemd:virtual-security-auth clone op start
interval=0s timeout=200s
pcs resource create virtual-security-web systemd:virtual-security-web clone

pcs constraint order virtual-security-auth-clone then virtual-security-admin-clone
pcs constraint order virtual-security-auth-clone then virtual-security-web-clone
pcs constraint order virtual-security-admin-clone then virtual-security-web-clone
pcs constraint order virtual-security-web-clone then basis-vbalancer

pcs constraint colocation add bvs-pri-ip with basis-vbalancer INFINITY
pcs constraint order pgsql-clone then virtual-security-auth-clone
```

Проверьте статус кластера, для корректного запуска ресурсов может потребоваться несколько минут:

```
pcs status
Cluster name: cluster
Stack: corosync
Current DC: control-2 (version 2.0.1-9e909a5bdd) - partition with quorum
Last updated: Wed Sep 13 12:27:20 2023
Last change: Wed Sep 13 12:21:16 2023 by root via cibadmin on control-1

3 nodes configured
15 resources configured

Online: [ control-1 control-2 control-3 ]

Full list of resources:

 pgsql-vip    (ocf::heartbeat:IPaddr2):    Started control-1
 Resource Group: bvs
     bvs-pri-ip    (ocf::heartbeat:IPaddr2):    Started control-3
     tvs-balancer    (systemd:tvs-balancer):    Started control-3
 Clone Set: pgsql-clone [pgsql] (promotable)
     Masters: [ control-1 ]
     Slaves: [ control-2 control-3 ]
 Clone Set: tvs-core-clone [tvs-core]
     Started: [ control-1 control-2 control-3 ]
 Clone Set: tvs-security-clone [tvs-security]
     Started: [ control-1 control-2 control-3 ]
 Clone Set: tvs-web-clone [tvs-web]
     Started: [ control-1 control-2 control-3 ]
```

Если статус ресурсов не соответствует "Started" и отображаются ошибки запуска ресурсов, воспользуйтесь командой `pcs resource cleanup` и поверьте работоспособность ресурсов, через несколько минут.

Вычислительная часть

Примечание

Все процессы описанные ниже производятся на вычислительных узлах кластера.

Важно

При установке вычислительной части Virtual Security на ОС Astra Linux Special Edition 1.7.4 (Smolensk) не удовлетворяются зависимости по некоторым пакетам. Необходимые версии пакетов есть в репозитории "Расширенный", но при его использовании система переходит в "состояние совместимости с Astra Linux Common Edition". Расширенный репозиторий содержит необходимые версии пакетов, не включенных в состав основного и базового репозиториев. Особенности перехода в режим совместимости:

- **не проходит сертификационные испытания в составе Astra Linux Special Edition;**
- функционирует в среде Astra Linux;
- не подвергается доработке для интеграции функций безопасности с комплексом средств защиты (КСЗ) Astra Linux Special Edition и для устранения несоответствий сертификационным требованиям;
- может быть несовместимо с пакетами, имеющимися в базовом и расширенном репозитории.

Процесс установки и настройки вычислительной части не отличается от простой инсталляции. Данная процедура подробно описана в разделе - "Настройка агента вычислительного узла (см. стр. 12).

Обновление программного модуля Базис.Virtual Security

Важно

Обновление Базис.Virtual Security v.3.0 с более ранних версий не поддерживается.

Данная инструкция описывает процесс обновления Базис.Virtual Security v.3.0 в рамках минорных версий продукта:

- Предварительные действия перед обновлением (см. стр. 46)
- Управляющая часть (см. стр. 46)
  - Обновление пакетов (см. стр. 46)
  - Обновление конфигурации сервисов (см. стр. 46)
  - Запуск сервисов (см. стр. 47)
- Вычислительная часть (см. стр. 47)
  - Обновление конфигурации сервисов (см. стр. 47)
  - Запуск сервисов (см. стр. 47)

Предварительные действия перед обновлением

На период обновления платформы необходимо остановить все работы в облаке:

1. Остановите все запущенные ВМ.
2. Откючите все вычислительные узлы.
3. Создайте резервную копию БД.

Примечание

Команды приведенные ниже указаны для ОС AlmaLinux. Данная инструкция применима для всех поддерживаемых ОС.

Управляющая часть

Обновление пакетов

Остановите службы системы управления:

```
systemctl stop virtual-security-auth virtual-security-admin virtual-security-proxy
virtual-security-web basis-vbalancer
```

Проверьте наличие обновлений для пакетов Virtual Security:

```
dnf check-update --refresh | grep 'virtual-security-'
dnf check-update --refresh | grep 'basis-'
```

Обновите пакеты Базис Базис.Virtual Security:

```
dnf -y update virtual-security-auth virtual-security-admin virtual-security-proxy
virtual-security-web basis-vbalancer
```

Для обновления всей системы используйте обычную команду update:

```
dnf -y update
```

Обновление конфигурации сервисов

После обновления пакетов имеющийся файл конфигурации сервисов будет переименован как `applications.properties.rpmsave`. В новом файле `application.properties` необходимо перенести все указанные в старой конфигурации параметры сервисов. После чего сохраните изменения.

Запуск сервисов

Запустите службы:

```
systemctl start virtual-security-auth virtual-security-admin virtual-security-proxy
virtual-security-web basis-vbalancer
```

Включите автозагрузку служб:

```
systemctl enable virtual-security-auth virtual-security-admin virtual-security-proxy
virtual-security-web basis-vbalancer
```

Вычислительная часть

Остановите агент Virtual Security:

```
systemctl stop virtual-security-agent
```

Проверьте наличие обновлений:

```
dnf check-update --refresh | grep 'virtual-security-'
```

Обновите пакет:

```
dnf -y update virtual-security-agent
```

Для обновления всей системы используйте обычную команду update:

```
dnf -y update
```

Обновление конфигурации сервисов

После обновления пакета имеющийся файл конфигурации сервиса будет переименован как `applications.properties.rpmsave`. В новом файле `application.properties` необходимо перенести все указанные в старой конфигурации параметры сервиса. После чего сохраните изменения.

Запуск сервисов

Запустите службы:

```
systemctl start virtual-security-agent
```

Включите автозагрузку служб:

```
systemctl enable virtual-security-agent
```

Приложения к руководству по установке программного модуля Базис.Virtual Security

Приложение 1. Настройка модуля PostgreSQL PAM для программного модуля Базис.Virtual Security

- Введение (см. стр. 48)
- Установка модуля (см. стр. 48)
- Настройка модуля (см. стр. 49)
- Настройка модуля для иных OpenID-провайдеров (см. стр. 50)
- Проверка установки и настройки (см. стр. 52)

Введение

Данная статья посвящена установке и настройке модуля аутентификации доступа к базе данных PostgreSQL с использованием токенов JWT, генерируемых Basis Virtual Security и другими OpenID-провайдерами.

Поддерживаются версии PostgreSQL 10 по 12 включительно. Рекомендуется использование версии PostgreSQL версии 12.14.

Возможности модуля

PG PAM позволяет интегрироваться в нескольких вариантах инсталляции PostgreSQL:

- PG PAM по умолчанию предназначен для работы с обычной инсталляцией PostgreSQL;
- PG PAM, начиная с версии 2.4.0, позволяет работать в Standby-варианте запуска PostgreSQL, запущенный в кластерном режиме. В этом случае PG PAM может быть настроен на всех инстансах PostgreSQL, распределяя нагрузку по доступу к базам данных в режиме для чтения;
- PG PAM, начиная с версии 2.4.0, позволяет менять контекст пользователя как на уровне SESSION, так и на уровне LOCAL, то есть внутри транзакционного блока.

Новые возможности PG PAM не требуют отдельной настройки со стороны модуля и PG PAM. Прямое обновление PG PAM поддерживается с версии 2.2.0.

Ограничения модуля

PG PAM имеет некоторые ограничения:

- поддерживает только авторизацию пользователя при доступе к базе в режиме SUPERUSER. Другие роли не поддерживаются;
- при переключении пользователя на уровне SESSION все соединения сессии будут переключены в режим SUPERUSER. Это сломает работу пулов соединений. Поэтому при использовании pgpool или пула соединения для JDBC необходимо переключение пользователя для каждой отдельной транзакции;
- работа PG PAM в PostgreSQL в режиме Active/Active не проверена и на текущий момент официально не поддерживается.

Установка модуля

Официально пакеты для установки доступны для Alt Linux p10:

- Пакеты для обычной (открытой) версии PostgreSQL. Поддерживаются версии 11 и 12;
- Пакеты для PostgreSQL Pro. Поддерживаются версии с 10 по 12 включительно.

Ссылки на RPM-пакеты (репозитории пока не готовы):

- PostgreSQL Community (открытая):
  - Версия 11[7]
  - Версия 12[8]
- PostgreSQL Pro для сертифицированных Alt Linux:
  - Версия 10[9]

---

7 https://bvs-public:bvs-public@maven.tionix.ru:443/artifactory/tvs-rpm/dev/altlinux/x86_64/tionix-tvs-pg-pam-comm-11-altp10-2.4.0-2307171510.alt.x86_64.rpm

8 https://vcore-public:vcore-public@maven.tionix.ru:443/artifactory/tvs-rpm/dev/altlinux/x86_64/tionix-tvs-pg-pam-comm-12-altp10-2.4.0-2307171510.alt.x86_64.rpm

9 https://bvs-public:bvs-public@maven.tionix.ru:443/artifactory/tvs-rpm/dev/altlinux/x86_64/tionix-tvs-pg-pam-std-pro-10-2.4.0-2307171510.alt.x86_64.rpm

- Версия 11[10]
- Версия 12[11]

Загрузите пакет (для примера - PostgreSQL Comm версии 12):

```
apt-get -y install wget
wget https://vcore-public:vcore-public@maven.tionix.ru/artifactory/tvs-rpm/dev/altlinux/x86_64/tionix-tvs-pg-pam-comm-12-altp10-2.3.0-2304261313.alt.x86_64.rpm
```

Установите PostgreSQL:

```
apt-get -y install postgresql12-server
```

Установите пакет модуля через rpm:

```
rpm -ihv tionix-tvs-pg-pam-comm-12-altp10-2.3.0-2304261313.alt.x86_64.rpm
```

Настройка модуля

Запуск PostgreSQL

Перед началом настройки модуля настройте сервис PostgreSQL. Для этого обратитесь к документации Alt Linux[12], раздел "Установка и начальный запуск").

Настройка для Basis Virtual Security

По умолчанию пользователь root не включен в списки пользователей в sudoers и в этом случае при попытке запустить sudo от этого пользователя вы получите ошибку:

```
root is not in the sudoers file.
```

Для исправления этой проблемы необходимо включить root в список sudoers. Для этого найдите строку в `/etc/sudoers` и откомментируйте следующую строку:

```
# root ALL=(ALL:ALL) ALL
```

Сохраните изменение.

Вначале зарегистрируйте файл расширения для PostgreSQL:

```
sudo -u postgres psql -f "/usr/share/pgsql/extension/tionix_pam--1.0.sql"
sudo -u postgres psql -c "CREATE EXTENSION tionix_pam;"
```

Откройте файл проверки работы алгоритма верификации по пути `/usr/lib/pgsql/extension/tionix_pam_demo-1.0.sql` и найдите следующие строки:

```
    req_claims := ARRAY['("iss","https://tvs.tionix.ru/realms/posgres_pam")'::CLAIM,
                        '("sub","c6002500-2ca6-453a-9e4b-c4f1174031df")'::CLAIM,
                        '("typ","ID")'::CLAIM
                       ];
```

Здесь нужно включить следующие изменения:

- в поле *iss* нужно включить адрес BVS, вместо *posgres_pam* укажите имя клиентской системы;
- в поле *sub* нужно указать идентификатор клиентской платформы.

Сохраните изменения и примените этот файл в окружении СУБД:

```
sudo -u postgres psql -f "/usr/share/pgsql/extension/tionix_pam_demo--1.0.sql"
```

10 https://bvs-public:bvs-public@maven.tionix.ru:443/artifactory/tvs-rpm/dev/altlinux/x86_64/tionix-tvs-pg-pam-std-pro-11-2.4.0-2307171510.alt.x86_64.rpm
11 https://bvs-public:bvs-public@maven.tionix.ru:443/artifactory/tvs-rpm/dev/altlinux/x86_64/tionix-tvs-pg-pam-std-pro-12-2.4.0-2307171510.alt.x86_64.rpm
12 https://www.altlinux.org/PostgreSQL

Откройте файл `/var/lib/pgsql/postgresql.conf` и перейдите в конец файла, после чего добавьте следующие строки:

```
tionix_pam.well_known_url = https://tvs.tionix.ru/realms/posgres_pam/.well-known/openid-configuration
tionix_pam.client_id = 'posgres_pam'
tionix_pam.enable_insecure = 0
```

Здесь нужно включить следующие изменения:

- в поле *tionix_pam.well_known_url* нужно указать полный URL до конфигурации OpenID. Вместо *posgres_pam* укажите имя клиентской системы;
- в поле *tionix_pam.client_id* укажите имя клиентской системы. Он должен совпадать с тем, что указано в URL;
- поле *tionix_pam.enable_insecure* регулирует возможность работы с провайдером, который содержит невалидный сертификат для HTTPS. Укажите 1 для включения пропуска невалидных сертификатов.

Сохраните эти изменения.

В окружении PostgreSQL добавьте пользователя, имя которого совпадает с именем сервисного пользователя клиентской системы в BVS (вместо *posgres_pam* укажите имя клиентской системы:

```
sudo -u postgres psql -c "CREATE USER \"service-account-posgres_pam\";"
```

Перезапустите PostgreSQL:

```
systemctl restart postgresql
```

Настройка модуля для иных OpenID-провайдеров

На текущий момент скрипт гарантирует работу только с BVS "из коробки", однако имеется возможность подключения иных OpenID-провайдеров, если токены JWT содержат необходимые данные:

- Для работы требуется наличие поля x5c с публичным сертификатом для валидации токенов.
- Заголовок токен должен содержать поля kid, alg и typ.
- Поля aud и azp должны быть равны полю clientId в токене JWT. Эта информация используется для определения имени клиентской системы.
- Поле iss проверяется на корректность URL.
- Поле sub используется для определения идентификатора клиентской системы.
- preferred_username используется как имя роли для СУБД PostgreSQL.

Для примера рассмотрим следующий токен JWT, который сгенерирован другим OpenID-провайдером (показаны заголовок и основное тело JWT, декодированный в текст):

```
header:
{
  "kid": "4u90eiuhfw8ayfhr98ea4yf09et94ty8h34",
  "typ": "JWT",
  "alg": "RS256"
}

body:
{
  "sub": "4e8280b1-ae41-45bb-a7dd-53f310b97330",
  "std": "68T001",
  "clientId": "arm",
  "roles": [],
  "iss": "https://example.com",
  "groups": [],
  "preferred_username": "3759aa71-9bc2-4e09-8497-fed46b702693",
  "given_name": "Имя",
  "middle_name": "Отчество",
  "domain_id": "6b57fbf7-7910-4d07-b6e7-47be9b9af5de",
  "aud": "arm",
  "acr": "1",
```

```
  "accountId": "4e8280b1-ae41-45bb-a7dd-53f310b97330",
  "nbf": 0,
  "azp": "arm",
  "scope": "profile openid S3Client",
  "name": "Фамилия Имя Отчество",
  "realm": "test_dep",
  "exp": 1681993677,
  "session_state": "a0238cca-499d-439e-9c8f-46ec9f532b26",
  "iat": 1681990077,
  "family_name": "Фамилия",
  "jti": "06dd3433-fe87-458a-8f36-24cfdf017ed5"
}
```

Из этого токена модуль получит следующие данные:

- для имени пользователя внутри PostgreSQL будет использоваться строка *3759aa71-9bc2-4e09-8497-fed46b702693;*
- в качестве имени клиентской системы в настройках модуля будет использована строка "*arm*", который содержится в поле clientId.

Шаги по настройке в целом совпадают с настройкой для BVS, но имеются некоторые различия.

Вначале зарегистрируйте файл расширения для PostgreSQL:

```
sudo -u postgres psql -f "/usr/share/pgsql/extension/tionix_pam--1.0.sql"
sudo -u postgres psql -c "CREATE EXTENSION tionix_pam;"
```

Откройте файл проверки работы алгоритма верификации по пути `/usr/lib/pgsql/extension/tionix_pam_demo-1.0.sql` и найдите следующие строки:

```
    req_claims := ARRAY['("iss","https://example.com")'::CLAIM,
                        '("sub","c6002500-2ca6-453a-9e4b-c4f1174031df")'::CLAIM,
                        '("typ","ID")'::CLAIM
                       ];
```

Здесь нужно включить следующие изменения:

- в поле *iss* нужно включить адрес OpenID-провайдера до части URL */.well-known/openid-configuration;*
- в поле *sub* нужно указать идентификатор клиентской платформы (из поля *sub* в токене);
- в *typ* нужно указать содержимое поля *typ* в заголовке токена (в примере он равен "*JWT*")

Сохраните изменения и примените этот файл в окружении СУБД:

```
sudo -u postgres psql -f "/usr/share/pgsql/extension/tionix_pam_demo--1.0.sql"
```

Откройте файл `/var/lib/pgsql/postgresql.conf` и перейдите в конец файла, после чего добавьте следующие строки:

```
tionix_pam.well_known_url = https://example.com/.well-known/openid-configuration
tionix_pam.client_id = 'posgres_pam'
tionix_pam.enable_insecure = 0
```

Здесь нужно включить следующие изменения:

- в поле *tionix_pam.well_known_url* нужно указать полный URL до конфигурации OpenID;
- в поле *tionix_pam.client_id* укажите содержимое поля clientID в токене JWT (в примере "*arm*");
- поле *tionix_pam.enable_insecure* регулирует возможность работы с провайдером, который содержит невалидный сертификат для HTTPS. Укажите 1 для включения пропуска невалидных сертификатов.

Сохраните эти изменения.

В окружении PostgreSQL добавьте пользователя с именем, который равен строке в поле *preferred_username* в токене JWT.

```
sudo -u postgres psql -c "CREATE USER \"3759aa71-9bc2-4e09-8497-fed46b702693\";"
```

Перезапустите PostgreSQL:

```
systemctl restart postgresql
```

Проверка установки и настройки

Для проверки работы алгоритма требуется сгенерированный токен JWT с корректной подписью. Для примера приведем такую проверку для BVS.

Вначале получите токен из BVS:

```
export BVS_DOMAIN="https://tvs.tionix.ru"
export REALM="posgres_pam"
export CLIENT_ID_PASSWORD="3f3b1bc7-447e-4736-ae3b-5372c2c937f1"
export TOKEN_ID=$(curl --insecure --silent -XPOST ${BVS_DOMAIN}/realms/${REALM}/
protocol/openid-connect/token -H 'content-type: application/x-www-form-urlencoded' -d
'client_id=${REALM}&client_secret={CLIENT_ID_PASSWORD}
&grant_type=client_credentials&scope=openid' | grep -oP '(?<="id_token":")[^"]*')
```

В экспортах нужно указать:

- BVS_DOMAIN - домен сервиса BVS;
- REALM - имя клиентской системы;
- CLIENT_ID_PASSWORD - пароль для клиентской системы.

После выполнения в переменную окружения TOKEN_ID попадет содержимое токена JWT для аутентификации.

Теперь выполните запуск функции pam_demo(), которая предназначена для вызова процесса валидации токена:

```
sudo -i -u postgres psql -c "SELECT * FROM pam_demo('${TOKEN_ID}');"
```

Приложение 2. Интеграция программного модуля Базис.Virtual Security с Базис.vCore

- Введение (см. стр. 52)
- Управляющая часть (см. стр. 52)
- Вычислительная часть (см. стр. 53)

Введение

Данная статья описывает процесс настройки Базис.Virtual Security для корректной работы с платформой Базис.vCore в рамках единого кластера управления.

Для установки vCore используйте один из возможных вариантов:

- Демонстрационная инсталляция.
- Простая инсталляция.
- Инсталляция с поддержкой высокой доступности.

Подробный процесс установки Базис.Virtual Security описан в "Руководстве по установке" данного продукта.

Управляющая часть

Настройте конфигурационный файл сервиса Control - `/opt/vcore/control/config/application.properties:`

```
quarkus.http.port=8086
broker-rest-api/mp-rest/url=http://BROKER_IP_DNS_ADDR:8085/api
```

Настройте конфигурационный файл сервиса Broker - `/opt/vcore/broker/config/application.properties`:

```
quarkus.http.port=8085
```

```
quarkus.mod-cluster-client.services.api.port=8085
```

Настройте firewall:

```
firewall-cmd --permanent --add-port=8085/tcp
firewall-cmd --permanent --add-port=8086/tcp
firewall-cmd --reload
```

Перезапустите сервисы Control и Broker:

```
systemctl restart vcore-broker vcore-control
```

После перезапуска сервисов веб-интерфейс платформы vCore будет доступен по адресу - `http://IP-ADDRESS:8086`.

Добавьте репозиторий проекта в - `/etc/yum.repos.d/bvs.repo`:

```
[bvs]
name=BVS $releasever
baseurl=https://tvs-public:tvs-public@maven.tionix.ru/artifactory/tvs-rpm/dev/fedora35/
$basearch
enabled=1
gpgcheck=0

[bvs-noarch]
name=BVS $releasever
baseurl=https://tvs-public:tvs-public@maven.tionix.ru/artifactory/tvs-rpm/dev/fedora35/
noarch
enabled=1
gpgcheck=0

[vcore-contrib]
name=vCore Contrib Packages
baseurl=https://vcore-public:vcore-public@maven.tionix.ru/artifactory/vcore-contrib/
fedora/linux/$releasever/$basearch
enabled=1
gpgcheck=0
```

Обновите данные репозиториев:

```
dnf update
```

Установите пакет postgresql:

```
dnf -y install postgresql12-server
```

Установите пакет tionix-tvs-system:

```
dnf install tionix-tvs-system
```

Запустите скрипт настройки:

```
tvs_configure.sh
```

Укажите необходимые параметры. По окончании процесса настройки все сервисы будут запущены автоматически.

Вычислительная часть

В данном разделе описана процедура установки агентского программного модуля Базис.Virtual Security на вычислительном узле (гипервизоре) Базис.vCore.

Добавьте репозиторий проекта в - `/etc/yum.repos.d/bvs.repo`:

```
[bvs]
```

```
name=BVS $releasever
baseurl=https://tvs-public:tvs-public@maven.tionix.ru/artifactory/tvs-rpm/dev/
fedora35/$basearch
enabled=1
gpgcheck=0

[bvs-noarch]
name=BVS $releasever
baseurl=https://tvs-public:tvs-public@maven.tionix.ru/artifactory/tvs-rpm/dev/fedora35/
noarch
enabled=1
gpgcheck=0
```

Установите необходимые пакеты:

```
dnf install -y tionix-tvs-agent
```

Запустите скрипт настройки

```
tvs_configure.sh
```

По окончании процесса настройки все сервисы будут запущены автоматически.

Перезапустите службу агента вычислительного узла:

```
systemctl restart vcore-agent
```

Приложение 3. Интеграция программного модуля Базис.Virtual Security с кластерной инсталляцией программного модуля Базис.vCore

- Подготовка кластера vCore (см. стр. 54)
  - Выключение объектов в vCore (см. стр. 54)
  - Выключение всех сервисов vCore (см. стр. 54)
  - Создание копии конфигурации (см. стр. 55)
- Настройка Pacemaker (см. стр. 55)
- Настройка портов (см. стр. 56)
- Установка Virtual Security (см. стр. 57)
- Настройка кластера PostgreSQL (см. стр. 60)
- Настройка кластера Pacemaker для PostgreSQL (см. стр. 63)

> **Примечание**
>
> Документация актуальна для кластера vCore на базе узлов с ОС AlmaLinux 8.7.

Подготовка кластера vCore

Выключение объектов в vCore

Перед началом процесса обновления необходимо выключить все активные ВМ во всех кластерах vCore.

Выключение всех сервисов vCore

Узлы управления

При обновлении управляющей части необходимо строго соблюдать порядок работ на узлах.

Остановите службы системы управления во всех узлах с учётом статуса миграции (статус миграции необходимо проверять в параметре provider.migration.enabled, который указан в файле `/opt/vcore/broker/config/application.properties`):

- вначале control последовательно на всех узлах:

```
systemctl stop vcore-control
```

- затем broker последовательно на всех узлах. В первую очередь необходимо выключить брокер с включенной миграцией (true), а потом - брокеры с выключённой (false), если таковые имеются:

```
systemctl stop vcore-broker
```

Также необходимо остановить и удалить ресурс `vcore_balancer` в Pacemaker (ресурс VIP можно не трогать):

```
pcs resource disable vcore_balancer
pcs resource delete vcore_balancer
```

Агент ВУ

> **Примечание**
>
> Здесь и далее предполагается, что ОС агента ВУ не будет переустановлена полностью. В случае переустановки ОС на ВУ нужно провести стандартную настройку агента, описанный в документации для 1.2 или 1.3.

Выключите агент ВУ и сервис конфигуратора для tty:

```
systemctl stop vcore-agent vcore-hyper-configurator
```

Создание копии конфигурации

Узлы управления

На всех узлах управления необходимо сохранить каталоги внутри /opt/vcore каталоги в полном составе и с сохранением прав, например, в каталог /root:

```
mkdir -p /root/vcore_backup
cp -rp /opt/vcore/* /root/vcore_backup
```

При наличии балансера необходимо сохранить конфигурацию этого сервиса во всех узлах:

```
cp -p /etc/sysconfig/vcore-balancer /root/vcore_backup
```

Агент ВУ

На всех узлах ВУ необходимо сохранить каталог /opt/vcore в полном составе и с сохранением прав, например, в каталог /root:

```
cp -rp /opt/vcore/* /root/vcore_backup
```

Настройка Pacemaker

Для кластера PostgreSQL необходимо использовать имена узлов кластера, вместо IP-адресов.

Узлы кластера должны быть доступны по именам, если не используется DNS-сервер, то необходимо внести соответствующие записи в файл /etc/hosts. Пример файла - /etc/hosts:

```
10.18.0.120   vcore-control-120-1
10.18.0.115   vcore-control-120-2
10.18.0.93    vcore-control-120-3
10.18.0.40    pgsql-vip
10.18.0.34    bvs-vip
```

Проверьте доступность узлов кластера по именам, они должны отвечать по IP-адресам:

```
ping vcore-control-120-1
```

Остановите кластер Pacemaker:

```
pcs cluster stop
```

Удалите настройки кластера Pacemaker, на каждом узле отдельно:

```
pcs cluster destroy
```

Произведите аутентификацию узлов кластера:

```
pcs host auth vcore-control-120-1 vcore-control-120-2 vcore-control-120-3
```

Создайте кластер:

```
pcs cluster setup vcorecluster vcore-control-120-1 vcore-control-120-2 vcore-control-120-3
```

Запустите и добавьте кластер в автозагрузку:

```
pcs cluster start --all
pcs cluster enable --all
```

Выключите опцию фенсинга:

```
pcs property set no-quorum-policy=ignore
pcs property set stonith-enabled=false
```

Проверьте доступность кластера:

```
pcs status
```

Создайте ресурс VIP:

```
pcs resource create vcore_vip ocf:heartbeat:IPaddr2 ip=10.18.0.33  cidr_netmask=32 op
monitor interval=30s
```

Создайте ресурс балансировщика и настройте соответствующие зависимости:

```
 pcs constraint colocation add vcore_balancer with vcore_vip
 pcs constraint order start vcore_vip then start vcore_balancer
```

Настройка портов

Программный модуль Базис.Virtual Security использует одинаковые порты с сервисами программного модуля Базис.vCore. Измените в настройках vcore broker, control и balancer следующие порты:

- Конфигурация broker - /opt/vcore/broker/config/application.properties:

  ```
  quarkus.http.port=8085
  quarkus.mod-cluster-client.services.api.port=8085
  quarkus.mod-cluster-client.advertise-multicast-port=23367
  ```

- Конфигурация control - /opt/vcore/control/config/application.properties:

  ```
  quarkus.http.port=8086
  broker-rest-api/mp-rest/url=http://IPADDRESS:8085/api
  quarkus.mod-cluster-client.advertise-multicast-port=23368
  ```

  Конфигурация balancer - /opt/vcore/balancer/config/standalone.xml. Для socket-binding "modcluster" и для socket-binding "modcluster-broker" необходимо указать порты :

  ```
  socket-binding name="modcluster" ... multicast-port="23368"/>
  socket-binding name="modcluster-broker" ... multicast-port="23367"/>
  ```

Перезапустите сервис balancer:

```
pcs resource restart vcore_balancer
```

Проверьте статус кластера:

```
pcs status
```

Пример ответа:

```
Cluster name: vcorecluster
Cluster Summary:
  * Stack: corosync
  * Current DC: 192.168.122.11 (version 2.1.4-4.fc35-dc6eb4362e) - partition with quorum
  * Last updated: Fri Oct 21 18:53:47 2022
  * Last change:  Fri Oct 21 14:28:46 2022 by root via cibadmin on vcore-control-120-1
  * 3 nodes configured
  * 2 resource instances configured

Node List:
  * Online: [ vcore-control-120-1 vcore-control-120-2 vcore-control-120-3  ]

Full List of Resources:
  * vcore_vip   (ocf::heartbeat:IPaddr2):        Started  vcore-control-120-1
  * vcore_balancer      (systemd:vcore-balancer):   Started vcore-control-120-1
<------ Ресурс сервиса balancer в статусе Started


Daemon Status:
  corosync: active/enabled
  pacemaker: active/enabled
  pcsd: active/enabled
```

Запустите Broker последовательно на всех узлах:

```
systemctl start vcore-broker
```

Запустите Control последовательно на всех узлах:

```
systemctl start vcore-control
```

Проверьте доступность веб-интерфейса и работоспособность системы.

Установка Virtual Security

Понадобится 2 дополнительных VIP-адреса для веб-интерфейса Virtual Security и для PGSQL. Пример:

```
10.18.0.40    pgsql-vip
10.18.0.34    bvs-vip
```

Настройте адресацию в файле /etc/hosts на всех управляющих узлах, указав реальные имена. Пример:

```
10.18.0.115 vcore-control-120-1
10.18.0.120 vcore-control-120-1
10.18.0.93 vcore-control-120-3
10.18.0.40    pgsql-vip
10.18.0.34    bvs-vip
```

Узнайте текущий часовой пояс:

```
timedatectl
```

Смените часовой пояс, если он установлен некорректно:

```
sudo timedatectl set-timezone Europe/Moscow
```

Скачайте пакет PostgreSQL:

```
dnf install https://download.postgresql.org/pub/repos/yum/reporpms/EL-8-x86_64/pgdg-redhat-repo-latest.noarch.rpm
```

Отключите модуль PostgreSQL по умолчанию:

```
dnf module disable -y postgresql
```

Добавьте порты сервисов в firewall, если используется firewall:

```
firewall-cmd --permanent --add-service=high-availability
firewall-cmd --permanent --add-service=postgresql
firewall-cmd --reload
```

Добавьте репозиторий программного модуля Базис.Virtual Security:

```
cat << 'EOF' > /etc/yum.repos.d/bvs.repo
[bvs]
name=BVS $releasever
baseurl=https://tvs-public:tvs-public@maven.tionix.ru/artifactory/tvs-rpm/dev/centos8/$basearch
enabled=1
gpgcheck=0

[bvs-noarch]
name=BVS $releasever - noarch
baseurl=https://tvs-public:tvs-public@maven.tionix.ru/artifactory/tvs-rpm/dev/centos8/noarch
enabled=1
gpgcheck=0
EOF
```

Проверьте наличие русской локали:

```
locale -a |grep ru
```

При её отсутствии установите пакет:

```
dnf install glibc-langpack-ru
```

Установите пакеты PostgreSQL, на каждом узле:

```
dnf install -y postgresql12 postgresql12-contrib resource-agents-paf  --refresh
```

Установите пакеты программных модулей Базис.vCore и Базис.Virtual Security на каждом узле:

```
dnf install tionix-tvs-balancer* tionix-tvs-configure* tionix-tvs-core* tionix-tvs-openjdk* tionix-tvs-security* tionix-tvs-web*
```

Важно

Следующий блок выполняется последовательно, на каждом узле кластера.

Установите пакет, выполняющий инициализацию БД:

```
dnf install tionix-tvs-db*
```

Добавьте ранее выбранные VIP-адреса веб-интерфейса и кластера PostgreSQL:

```
ip address add 10.18.0.34/32 dev eth0
ip address add 10.18.0.40/32 dev eth0
```

Настройте Virtual Security только на VIP узле, указывая для балансера адрес 10.18.0.34, а адрес PostgreSQL 10.18.0.40 (pgsql-vip):

```
tvs_configure.sh
```

Пример выполнения мастера настройки:

```
tvs_configure.sh

Выберите язык (select language) (ru,en)  [ru]:
Укажите сеть доступа к серверу PostgreSQL (пример, 192.168.0.0/24): 10.18.0.0/24 Сеть из которой
будет осуществляться подключение к серверу БД
Укажите IP-адрес сервера базы данных (пример, 192.168.0.1): 10.18.0.40 - VIP адрес БД, в
нашем случае pgsql-vip
Укажите имя базы данных [tvs]:
Введите пользователя базы данных [tvs]:
Введите пароль пользователя базы данных [tvs]:
Повторите ввод пароля [tvs]:
Настройка PostgreSQL...
Initializing database ... OK

Запуск PostgreSQL...
Created symlink /etc/systemd/system/multi-user.target.wants/postgresql-12.service → /
usr/lib/systemd/system/postgresql-12.service.
Настройка базы данных...
ALTER ROLE
CREATE SCHEMA
CREATE SCHEMA
Firewall не активен. Для доступа к серверу PostgreSQL необходимо открыть доступ к tcp порту 5432
Сохранение конфигурации...
Укажите имя узла сервера балансировки (в кластере не должно быть одинаковых имен) [vcore-control-12
0-2.novalocal-balancer]:   - можно оставить по дефолту
Укажите IP-адрес для сервера балансировки (пример, 192.168.0.1): 10.18.0.34 - vip адрес, по
которому будем подключаться к web-управлялке TVS
Укажите IP-адрес внешнего сервера времени (пустой ввод - будут использованы сервера по умолчанию)
[]:
Настроить сервер синхронизации времени на узле балансировки? (y,n) [y]: n
Внимание! Для настройки SSL необходимо иметь подготовленный файл контейнера с закрытым ключом и
сертификатом в формате pfx или java-keystore
Кроме того, если сертификат не является доверенным, его (или его корневые сертификаты) необходимо
добавить в список доверенных для
виртуальной машины Java с помощью утилиты keytool на все узлы, где запущены модули tvs-core и
 tvs-web
Настроить SSL для узла балансировки? (y,n) [n]: n
Настройка сервера балансировки...
Firewall не активен. Для работы сервера балансировки необходимо, чтобы был открыт доступ на порт
8080 и 8090
Запуск сервера балансировки, это может занять продолжительное время...
Сохранение конфигурации...
Укажите имя узла сервера безопасности (в кластере не должно быть одинаковых имен) [vcore-control-12
0-2.novalocal-security]:
Укажите IP-адрес для сервера безопасности (пример, 192.168.0.1): 10.18.0.115 - указывает
локальный ip адрес сервера кластера
Настройка сервера безопасности...
Введите пароль администратора LDAP сервера [password]:
Повторите ввод пароля [password]:
Firewall не активен. Для работы сервера безопасности необходимо открыть доступ к tcp портам: 8082,
10640, 54200, 56300, 56400 - также доступ на адреса 224.1.0.105/32 и 230.0.0.0/24
Запуск сервера безопасности...
Сохранение конфигурации...
Укажите имя узла сервера администрирования (в кластере не должно быть одинаковых имен) [vcore-
control-120-2.novalocal-core]:
Укажите IP-адрес для сервера администрирования (пример, 192.168.0.1) [10.18.0.34]: 10.18.0.11
5 - указывает локальный ip адрес сервера кластера
Укажите URL-системы (пример, http://my.domain.ru:8080): http://10.18.0.34:8080
Настройка сервера администрирования...
Firewall не активен. Для работы сервера безопасности необходимо открыть доступ к tcp портам: 8081,
5673, 54200, 56300, 56400 - также доступ на адреса 224.1.0.105/32 и 230.0.0.0/24
Запуск сервера администрирования...
```

```
Сохранение конфигурации...
Укажите имя узла web сервера (в кластере не должно быть одинаковых имен) [vcore-control-120-2.no
valocal-web]:
Укажите IP-адрес для web сервера (пример, 192.168.0.1) [10.18.0.34]: 10.18.0.115 -
указывает локальный ip адрес сервера кластера
Настройка web сервера...
Firewall не активен. Для работы web сервера необходимо открыть доступ к tcp портам: 8083, 54200,
56300, 56400 - также доступ на адреса 224.1.0.105/32 и 230.0.0.0/24
Запуск web сервера, это может занять продолжительное время...
Сохранение конфигурации...
```

Проверьте статус Virtual Security:

```
systemctl list-units |grep tvs
tvs-balancer.service                                          loaded active running
  Tionix Vrtual Security Banancer Service
tvs-core.service                                              loaded active running
  Tionix Vrtual Security Core Service
tvs-security.service                                          loaded active running
  Tionix Vrtual Security Service
tvs-web.service                                               loaded active running
  Tionix Vrtual Security Web Service
```

Проверьте статус PostgreSQL:

```
systemctl list-units |grep postgre
postgresql-12.service                                         loaded active running
  PostgreSQL 12 database server
```

Остановите сервис балансировки:

```
systemctl stop tvs-balancer
```

Удалите VIP-адреса веб-интерфейса и кластера PostgreSQL:

```
ip address delete 10.18.0.34/32 dev eth0
ip address delete 10.18.0.40/32 dev eth0
```

Удалите пакет инициализации БД:

```
dnf remove tionix-tvs-db
```

Повторите операции на всех узлах кластера.

После завершения первоначальной инициализации, выполните на всех узлах кластера.

Остановите сервисы Virtual Security:

```
systemctl stop tvs-security tvs-core tvs-web tvs-balancer
```

Остановите PostgreSQL:

```
systemctl stop postgresql-12
```

Настройка кластера PostgreSQL

Выберите один, любой, узел для роли master.

Настройте master-узел. Настройка должна осуществляться от имени пользователя `postgres`:

```
su - postgres
```

Укажите значение параметра `listen_addresses` в - `postgres/12/data/postgresql.conf`:

```
listen_addresses = '*'
```

Добавьте параметры в - `postgres/12/data/postgresql.conf`:

```
hba_file = '/var/lib/pgsql/12/pg_hba.conf'
wal_keep_segments = 32
include = '../standby.conf'
```

Укажите имя узла вместо $(hostname -s) и VIP-адрес PostgreSQLв - `postgres/12/standby.conf`:

```
primary_conninfo = 'host=10.18.0.40 application_name=$(hostname -s)'
```

Удалите конфигурацию `pg_hba.conf`:

```
rm ~postgres/12/data/pg_hba.conf
```

Сформируйте конфигурацию `pg_hba.conf`, вписав точное имя хоста вместо $(hostname -s):

```
cat <<EOF > ~postgres/12/pg_hba.conf
local all         all                     trust
host  all         all      0.0.0.0/0      trust
host  all         all      ::/0           trust
# forbid self-replication from VIP
host  replication postgres 10.18.0.40/32  reject
# forbid self-replication its own IP
local replication all                     reject
host  replication all      10.18.0.115 reject
host  replication all      127.0.0.0/8    reject
host  replication all      ::1/128        reject
# allow any standby connection
host  replication postgres 0.0.0.0/0      trust
host tvs tvs 10.18.0.0/24 md5
EOF
```

Осуществите выход из под пользователя `postgres`:

```
exit
```

Запустите PostgreSQL:

```
systemctl start postgresql-12
```

Добавьте VIP-адрес для PostgreSQL:

```
ip address add 10.18.0.40/32 dev eth0
```

Последовательно, настройте два оставшихся узла в роли slave. Настройка должна осуществляться от имени пользователя `postgres`:

```
su - postgres
```

Укажите точное имя узла вместо $(hostname -s):

```
cat <<EOF > ~postgres/12/standby.conf

primary_conninfo = 'host=10.18.0.40 application_name=$(hostname -s)'
EOF
```

Сформируйте конфигурацию `pg_hba.conf`, вписав точное имя хоста вместо $(hostname -s):

```
cat <<EOF > ~postgres/12/pg_hba.conf
```

```
local all         all                     trust
host  all         all      0.0.0.0/0      trust
host  all         all      ::/0           trust
# forbid self-replication from VIP
host  replication postgres 10.18.0.40/32  reject
# forbid self-replication its own IP
local replication all                     reject
host  replication all      $(hostname -s) reject
host  replication all      127.0.0.0/8    reject
host  replication all      ::1/128        reject
# allow any standby connection
host  replication postgres 0.0.0.0/0      trust
host tvs tvs 10.18.0.0/24 md5
EOF
```

Удалите ненужные файлы из каталога:

```
rm -rf /var/lib/pgsql/12/data/*
```

Создайте репликацию из главного (master) узла в текущий (slave) узел:

```
pg_basebackup -h pgsql-vip -D ~postgres/12/data/ -X stream -P
```

Пометьте узел как slave:

```
touch ~postgres/12/data/standby.signal
```

Выйдите из пользователя `postgres`:

```
exit
```

Запустите узел slave:

```
systemctl start postgresql-12
```

Проверьте статус узла:

```
systemctl status postgresql-12
```

Войдите в БД:

```
sudo -u postgres psql -U postgres -h pgsql-vip
```

Проверьте запись о репликации:

```
select * from pg_stat_replication;

postgres=# select * from pg_stat_replication;
  pid  | usesysid | usename  | application_name | client_addr | client_hostname |
client_port |         backend_start         |
-------+----------+----------+------------------+-------------+-----------------
+-------------+-------------------------------+
 39759 |       10 | postgres | control-120-2    | 10.18.0.120 | control-120-2   |
52416 | 2023-06-02 13:04:29.352829+03 |

(1 row)
```

Выйдите из БД:

```
exit
```

Настройка кластера Pacemaker для PostgreSQL

Теперь, когда PostgreSQL уже в master-slave конфигурации, необходимо настроить автоматическое переключение Pacemaker.

Отключите PostgreSQL. Сначала на slave узлах, затем на master:

```
systemctl disable --now postgresql-12
```

Удалите VIP-адрес с мастер узла:

```
ip addr del 10.18.0.40/32 dev eth0
```

Создайте ресурс VIP-кластера PostgreSQL:

```
pcs resource create pgsql-pri-ip ocf:heartbeat:IPaddr2 ip=10.18.0.40 cidr_netmask=32 op
monitor interval=10s
```

Убедитесь, что версия не ниже resource-agents-paf-2.3.0-1.noarch (v2.2 не подходит для postgresql-12):

```
rpm -qa|grep paf
```

Создайте ресурс `postgresql`:

```
pcs resource create pgsqld ocf:heartbeat:pgsqlms bindir=/usr/pgsql-12/bin pgdata=/var/
lib/pgsql/12/data op start timeout=60s op stop timeout=60s op promote timeout=30s op
demote timeout=120s op monitor interval=15s timeout=10s role="Master" op monitor
interval=16s timeout=10s role="Slave" op notify timeout=60s promotable notify=true
```

Добавьте метадату количества master-узлов в кластере:

```
pcs resource meta pgsqld master-max=1
```

Зависимости ip и postgre:

```
pcs constraint colocation add pgsql-pri-ip with master pgsqld-clone INFINITY
pcs constraint order promote pgsqld-clone then start pgsql-pri-ip symmetrical=false
 kind=Mandatory
pcs constraint order demote pgsqld-clone then stop pgsql-pri-ip symmetrical=false
 kind=Mandatory
```

Отключите зависимости tvs-systemd. Скомментируйте строку `#Wants=`:

```
systemctl edit --full tvs-balancer
systemctl edit --full tvs-security
systemctl edit --full tvs-core
systemctl edit --full tvs-web
```

Отключите контроль systemd над программным модулем Базис.Virtual Security:

```
systemctl disable --now tvs-security tvs-core tvs-web tvs-balancer
```

Создайте VIP для программного модуля Базис.Virtual Security:

```
pcs resource create tvs-pri-ip ocf:heartbeat:IPaddr2 ip=10.18.0.34 cidr_netmask=32 op
monitor interval=10s --group tvs
#
pcs resource create tvs-balancer systemd:tvs-balancer --group tvs

pcs resource create tvs-core systemd:tvs-core clone
pcs resource create tvs-security systemd:tvs-security clone
pcs resource create tvs-web systemd:tvs-web clone
#
pcs constraint order tvs-security-clone then tvs-core-clone
pcs constraint order tvs-security-clone then tvs-web-clone
```

```
pcs constraint order tvs-core-clone then tvs-web-clone
pcs constraint order tvs-web-clone then tvs-balancer
#
pcs constraint colocation add tvs-pri-ip with tvs-balancer INFINITY
pcs constraint order pgsqld-clone then tvs-security-clone
```

Проверьте статус кластера:

```
pcs status
```

Проверьте доступность веб-интерфейсов программных модулей Базис.Virtual Security и vCore.

Приложение 4. Список портов сервисов программного модуля Базис.Virtual Security

Компонент, инициализирующий подключение (исх.трафик)	Компонент, обслуживающий подключение (вх.трафик)	Протокол/порт (указывается порт сервиса, который обслуживает подключение)	Назначение порта (содержание трафика)	Файл настройки компонента	Параметр настройки компонента
	tvs-agent	HTTP/8010	Сетевой порт сервиса.	`/opt/tvs/agent/config/application.properties`	`quarkus.http.port=<значение>`
tvs-agent	tvs-core	TCP/5673	Подключение к брокеру сообщений. Трафик может идти в обе стороны.	`/opt/tvs/agent/config/application.properties`	`tvs-agent: agent.provider.jms.broker-uris=failover:(amqps://security-1:<значение>?transport.trustAll=true)`
HTTP-клиент	tvs-security-proxy	HTTP/8099	Порт доступа проксированных API или иных HTTP-сервисов за Virtual Security.	`/opt/tvs/security-proxy/config/application.properties`	`quarkus.http.port=<значение>`
Сервисы Virtual Security	tvs-balancer	UDP/23364	Мультикаст-порт для регистрации сервиса в службе балансировки.	`/opt/tvs/security-proxy/config/application.properties`	`tvs-security-proxy: quarkus.mod-cluster-client.advertise-multicast-port=<значение>`

Компонент, инициализирующий подключение (исх.трафик)	Компонент, обслуживающий подключение (вх.трафик)	Протокол/порт (указывается порт сервиса, который обслуживает подключение)	Назначение порта (содержание трафика)	Файл настройки компонента	Параметр настройки компонента
tvs-security-proxy	tvs-core и tvs-auth через tvs-balancer	HTTPS/8443	Порт балансировщика для доступа до сервисов Core и Auth.	`/opt/tvs/security-proxy/config/application.properties`	`proxy.admin.url=https://tvs-test:<значение>` `proxy.security.url=https://tvs-test:<значение>`
tvs-balancer	vs-auth (tvs-security)	HTTP/8082	Порт конечного доступа до сервиса Auth через балансировщик.		
LDAP-клиент	LDAP-сервис	TCP/10389	Обычный порт подключения к встроенному LDAP-сервису.	`/application.properties` `(сервис vs-auth)`	`ldap.transport.tcp.port=<значение>`
LDAP-клиент	LDAP-сервис	TCP/10638	Шифрованный порт подключения к встроенному LDAP-сервису.	`/application.properties` `(сервис vs-auth)`	`ldap.transport.tcp-ssl.port=<значение>`

Компонент, инициализирующий подключение (исх.трафик)	Компонент, обслуживающий подключение (вх.трафик)	Протокол/порт (указывается порт сервиса, который обслуживает подключение)	Назначение порта (содержание трафика)	Файл настройки компонента	Параметр настройки компонента
	tvs-balancer	HTTP/8080	Основной порт балансировщика.	`?/standalone.xml`	`<socket-binding name="http" port="${jboss.http.port:<значение>}"/>`
tvs-balancer	tvs-core	HTTP/8081	Основной порт сервиса tvs-core, по которому осуществляется доступ до сервиса через балансировщик.	`? /application.yaml`	`quarkus.http.port=<значение>`
tvs-balancer	vs-auth (tvs-security)	HTTP/8082	Основной порт сервиса vs-auth (tvs-security), по которому осуществляется доступ до сервиса через балансировщик.	`? /keycloak.conf`	`http.port=<значение>`

Компонент, инициализирующий подключение (исх.трафик)	**Компонент, обслуживающий подключение (вх.трафик)**	**Протокол/ порт (указывается порт сервиса, который обслуживает подключение)**	**Назначение порта (содержание трафика)**	**Файл настройки компонента**	**Параметр настройки компонента**
tvs-balancer	tvs-web	HTTP/8083	Основной порт сервиса tvs-web, по которому осуществляется доступ до сервиса через балансировщик.	`? /application.properties`	`quarkus.http.port=<значение>`
Core, Auth,Proxy	PostgreSQL	TCP/5432	Основной порт СУБД PostgreSQL.	`tvs-core: ?/application.yaml` `tvs-auth: ?/keycloak.conf`	`tvs-core: datasource:jdbc:url: jdbc:postgresql://localhost:<значение>/vs` `tvs-auth: db-url-port=<значение>`
tvs-core		TCP/5673	Основной порт брокера сообщений AvtiveMQ Artemis		
tvs-balancer		TCP/8090	Порт состояния балансировщика.	`? /standalone.xml`	`<socket-binding name="mcmp-management" interface="private" port="${jboss.mcmp.port:<значение>}"/>`