KeyStack SE

Руководство администратора

2025

Содержание

1. ВВЕДЕНИЕ 5
2. НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ ПО «KeyStack SE» 6
2.1 Назначение и описание функций безопасности 6
2.2 Среда функционирования 6
2.3 Требования к программному обеспечению 7
2.4 Требования к сегментации сети при настройке ПО «KeyStack SE» 8
3. ДЕЙСТВИЯ ПО ПРИЕМКЕ 12
4. ДЕЙСТВИЯ ПО БЕЗОПАСНОЙ УСТАНОВКЕ И НАСТРОЙКЕ СРЕДСТВА 13
4.1 Указания по эксплуатации 13
4.2 Установка ПО «KeyStack SE» 13
4.2.1 Целевая аудитория 13
4.2.2 Состав дистрибутива 13
4.2.3 Создание QCOW2 образа из ISO образа Platform V SberLinux OS Server 14
4.2.4 Подготовка серверной инфраструктуры 15
4.2.5 Установка и настройка компонентов 17
4.2.5.1 Установка KeyStack LCM 17
4.2.5.2 Подготовка baremetal-узлов для установки KeyStack 20
4.2.5.3 Создание региона 26
4.2.5.4 Развёртывание сети региона 37
4.2.6 Подключение и настройка СХД 38
4.2.7 Подключение служебных сервисов 42
4.2.7.1 Подключение Портала администратирования 42
4.2.7.2 Подключение сервисов мониторинга 43
4.2.7.3 Подключение сервисов аудита 45
4.2.8 Проверка работоспособности системы 48
4.2.9 Создание резервной копии LCM-сервера 48

4.2.10 Восстановление LCM-сервера из резервной копии 48
4.2.11 Настройка аудита логов в OpenSearch Dashboards 49
4.3 ПОДГОТОВКА К РАБОТЕ 51
5.1 Проверка соответствия требованиям 51
5.2 Подготовка ПО «KeyStack SE» к работе 51
5.3 Подключение к Порталу администратора 51
5. ПРИНЦИПЫ БЕЗОПАСНОЙ РАБОТЫ ПО «KeyStack SE» 66
6.1 Защита ПО «KeyStack SE» от несанкционированного доступа 66
6. АВАРИЙНЫЕ СИТУАЦИИ 68
7.1 Сбои и ошибки эксплуатации 68
7.2 Признаки наличия сбоев 68
7.3 Действия при наличии сбоев 68
Порядок проверки сервисов при аварийном переключении (failover) 68
Проверка состояния кластера RabbitMQ 68
Пайплайн по зачистке RabbitMQ и рестарту сервисов 69
Проверка состояния кластера MariaDB 69
Проверка вспомогательных компонентов мониторинга и логирования 70
Получить список состояний компонентов сервиса nova в OpenStack 70
Проверка журналов компонентов neutron на предмет актуальных ошибок 71
Failover одного управляющего узла 71
Failover двух управляющих узлов 72
Failover всех управляющих узлов 72
Failover вычислительного узла 72
Устранение неисправностей при создании/восстановлении резервной копии LCM-сервера 73
Примечания 74

Сокращения

ОО	–	Объект оценки
ПО	–	Программное обеспечение
ОП	–	Облачная платформа
СУБД	–	Система управления базами данных
СХД	–	Система хранения данных
ФСТЭК России	–	Федеральная служба по техническому и экспортному контролю России
LCM	–	(англ. Life-Cycle Manager), Система управления инсталляциями (экземплярами) ПО «KeyStack SE»
BMC	–	(англ. Baseboard Management Controller), Контроллер управления материнской платой сервера
ВМ, VM	–	Виртуальная машина

1. ВВЕДЕНИЕ

Идентификация документа

Наименование документа	Программное обеспечение «Облачная платформа KeyStack SE v.1». Руководство администратора
Версия документа	Версия 1.0
Обозначение документа	1087746411378.62.01.29.000.001 91
Идентификация ОО	Программное обеспечение «Облачная платформа KeyStack SE v.1». (ПО «KeyStack SE»)

Уровень доверия	ПО «KeyStack SE» соответствует 4 уровню доверия, в соответствии с Требованиями по безопасности информации, устанавливающими уровни доверия к средствам технической защиты информации и средствам обеспечения безопасности информационных технологий, утвержденными приказом ФСТЭК России от 2 июня 2020 г. № 76

2. НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ ПО «KeyStack SE»

2.1 Назначение и описание функций безопасности

Основным назначением программного обеспечения «Облачная платформа KeyStack SE v.1» (далее – ПО «KeyStack SE», Платформа) является создание гибкой, масштабируемой и безопасной инфраструктуры виртуализации с автоматизированными процессами развертывания и управления.

ПО «KeyStack SE» представляет собой совокупность функциональных подсистем, обеспечивающих выполнение функций по виртуализации вычислительных ресурсов, созданию и управлению виртуальными машинами, мониторингу состояния инфраструктуры и осуществляющих централизованное управление средой виртуализации.

ПО «KeyStack SE» реализует следующие меры защиты согласно требованиям документа «Требования по безопасности информации к средствам виртуализации» (введен в действие приказом ФСТЭК России № 187 от октября 2022 г.):

1) функционирование в среде хостовой операционной системы, соответствующей 4 классу защиты и 4 уровню доверия;
2) реализация следующих функций безопасности:
   - доверенная загрузка виртуальных машин;
   - контроль целостности;
   - регистрация событий безопасности;
   - ролевой метод управления доступом;
   - резервное копирование;
   - ограничение программной среды;
   - управление потоками информации;
   - защита памяти;
   - идентификация и аутентификация пользователей;
   - централизованное управление образами виртуальных машин и виртуальными машинами.

2.2 Среда функционирования

ПО «KeyStack SE» функционирует в среде, которая сконфигурирована и контролируется администраторами в соответствии с эксплуатационной документацией.

Платформа использует для управления инфраструктурой узлы нескольких типов:

LCM (Seed node) — главный узел управления сервисной инфраструктурой Платформы. На нем хранится кодовая база основных компонентов Платформы и основная конфигурация узлов остальных типов. С этого узла начинается развертывание Платформы и управление ее жизненным циклом.

Controller — узел с управляющими модулями вычислительных ресурсов, сетевых агентов и вспомогательными службами. В некоторых случаях может совмещать роль узла Network.

Compute — узел с гипервизором KVM, управляет виртуальными машинами (ВМ) основной инфраструктуры. Для обеспечения сетевой связности ВМ используется агент сетевой службы.

Storage — узел с дисками блочных хранилищ. Блочные хранилища используются для организации индивидуального и/или совместного дискового пространства для ВМ.

Network — узел с компонентами программно-определяемых сетей.

Характеристики аппаратного элемента вычислительной инфраструктуры (сервера) ПО «KeyStack SE» приведены в Таблице 1.

Таблица 1 – Характеристики аппаратного элемента вычислительной инфраструктуры (сервера) ПО «KeyStack SE»

№ п/п	Название характеристик	Назначение серверов			
		Сервер вычислительных ресурсов (Controller-сервер)	**Сервер гипервизора KVM[1] (Compute-сервер)**	**Сервер сетевых служб (Network-сервер)[2]**	**Сервер управления сервисной инфраструктурой (LCM-сервер)**
1	Архитектура CPU	x86_64	x86_64	x86_64	x86_64
2	Количество CPU	2	2	2	2
3	Количество ядер одного CPU без учёта мультипоточности.	12	12	8	12
4	Объём RAM, Гб	64	64	32	128
5	Системные диски	2x 480GB, Enterprise SSD MixUse (RAID1)	2x 480GB, Enterprise SSD MixUse (RAID1)	2x 480GB, Enterprise SSD MixUse (RAID1)	2x 480GB, Enterprise SSD MixUse (RAID1)
6	Диски для данных	2x 1.9TB, Enterprise SSD MixUse (RAID1)	не требуется	не требуется	2x 3.8TB, Enterprise SSD MixUse (RAID1)
7	Количество портов 10/25 Gbps Ethernet	4	4	6	2
8	Количество портов FC SAN (HBA)	2	2	0	0

2.3 Требования к программному обеспечению

ПО «KeyStack SE» устанавливается на виртуальных машинах с архитектурой x86-64 под управлением ОС Platform V SberLinux OS Server (Сертификат ФСТЭК России №4884 от 04.12.2024, действителен до 04.12.2029. ПО «KeyStack SE» функционирует в среде,

[1]При росте количества серверов гипервизоров KVM требования к серверам пересматриваются в сторону увеличения.
[2]Опционально. В ряде случае данную роль совмещает в себе сервер вычислительных ресурсов (Controller-сервер).

которая сконфигурирована и контролируется администратором среды функционирования в соответствии с эксплуатационной документацией.

Система обеспечивает функционирование на базе следующего общего программного обеспечения, указанного в Таблице 2

Таблица 2 – Программная среда функционирования ПО «KeyStack SE»

№	Назначение сервера	Элемент среды функционирования	Наименование
1	Сервер управления сервисной инфраструктурой (LCM-сервер)	ОС	Platform V SberLinux OS Server (Сертификат ФСТЭК России №4884 от 04.12.2024, действителен до 04.12.2029)
		Компонент ОС	Podman (входит в состав сертифицированного образа Platform V SberLinux OS Server)
		СУБД	Platform V Pangolin (Сертификат ФСТЭК России №4491 от 15.12.2021, действителен до 15.12.2026)
2	Сервер гипервизора KVM (Compute-сервер)	ОС	Platform V SberLinux OS Server (Сертификат ФСТЭК России №4884 от 04.12.2024, действителен до 04.12.2029)
		Компонент ОС	Podman (входит в состав сертифицированного образа Platform V SberLinux OS Server)
3	Сервер сетевых служб (Network-сервер)	ОС	Platform V SberLinux OS Server (Сертификат ФСТЭК России №4884 от 04.12.2024, действителен до 04.12.2029)
		Компонент ОС	Podman (входит в состав сертифицированного образа Platform V SberLinux OS Server)
4	Сервер вычислительных ресурсов (Controller-сервер)	ОС	Platform V SberLinux OS Server (Сертификат ФСТЭК России №4884 от 04.12.2024, действителен до 04.12.2029)
		Компонент ОС	Podman (входит в состав сертифицированного образа Platform V SberLinux OS Server)

2.4 Требования к сегментации сети при настройке ПО «KeyStack SE»

Описание разделения сети на логические сегменты с помощью VLAN (виртуальной локальной сети) представлено в Таблице 3.

Таблица 3 – Описание сегментации сети в ПО «KeyStack SE»

Роль, нода	VLAN	Описание
LCM	MGMT External	Сеть внешнего управления. Модули компонентов (контейнеры) ОП доступные извне и находящиеся на поверхности атаки: - Nginx (port ext)
	PXE	Загрузка узлов по сети для первоначальной установки ОП: - DHCP/TFTP/HTTP загрузка с LCM ноды
	RMI	Remote Management Interface - управление узлом control через Web/Redfish
	MGMT Internal	1) Доступ к сервисам LCM: - OpenStack Bifrost - RPM-repo - Vault - GitLab CI - Netbox 2) Второй интерфейс Nginx (port int) для внутреннего взаимодействия сервисов LCM
Controller	MGMT External	Сеть внешнего управления. Модули компонентов (контейнеры) ОП доступные извне: Nginx (внутри adminui-frontend), AdminUI-backend, Nova_API, Nova_novncproxy, Nova_serialproxy, Heat_API, Placement_API, DRS_API, Neutron_server, Octavia_API, Barbican_API, Opensearch, Cinder_API, Glance_API, HAProxy, Grafana, KeyStone, Prometheus_alertmanager
	PXE	Сеть PXE-загрузки контроллера с LCM для первоначальной установки ОП: - DHCP/TFTP/HTTP загрузка с LCM ноды
	RMI	Remote Management Interface - управление узлом control через Web/Redfish
	MGMT Internal	Управляющая сеть (служебный трафик). Для взаимодействия компонентов ОП между собой:

		KeyStone, KeyCloak[3], AdminUI-frontend, Nova, Heat, Hashicorp consul, Neutron, FRR, Octavia, FRR, OVS, OVN, Barbican, Prometheus, AlertManager, Grafana, Vicoriametrics, Fluentd, Opensearch, Cinder, multipathd, iscsid, Redis, Redis Sentinel, Memcached, MariaDB, ProxySQL, Bird, ExaBGP, RabbitMQ
	CTRL DATA (tenant VLAN)	Пользовательские сети (DHCP/Metadata). Отдельно маршрутизируются от MGMT External и Internal VLAN
Compute	MGMT Internal	Управляющая сеть (служебный трафик). Для взаимодействия компонентов ОП между собой: KeyStone , AdminUI-frontend, Nova, Heat, Hashicorp consul, Neutron, FRR, Octavia, FRR, OVS, OVN, Barbican, Prometheus, AlertManager, Grafana, Vicoriametrics, Fluentd, Opensearch dashboards, Cinder, multipathd, iscsid, Redis, Redis Sentinel, Memcached, MariaDB, ProxySQL, Bird, ExaBGP, RabbitMQ
	PXE	PXE-загрузка compute-узла для первоначальной установки ОП: - DHCP/TFTP/HTTP загрузка с LCM ноды
	RMI	Remote Management Interface - управление узлом control через Web/Redfish
	CMP DATA (tenant VLAN)	Пользовательские сети для VM. Отдельно маршрутизируются от MGMT External и Internal VLAN

Примечания:

- CTRL DATA и CMP DATA (tenant networks) не смешиваются с VLAN-ами управления (MGMT Internal, MGMT External, PXE, RMI) и должны разделяться с использованием сертифицированных средств.

- Сегмент MGMT External также используется для управления и мониторинга Платформой. Пользователям доступ предоставляется для управления разрешенными им объектам.

Доступ пользователей ПО «KeyStack SE» должен предоставляться только для сетей/сервисов:

[3] Здесь предполагается сертифицированный продукт, работающий по протоколам OpenID/Oath, в основе которого использовано ПО «Keycloak»

1. Пользовательский (tenant) трафик VLAN-ов CTRL DATA и CMP DATA.
2. Сеть внешнего управления MGMT External:
    - Nginx (LCM) https://[DNS имя ОП]:443
    - KeyStone https://[DNS имя ОП]:5000
    - Nginx (adminui-frontend) https://[DNS имя ОП]:12999
    - AdminUI-backend https://[DNS имя ОП]:13000
    - Nova_API https://[DNS имя ОП]:8775
    - Nova_novncproxy https://[DNS имя ОП]:6080
    - Nova_serialproxy https имя ОП]:6083
    - Heat_API https://[DNS имя ОП]:8004
    - Placement_API https://[DNS имя ОП]:8780
    - DRS_API https://[DNS имя ОП]:12998
    - Neutron_server https://[DNS имя ОП]:9696
    - Octavia_API https://[DNS имя ОП]:9876
    - Barbican_API https://[DNS имя ОП]:9311
    - Opensearch_dashboards https://[DNS имя ОП]:5601
    - Cinder_API https://[DNS имя ОП]:8776
    - Glance_API https://[DNS имя ОП]:9292
    - Haproxy https://[DNS имя ОП]:5140
    - Grafana https://[DNS имя ОП]:3000

    Prometheus_alertmanager https://[DNS имя ОП]:9093

3. ДЕЙСТВИЯ ПО ПРИЕМКЕ

Приемка поставленного ПО «KeyStack SE» осуществляется в соответствии с указаниями, содержащимися в документе «Программное обеспечение «Облачная платформа KeyStack SE v.1». Технические условия», 1087746411378.62.01.29.000.001 ТУ.

4. ДЕЙСТВИЯ ПО БЕЗОПАСНОЙ УСТАНОВКЕ И НАСТРОЙКЕ СРЕДСТВА

4.1 Указания по эксплуатации

Персонал, ответственный за функционирование ОО, должен обеспечивать надлежащее функционирование ОО в соответствии с документами:

- Программное обеспечение «Облачная платформа KeyStack SE v.1». Руководство пользователя (1087746411378.62.01.29.000.001 34);
- Программное обеспечение «Облачная платформа KeyStack SE v.1». Руководство администратора (1087746411378.62.01.29.000.001 91);
- Программное обеспечение «Облачная платформа KeyStack SE v.1». Формуляр (1087746411378.62.01.29.000.001 ФО).

4.2 Установка ПО «KeyStack SE»

В общем виде развертывание Платформы состоит из шагов:

- Подготовка инфраструктуры.
- Распаковка дистрибутива Платформы.
- Установка основных компонентов Платформы.
- Настройка сетевой связности.
- Ручной запуск пайплайнов.
- Подключение и настройка служебных сервисов.
- Проверка работоспособности Платформы.

4.2.1 Целевая аудитория

Целевой аудиторией являются системные администраторы, которые разворачивают и настраивают ПО «KeyStack SE» на целевой инфраструктуре.

К системным администраторам предъявляются требования:

- Навыки системного администрирования Linux.
- Навыки организации и настройки сетевой связности между узлами.
- Понимание принципа работы сервисов OpenStack.
- Умение собирать диагностические данные для эскалации сложных проблем производителю.

4.2.2 Состав дистрибутива

Установочный файл включает в себя:

- архив podman-образов компонентов LCM, включая RPM-repo, Vault, GitLab, NetBox, Nginx;
- необходимые системные пакеты (rpm), для развертывания LCM без доступа к интернету;
- образ операционной системы (qcow2) для автоматизированной установки на baremetal-узлы;

- скрипты инсталляции и развертывания LCM.

4.2.3 Создание QCOW2 образа из ISO образа Platform V SberLinux OS Server

Процесс состоит из четырёх фаз: подготовка хоста, создание пустого qcow2, инсталляция ОС SberLinux из ISO образа в виртуальной машине и post-clean-обработка (cloud-init, усадка и сжатие). Методика опирается на стандартные инструменты KVM (qemu-img, virt-install).

Перед началом убедитесь, что ваша система поддерживает виртуализацию (VT-x или AMD-V) и имеет достаточно дискового пространства, оперативной памяти и процессоров. Также потребуется доступ к ISO-образу Platform V SberLinux OS Server (находится в архиве дистрибутива продукта).

1. **Подготовка среды**

Установите предварительно следующие пакеты (в случае, если они отсутствуют):

sudo dnf install -y qemu-kvm qemu-img virt-install libvirt-daemon \cloud-utils-growpart guestfs-tools

sudo systemctl enable --now libvirtd

Создайте рабочую директорию (пример: ~/images/sberlinux[4]) и скачайте актуальный ISO из предоставленного дистрибутива («Platform V SberLinux OS Server»)

Необязательно: для ускорения и упрощения конфигурирования процесса инсталляции используйте Kickstart (/home/<username>/ks.cfg).

Также можно использовать инструкцию разработчика ОС Platform V Sberlinux Server OS, с описанием различных сценариев установки ОС:

https://platformv.sbertech.ru/docs/public/SLO/9.1.0-fstec/common/documents/pfstec/administration-guide/2-creating-vm.html#id3

2. **Создание пустого qcow2 диска**

export DISK=~/images/sberlinux/sberlinux-base.qcow2[5]
qemu-img create -f qcow2 "$DISK" 10G

Формат qcow2 поддерживает тонкое выделение места и сжатие, поэтому 10 ГиБ файл займёт считанные мегабайты до заполнения.

3. **Установка SberLinux во временной ВМ**

Выполните следующую команду (дополнить или удалить параметры при необходимости):

```
virt-install \
  --name     sberlinux-build \
  --memory   4096 --vcpus 2 \
  --disk     path=$DISK,format=qcow2,bus=virtio \
  --cdrom    ~/ISO/sberlinux_fstec-9.1-x86_64-dvd.iso \
  --os-variant rhel8.6 \
  --network  bridge=virbr0,model=virtio \
  --graphics none \
  --extra-args "console=ttyS0,115200 inst.text"
```

[4] Указать свой путь при необходимости. Использовать созданный путь далее в процессе конфигурации, заменив приведенные примеры.

[5] Приведен пример. Заменить на актуальный путь и удобное наименование файла.

Данная команда создаст виртуальную машину с 4096 МБ памяти, 2 виртуальными процессорами, диском 20 ГБ в формате qcow2 и запустит установку ОС из ISO (скорректировать путь при необходимости). Следуйте инструкциям на экране для завершения установки.
После завершения установки выключите ВМ: virsh shutdown sberlinux-build.

4. Post-clean и cloud-ready подготовка

1) Подключите диск для правки:

```
sudo virt-customize -a $DISK \
  --run-command 'sed -i "/^HWADDR\|^UUID/d" /etc/sysconfig/network-scripts/ifcfg-*"'
```

Удаляем сетевые UUID, чтобы каждый клон получал уникальные параметры.

2) Устанавливаем cloud-init (если не был выбран при инсталляции):

```
sudo virt-customize -a $DISK --install cloud-init
```

3) Обнуляем идентификаторы и ключи:

```
sudo virt-sysprep -a $DISK --operations machine-id,ssh-hostkeys,logfiles[6]
```

С помощью данной команды мы создаем «золотой» образ, что предотвращает конфликтующие host keys.

4) Производим усадку свободного пространства и сжатие образа

```
qemu-img convert -c -O qcow2 $DISK ~/images/sberlinux/sberlinux-clean.qcow2
qemu-img info ~/images/sberlinux/sberlinux-clean.qcow2
```

4.2.4 Подготовка серверной инфраструктуры

В данном разделе в качестве базового доменного имени облачной платформы используется cloud.itkey.com и имена сервисов по умолчанию. При выполнении инструкции указывайте ваши действительные имена.

1. Подготовьте серверы необходимых типов согласно указанным аппаратным и программным требованиям.
2. Установите на LCM-узле операционную систему согласно программным требованиям.
3. Разместите на LCM-узле публичный ключ для удалённого подключения по SSH.
4. Убедитесь, что на всех узлах время в BIOS настроено в зоне UTC.
5. Убедитесь в наличии пользователя root на всех узлах. Выполняйте все команды от пользователя root.
6. Если ваш LCM-узел работает под управлением ОС SberLinux, установите отображение времени в зоне UTC, выполнив на нем команду:

```
# timedatectl set-local-rtc 0
```

7. Убедитесь, что на всех узлах включена аппаратная виртуализация, выполнив команду:

```
# egrep '(vmx|svm)' /proc/cpuinfo
```

[6] Дополнить параметрами при необходимости

Вывод должен содержать параметр ``vmx`` или ``svm``. Если он отсутствует, включите аппаратную виртуализацию.

8. Получите дистрибутив одним из вариантов поставки.

   Доступный вариант поставки исходного дистрибутива ПО «KeyStack SE»: в отдельном архиве на физическом носителе.

9. Создайте DNS-зону для сервисов KeyStack в вашей службе DNS. В этом примере используется имя cloud.itkey.com. В этой зоне создайте доменные записи следующего вида. Вы можете также использовать другие имена для сервисов, в этом случае учитывайте их в последующих шагах.
   - ks-lcm.cloud.itkey.com — для сервисов GitLab, LCM (управление жизненным циклом);
   - registry.ks-lcm.cloud.itkey.com — для сервиса Gitlab (хранение образов Docker и Podman);
   - rpm-repo.cloud.itkey.com – отдельная область для хранения rpm-пакетов для установки и обновления OpenStack на узлах;
   - vault.cloud.itkey.com — для сервиса Vault (хранение паролей и сертификатов);
   - netbox.cloud.itkey.com — для сервиса NetBox (настройка baremetal-узлов);
   - internal.cloud.itkey.com — для внутреннего интерфейса Kolla;
   - external.cloud.itkey.com — для внешнего интерфейса Kolla.
10. Также рекомендуется создать имена для всех аппаратных узлов инфраструктуры для облегчения дальнейшего доступа и конфигурации. Например:
    - lcm-01.cloud.itkey.com — узел LCM;
    - ctrl-01.cloud.itkey.com — узел Control;
    - comp-01.cloud.itkey.com — узел Compute;
    - comp-02.cloud.itkey.com — узел Compute;
    - net-01.cloud.itkey.com — узел Network;
    - jump-01.cloud.itkey.com — узел Jump.
11. На LCM-узле проверьте сетевую связность и доступность всех узлов:
    1. Настройте доступ до IPMI всех узлов инфраструктуры, MGMT-интерфейсов и VIP-адресов.
    2. Проверьте доступность DHCP-сервера в сети PXE.
    3. Включите поддержку Redfish всех узлов инфраструктуры.
    4. Для всех baremetal-узлов создайте пользователя с административными правами доступа.
12. В вашей DNS-зоне создайте записи для RMI-интерфейсов всех аппаратных узлов. Они будут использованы для baremetal-провиженинга, а также для реализации фенсинга узлов:
    - lcm-01-rmi.cloud.itkey.com — узел LCM,
    - ctrl-01-rmi.cloud.itkey.com — узел Control,
    - comp-01-rmi.cloud.itkey.com — узел Compute,
    - и так далее.
13. Подготовьте данные о вашей инфраструктуре для импорта в NetBox:

    1. Скачайте и откройте файл шаблона netbox2csv.xlsx.
    2. На вкладках **SERVERS** и **Hardware** приведен пример заполнения данных. Ознакомьтесь с форматом заполнения и очистите таблицы от данных.

3. Заполните данные в скачанном файле:

На вкладке **SERVERS**:

- **Tenant** — наименование тенанта;
- **Name** — имя узла;
- **Role** — тип узла;
- **RMI IP/MASK** — IP-адрес IPMI-порта в формате <IP адрес>/<префикс сети>;
- **MGMT IP/MASK** — IP-адрес MGMT-интерфейса в формате <IP адрес>/<префикс сети>;
- **Hardware** — марка и модель узла, выбрать из выпадающего списка;
- **Tags** — теги для группировки узлов.
- На вкладке **Hardware** (если совместимого оборудования не было в списке **Hardware**):
- **Hardware** — название типа сервера;
- **Manufacturer** — производитель сервера;
- **Type** — модель сервера.

4. Убедитесь, что на остальных вкладках заполненного файла появились данные о новых типах серверов.

После выполнения всех вышеописанных шагов ваша серверная инфраструктура готова к установке ПО «KeyStack SE».

4.2.5 Установка и настройка компонентов

Для того чтобы установить и настроить основные компоненты, выполните по порядку следующие шаги:

Шаг 1 — Установите дистрибутив KeyStack на LCM-узел.

Шаг 2 — Подготовьте baremetal-узлы для эксплуатации.

Шаг 3 — Создайте нужное количество регионов.

Шаг 4 — Разверните сетевое окружение региона.

Шаг 5 — Создайте шаблоны образов и типов дисков.

4.2.5.1 Установка KeyStack LCM

Компонент KeyStack LCM (Lifecycle Manager) обеспечивает жизненный цикл облака. Он также используется в качестве seed node — с него происходит установка и настройка всех остальных узлов в составе облака.

Базовая установка LCM

1. Зайдите на LCM-узел по SSH.
2. Скопируйте дистрибутив KeyStack на него любым удобным способом.

3. Переключитесь на пользователя root, перейдите в папку с файлом дистрибутива и распакуйте дистрибутив платформы KeyStack SE.

   SberLinux

   ```
   # sudo su -
   # tar -xvf distr_OP.zip
   # cd installer/
   ```

4. Podman-образы для OpenStack и kolla-ansible будут сохранены в проекте project_k в gitlab.
5. При интеграции KeyStack SE с клиентским сервисом идентификации и аутентификации (Active Directory или другой сервис LDAP (источники каталогов AD/LDAP должны быть реализованы средствами, входящими в состав Platform V SberLinux OS Server (Сертификат ФСТЭК России №4884 от 04.12.2024, действителен до 04.12.2029) с TLS) в папке installer/certs создайте файл ldaps.pem и разместите в нем полную цепочку сертификатов LDAPs.

   Инструкция по интеграции модуля ПО «KeyStack SE» KeyStone с сертифицированным продуктом, поддерживающим стандарт OpenID Connect (OIDC), приведена в пункте 4.3 настоящего документа.

6. В распакованном архиве запустите скрипт installer.sh.
7. Последовательно введите запрашиваемые данные:

   - home dir for the installation: путь к директории для установки, по умолчанию /installer;
   - IP address of this machine: IP-адрес LCM-узла, соответствующий доменному имени ks-lcm;
   - auth LDAP for Gitlab:
     - y — включить поддержку интеграции с каталогами LDAP и ролевой модели в GitLab и NetBox; при выборе этого варианта укажите:
       - LDAP Server URI: адрес LDAP-сервиса в формате ldaps://<IP или FQDN>;
       - LDAP BIND DN: пользователь в LDAP с правами просмотра, используется для подключения к LDAP;
       - LDAP BIND Password: пароль пользователя для подключения к LDAP;
       - LDAP USER SEARCH BASEDN: имя контейнера (Distinguished Name) в LDAP, с которого начинается поиск пользователей;
       - LDAP GROUP SEARCH BASEDN: группа для поиска пользователя; вернет все группы, к которым принадлежит пользователь;
       - LDAP GROUP for reader role: группа LDAP, пользователи из которой соответствуют роли reader;
       - LDAP GROUP for auditor role: группа LDAP, пользователи из которой соответствуют роли auditor;
       - LDAP GROUP for operator role: группа LDAP, пользователи из которой соответствуют роли operator;
       - LDAP GROUP for admin role: группа LDAP, пользователи из которой соответствуют роли admin.

- root domain name: корневое доменное имя сервисов KeyStack, например, cloud.itkey.com, оно будет добавлено к введенным ниже именам сервисов, таким образом, получатся имена вида <сервис>.cloud.itkey.com;
- GitLab domain name: доменное имя сервиса GitLab, например, ks-lcm;
- Vault domain name: доменное имя сервиса Vault, например, vault;
- Netbox domain name: доменное имя сервиса NetBox, например, netbox;
- Generate Self-signed certificates:
  - y — сгенерировать новые самоподписанные сертификаты (вариант по умолчанию);
  - n — использовать существующие сертификаты; при выборе этого варианта разместите в папке installer/certs сертификаты:
    - ca.crt — корневой сертификат (root CA);
    - chain-ca.pem — цепочка CA-сертификатов;
    - ks-lcm.cloud.itkey.com.crt и ks-lcm.cloud.itkey.com.key — сертификат и приватный ключ сертификата для сервиса GitLab;
    - vault.cloud.itkey.com.crt и vault.cloud.itkey.com.key — сертификат и приватный ключ сертификата для сервиса Vault;
    - netbox.cloud.itkey.com.crt и netbox.cloud.itkey.com.key — сертификат и приватный ключ сертификата для сервиса NetBox.

8. Дождитесь завершения операции. На LCM-узле будут установлены и настроены компоненты:
    1. SSH-ключи для беспарольного доступа на узлы;
    2. самоподписанные сертификаты для служб LCM, при их использовании;
    3. система автоматизации (GitLab-CI);
    4. репозитории управления облаком (CI/CD);
    5. NetBox (CMDB).

   Данные с текущей конфигурацией инсталляции сохраняются в Vault в директорию secret_v2/deployments/ks-lcm.cloud.itkey.com/secrets/accounts.

9. При успешной установке будут показаны реквизиты для доступа к компонентам KeyStack. Сохраните реквизиты доступа к компонентам:
    1. GitLab root password
    2. GitLab runner token
    3. GitLab SSH private key
    4. GitLab SSH public key
    5. Netbox admin password
    6. Netbox postgres password
    7. Netbox redis password
    8. Netbox redis cache password
    9. Vault Initial Root Token
    10. Vault Unseal Key 1
    11. Root CA Certificate

   Реквизиты будут показаны только один раз. После закрытия или очищения терминала реквизиты доступа будут безвозвратно удалены.

10. Добавьте корневой сертификат (значение параметра Root CA Certificate) в формате Base64 ASCII в ваши доверенные корневые центры сертификации для корректной работы веб-интерфейсов платформы.

Настройка собственного корневого сертификата

CA (Certificate Authority, Центр сертификации) – Организация или сущность, ответственная за выдачу и управление цифровыми сертификатами, которые подтверждают подлинность публичных ключей в инфраструктуре открытых ключей (PKI).

Инсталлятор генерирует собственные сертификаты платформы с помощью openssl:

1. Создает корневой CA (центр сертификации): корневой сертификат и приватный ключ. Срок действия — 10 лет.
2. Создает wildcard-сертификат и приватный ключ для сервисов GitLab, Vault, NetBox и делает их доверенными этому CA. Также создается цепочка сертификатов вида chain-cert.pem для сервисов. Срок действия — 2 года.
3. Создает в Vault репозиторий installer для хранения сертификатов и создает в нем CA на основе корневого CA. Срок действия — 2 года.
4. Создает роль, с помощью которой можно выписывать сертификаты на основе запросов пользователей.

Все пароли и сертификаты платформы должны храниться в сервисе Vault.

4.2.5.2 Подготовка baremetal-узлов для установки KeyStack

Для установки продукта KeyStack SE на вашу серверную инфраструктуру необходимо выполнить подготовку серверов.

Генерация паролей и сертификатов для Bifrost

Для генерации секретов (паролей и сертификатов), которые будут использованы сервисом Bifrost для провиженинга серверов, запустите пайплайн:

1. Откройте веб-интерфейс развернутого GitLab (ks-lcm.cloud.itkey.com).
2. Авторизуйтесь с помощью реквизитов для GitLab, полученных на этапе установки дистрибутива KeyStack.
3. Откройте проект **project_k / deployments / gen-pwd**.
4. Создайте новый пайплайн: Build > Pipelines > Run Pipeline.
5. В открывшемся окне добавьте переменную OPENSTACK_ENV со значением bifrost и запустите пайплайн.
6. Запустите задачу **create-config** в созданном пайплайне.
7. Дождитесь завершения выполнения задачи.

Установка Bifrost

Установка Bifrost также выполняется через запуск пайплайна GitLab:

1. Перейдите в репозиторий **project_k / deployments / bifrost**.

2. Откройте файл inventory и замените в нем значение LCM_IP на IP-адрес LCM-узла в PXE-сети.
3. Откройте файл globals.d/REGION.yml и замените значение параметра network_interface на имя интерфейса LCM-узла в PXE-сети (например, mgmt).
4. Откройте файл config/bifrost/bifrost.yml и внесите в него изменения:
    1. В параметрах ipa_kernel_url и ipa_ramdisk_url замените LCM_IP на IP-адрес LCM-узла в PXE-сети:

    ipa_kernel_url: "http://LCM_IP:8080/sberlinux-ipa-debug-c.kernel"
    ipa_ramdisk_url: "http://LCM_IP:8080/sberlinux-ipa-debug-c.initramfs"

    2. Сконфигурируйте NTP, для этого раскомментируйте следующую строку и замените в ней 10.224.128.1 на IP-адрес своего NTP-сервера:

    #inspector_extra_kernel_options: "ipa-inspection-collectors=default,logs ipa-ntp-server=10.224.128.1"

    3. Задайте значения параметров DHCP для PXE-сети:

    dhcp_pool_start: 10.37.50.68
    dhcp_pool_end: 10.37.50.78
    dhcp_pool_mask: 255.255.255.224
    dnsmasq_router: 10.37.50.94 # адрес шлюза PXE-сети

5. Создайте и запустите новый пайплайн: Build > Pipelines > Run Pipeline.
6. Дождитесь завершения выполнения шага **setup**.
7. Запустите задачу **deploy-bifrost** на шаге **deploy**.
8. Дождитесь завершения выполнения пайплайна.
9. Запустите сервис dnsmasq. Для этого зайдите на LCM-узел по ssh и выполните команду:

    # podman exec -ti bifrost_deploy bash
    (bifrost-deploy)# systemctl enable dnsmasq
    (bifrost-deploy)# systemctl restart dnsmasq

Настройка правил firewall для Bifrost

Если на узле включен firewall, то добавьте в его конфигурацию исключения для Bifrost:

1. Зайдите на узел LCM по ssh.
2. Откройте TCP-порты 8080, 5050, 6385 и UDP-порты 67, 69:

    SberLinux

    Выполните команды настройки firewalld**:**

    # firewall-cmd --add-port=8080/tcp --permanent
    # firewall-cmd --add-port=5050/tcp --permanent
    # firewall-cmd --add-port=6385/tcp --permanent
    # firewall-cmd --add-service=dhcp --permanent

```
# firewall-cmd --add-service=tftp --permanent
# firewall-cmd –reload
```

Настройка пользователя и пароля IPMI-интерфейсов

Для первоначального провиженинга baremetal-узлов необходимо создать учетную запись пользователя на IPMI-интерфейсах. На всех узлах должен быть задан одинаковый пользователь и пароль. Сконфигурируйте IPMI-интерфейсы всех узлов:

1. Добавьте учетную запись (одинаковый пользователь и пароль на всех узлах).
2. Установите ему права доступа на:
   - однократную загрузку по PXE,
   - включение и выключение сервера.

PXE *(*Preboot Execution Environment) – Стандарт для загрузки компьютера с помощью сетевой карты без использования локальных носителей данных (жёсткого диска, USB-накопителя и т. п.).

При провиженинге узлов Bifrost будет брать имя пользователя и пароль из сервиса Vault. Добавьте их в Vault:

1. Перейдите в веб-интерфейс Vault (vault.cloud.itkey.com).
2. Авторизуйтесь с помощью реквизитов для Vault, полученных на этапе установки дистрибутива KeyStack.
3. Перейдите в директорию с настройками Bifrost: secret_v2 / deployments / ks-lcm.cloud.itkey.com / bifrost / rmi.
4. Нажмите Create new для создания новой версии секрета.
5. Задайте параметры user и password интерфейса IPMI-узлов и сохраните новую версию.

Настройка перечня оборудования в NetBox

1. Откройте файл с данными об оборудовании netbox2csv.xlsx, заполненный на этапе подготовки серверной инфраструктуры.
2. Откройте веб-интерфейс развернутого NetBox (netbox.cloud.itkey.com).
3. Авторизуйтесь с помощью реквизитов для NetBox, полученных на этапе установки дистрибутива KeyStack.
4. Откройте вкладку SERVERS в Excel. По одному добавьте в NetBox все уникальные значения тегов из столбца Tags, выполнив следующие шаги:
   1. В разделе NetBox Customization > Tags нажмите Add.
   2. Введите название тега, его slug (используется в URL) и цвет.
5. Импортируйте данные о тенантах из Excel в NetBox:
   1. Скопируйте данные с вкладки Excel netbox_tenant.csv.
   2. В разделе NetBox Organization > Tenants нажмите Import.
   3. Вставьте скопированные данные в поле Data.
6. Аналогичным образом импортируйте прочие данные из Excel в NetBox:
   1. с вкладки netbox_site.csv — в раздел Organization > Sites;
   2. с вкладки netbox_device.csv — в раздел Devices > Devices;
   3. с вкладки netbox_interface.csv — в раздел Devices > Interfaces;
   4. с вкладки netbox_ip.csv — в раздел IPAM > IP Addresses.
7. Создайте виртуальные локальные сети:

1. В разделе NetBox IPAM > VLANs нажмите Add.
2. Укажите соответствующие значения VLAN ID, Name, Tenant и Site.
3. В разделе Prefixes нажмите Add Prefix и укажите соответствующее сети VLAN значение IP-префикса.

8. Проверьте соответствие префиксов и адресов шлюзов:
   1. Перейдите в раздел IPAM > Prefixes.
   2. Убедитесь, что значения Prefix совпадает с указанными default_gateway. Если это не так, измените их и сохраните изменения.
9. Настройте конфигурационные контексты для каждой роли узла (control, compute и так далее). Для этого выполните следующие действия:
   1. Подготовьте описание конфигурационных контекстов для каждой роли узла в формате JSON:

Пример для Control

```
{
  "bonds": [
    {
      "Interface": "bond0",
      "MTU": 9000,
      "Slaves": [
      "ens1f0np0",
      "ens2f0np0"
      ],
      "Tagged_vlan": [
      "39",
      "484"
      ]
    },
    {
      "Interface": "bond1",
      "MTU": 9000,
      "Slaves": [
      "ens2f1np1",
      "ens1f1np1"
      ],
      "Tagged_vlan": []
    },
    {
      "Interface": "bond2",
      "MTU": 9000,
      "Slaves": [
      "eno1",
      "eno2",
      "eno3",
      "eno4"
      ],
      "Tagged_vlan": []
    }
  ],
  "default_gateway": "10.30.55.60/20",
```

```
    "dns_name": "lab.cloud.cfg_fake.ru",
    "dns_server": [
      "11.55.70.153",
      "11.25.44.11"
    ],
    "route_vxlan": [
      {
        "gw": "10.10.10.150/10",
        "route": "10.10.10.0/24"
      },
      {
        "gw": "10.10.10.150/10",
        "route": "10.10.20.0/24"
      }
    ],
    "vlans": [
      {
        "Interface": "mgmt",
        "MTU": "9000",
        "Parent": "bond0",
        "id": "39"
      },
      {
        "Interface": "vxlan",
        "MTU": "9000",
        "Parent": "bond0",
        "id": "484"
      }
    ]
}
```


Здесь:

- default_gateway — адрес и маска шлюза для подсети по умолчанию;
- dns_name — имя DNS-сервера;
- dns_server — IP-адрес DNS-сервера;
- Interface — название интерфейса;
- MTU — значение MTU;
- Parent — название родительского интерфейса, если предусмотрена иерархия;
- "id": "40" — MGMT VLAN на гипервизорах;
- "id": "39" — MGMT VLAN на Controller-узлах;
- "id": "484" — VLAN сети управления IPMI.

2. Перейдите в раздел NetBox Provisioning > Config Contexts.
3. Нажмите Add.
4. Заполните поля Name и Weight.
5. Вставьте конфигурацию в поле Data.
6. В поле Tags укажите соответствующие конфигурационному контексту теги.
7. Повторите настройку конфигурационного контекста для узлов каждой роли.

10. Завершите настройку NetBox:
    1. В NetBox в разделе API tokens, скопируйте токен.

2. На LCM-узле выполните скрипт из XLSX-файла в CLI, вкладка update_ctx.sh, предварительно заменив значения:
    - NETBOX_TOKEN — токен NetBox из раздела API tokens;
    - NETBOX_URI — адрес NetBox в формате https://netbox.cloud.itkey.com/.

11. Активируйте все BMC-узлы инфраструктуры:
    1. В веб-интерфейсе NetBox перейдите в раздел Devices > Devices.
    2. Отметьте все необходимые узлы.
    3. Нажмите Edit Selected.
    4. В открывшейся форме для параметра state в разделе Custom Fields установите значение ready.
    5. Сохраните изменения.

Запуск пайплайнов развертывания

Для корректной работы протокола PXE на портах коммутаторов сетевой инфраструктуры должна быть включена функция LACP Fallback.

Запустите пайплайн для развертывания системы. Данную операцию необходимо произвести для узлов каждого типа.

1. Откройте веб-интерфейс GitLab.
2. Откройте проект project_k / services / baremetal.
3. Создайте новый пайплайн: Build > Pipelines > Run Pipeline.
4. В открывшемся окне добавьте переменные:
    - TARGET_ROLE — тип узла, определенный в NetBox:
        - controller — для Controller-узлов,
        - network — для Network-узлов,
        - compute — для Compute-узлов.
    - TARGET_CLOUD — имя тега региона в NetBox;
    - TARGET_NODE — имя конкретного узла для развертывания, можно перечислить несколько имён узлов через пробел;
    - IRONIC_ENV — окружение для развертывания узлов (по умолчанию BIFROST);
    - IRONIC_SSH_KEY — собственные SSH-ключи (формат — один ключ на строку);
    - IRONIC_IMAGE_URL — путь до образа системы, который будет установлен на узел, в формате http://<IP адрес LCM узла>:8080/sberlinux-9.4-x86_64.qcow2. Требования к образу:
        - только поддерживаемый тип операционной системы;
        - формат — QCOW2;
        - hash-файл образа в формате <имя образа>.md5 располагается в одной папке с самим образом.
    - IRONIC_IMAGE_ROOTFS_UUID — UUID корневого раздела в образе, если используется software RAID;
    - IPA_KERNEL_NAME — путь до образа агента Ironic Python Agent (IPA) kernel image;
    - IPA_RAMDISK_NAME — путь до образа агента Ironic Python Agent (IPA) initramfs image;

- KOLLA_ANSIBLE_IMG_TAG — тег контейнера с kolla-ansible, используемый для пайплайна;
- EXPERIMENTAL_NETBOX_INTROSPECTION: true — использовать автозаполнение интерфейсов устройств в NetBox;
- KEYSTACK_RELEASE — тег релиза Keystack;
- CI_DEBUG_TRACE:
  - true — выводить отладочную информацию в пайплайне.

5. Запустите пайплайн, нажав кнопку Run pipeline.
6. Дождитесь завершения выполнения операции.
7. Повторите шаги запуска пайплайна для остальных типов узлов.
8. Перейдите в раздел Devices > Devices в NetBox и убедитесь, что все нужные узлы поменяли значение **state** на production.

После выполнения описанной выше подготовки серверной инфраструктуры можно переходить к развёртыванию регионов.

4.2.5.3 Создание региона

Регионы используются для логического разделения инфраструктуры и ресурсов платформы KeyStack SE. Регион представляет собой изолированную группу узлов, предоставляющих вычислительные, сетевые ресурсы и ресурсы хранения. Каждый регион разворачивается на своем, отдельном наборе узлов. Регионы управляются и настраиваются независимо друг от друга.

Чтобы создать несколько регионов, повторите инструкцию из этого раздела нужное количество раз.

Создание и подготовка репозитория нового региона

1. Откройте веб-интерфейс развёрнутого GitLab по адресу https://ks-lcm.cloud.itkey.com.
2. Создайте форк GitLab-репозитория region1. Это репозиторий-шаблон, который используется для создания новых регионов. Для этого выполните действия:
   1. Перейдите в проект project_k / deployments / region1.
   2. Нажмите кнопку Fork.
   3. В открывшемся окне укажите параметры:
      - Project name — имя региона.

        Имя региона должно удовлетворять условиям:

        - содержать латинские буквы [a-z] и цифры [0–9];
        - дефисы не могут быть в начале и конце имени;
        - не должно содержать два дефиса одновременно;
        - не должно содержать пробелы и специальные символы (например, !, $, &, _ и т. д.);
        - минимальная длина — 3, максимальная — 63 символа;
        - нечувствительно к регистру.
      - Project slug — укажите точно такой же как имя региона.
      - Select a namespace — выберите namespace для проекта, например, project_k/deployments.
   4. Нажмите кнопку Fork project.

3. Перейдите в созданный репозиторий нового региона.
4. Удалите связь с родительским репозиторием:
    1. Перейдите в раздел Settings > General > Advanced.
    2. Нажмите кнопку Remove fork relationship.
    3. Введите название региона для подтверждения действия.
5. Разрешите использование job-токена из всех проектов GitLab:
    1. Перейдите в раздел Settings > CI/CD > Job token permissions.
    2. Включите опцию All groups and projects.
6. Выполнение некоторых пайплайнов развёртывания может занимать продолжительное время. Увеличьте таймаут пайплайнов по умолчанию:
    1. Перейдите в раздел Settings > CI/CD > General pipelines.
    2. Установите значение Timeout — 5h.
7. Настройте параметры инфраструктуры созданного региона:
    1. Откройте файл inventory.
    2. Заполните информацию об узлах инфраструктуры [control], [network], [compute] в формате <имя узла> ansible_host=<IP адрес узла>.
    3. Откройте файл globals.d/REGION.yml и укажите значения параметров:
        - kolla_internal_vip_address — виртуальный IP-адрес для внутреннего VIP-интерфейса (используется для балансировки запросов);
        - kolla_internal_fqdn — DNS-имя для внутреннего VIP-адреса, например internal.cloud.itkey.com;
        - kolla_external_vip_address — виртуальный IP-адрес для внешнего VIP-интерфейса (используется для балансировки запросов);
        - kolla_external_fqdn — DNS-имя для внешнего VIP-адреса, например external.cloud.itkey.com;
        - network_interface — имя интерфейса для MGMT-сети, на котором будет разворачиваться OpenStack;
        - neutron_external_interface — имя интерфейса для связи с внешней инфраструктурой, например, сетями провайдеров, маршрутизаторами и плавающими IP-адресами.
    4. Если вы используете сервис балансировки нагрузки Octavia, откройте файл globals.d/octavia.yml и укажите значения параметра octavia_amp_network_cidr — IP-префикс подсети управления сервиса Octavia.

Генерация паролей и сертификатов для региона

1. Запустите пайплайн на генерацию паролей и сертификатов для региона:
    1. Откройте проект **project_k / deployments / gen-pwd**.
    2. Создайте новый пайплайн: Build > Pipelines > Run Pipeline.
    3. В открывшемся окне добавьте переменную OPENSTACK_ENV, в качестве значения укажите имя созданного региона.
    4. Запустите пайплайн Run pipeline.
    5. Запустите задачу **create-config** и дождитесь её завершения.
2. Повторите запуск пайплайна нужное количество раз, если создано несколько регионов.

Настройка паролей внешних сервисов

1. Для реализации автоэвакуации ВМ при сбое гипервизора, задайте пароль интерфейса IPMI baremetal-узлов:
    1. Перейдите в веб-интерфейс развёрнутого Vault.
    2. Перейдите в директорию с настройками региона secret_v2 / deployments / ks-lcm.cloud.itkey.com / <имя региона> / passwords_yml**.**
    3. Нажмите Create new version.
    4. Найдите и замените значение параметра bmc_password на пароль от интерфейса IPMI BMC-узлов.
2. По аналогии с предыдущим шагом в директории с настройками региона secret_v2 / deployments / ks-lcm.cloud.itkey.com / <имя региона> / passwords_yml укажите пароли от своих сервисов в соответствующих полях, например:
    - ldap_password — LDAP,
    - smtp_password — SMTP,
    - cinder_dorado_user_password — СХД.

Конфигурация BGP

В базовой конфигурации платформы механизм мониторинга состояния и сетевой доступности узлов реализуется сервисом keepalived на основе протокола VRRP. Альтернативно, для этих целей возможно использовать протокол BGP, который также поддерживается платформой.

Предварительные требования

- Поддержка протокола BGP со стороны опорной сетевой инфраструктуры (underlay) заказчика.
- Наличие выделенного для региона номера автономной системы. Это может быть частный номер автономной системы (private AS number).
- Выделены новые IP-адреса для имён, указанных в файле globals.d/REGION.yml в переменных kolla_internal_fqdn и kolla_external_fqdn, которые будут использоваться для маршрутизации с помощью BGP.
- Известны IP-адреса соседей BGP и пароль для BGP-аутентификации.

При переходе инфраструктуры на использование протокола BGP будет меняться сетевая конфигурация. В связи с этим также поменяются внешний и внутренний VIP-адреса региона. Запланируйте обновление DNS-записей для внутреннего и внешнего VIP-адресов облака. Смену значений доменных имен необходимо будет произвести после подготовки новой конфигурации.

Для поддержки протокола BGP в KeyStack SE используется сервис Bird. Для конфигурации сервиса Bird выполните перечисленные шаги:

1. Добавьте пароль для подключения к автономной системе в хранилище секретов:
    1. Подготовьте пароль для подключения к автономной системе через сервис Bird.
    2. Откройте веб-интерфейс сервиса Vault.
    3. Перейдите в директорию secret_v2 / deployments / <FQDN сервиса GitLab> / <имя региона> / passwords_yml**.**
    4. В секции Current version нажмите Create new.

5. Добавьте в JSON-файл параметр bird_external_bgp_password со значением вашего пароля на подключение к автономной системе через сервис Bird.

2. Внесите изменения в конфигурацию региона:
    1. Откройте веб-интерфейс GitLab.
    2. Перейдите в репозиторий региона project_k > deployments > <имя региона>.
    3. Откройте файл globals.d/REGION.yml и обновите значения переменных kolla_external_vip_address и kolla_internal_vip_address на IP-адреса, соответствующие новой конфигурации, использующей BGP.
3. Добавьте конфигурацию BGP в конфигурацию региона:
    1. Откройте веб-интерфейс GitLab.
    2. Перейдите в репозиторий региона project_k > deployments > <имя региона>.
    3. Создайте файл globals.d/bgp.yml со следующей конфигурацией:

```
##bgp options
enable_keepalived: "no"
enable_bird: "yes"
enable_exabgp: "yes"
internal_BGP: "65001"
external_BGP: "65002"
external_BGP_IP:
 - 10.224.57.4
 - 10.224.56.4
external_BGP_password: "{{ bird_external_bgp_password }}"
announce_networks:
 - "{{ kolla_external_vip_address }}/32"
 - "{{ kolla_internal_vip_address }}/32"
```

В этом примере:

- enable_keepalived: "no" — выключение keepalived, поскольку он не будет использоваться при включении BGP;
- enable_bird: "yes" — включение сервиса Bird;
- enable_exabgp: "yes" — включение сервиса exabgp;
- announce_networks — указываются VIP-адреса для сервисов (internal_vip, external_vip), с указанием длины префикса, равной /32;
- external_BGP_IP — список IP-адресов eBGP-соседей;
- internal_BGP: "65452" — номер внутренней автономной системы;
- external_BGP: "65453" — номер внешней автономной системы.

Отключение сервисов DRS и HA

Во избежание нежелательного срабатывания DRS и HA во время применения новой конфигурации, временно выключите их.

Для выключения DRS:

1. Зайдите в веб-интерфейс Портала администратора.
2. Перейдите в раздел DRS > Jobs.
3. Выключите все задачи DRS.

Для выключения HA:

1. Зайдите на каждый Control-узел по SSH и выполните команду:
2. # systemctl stop kolla-consul-container.service

Обновление записей DNS

Выполните запланированные изменения в DNS:

1. Обновите адрес записи внутреннего VIP-интерфейса региона.
2. Обновите адрес записи внешнего VIP-интерфейса региона.
3. Дождитесь обновления данных на DNS-сервере, используемом узлами региона. При обращении на доменные имена, указанные в файле globals.d/REGION.yml в переменных kolla_internal_fqdn и kolla_external_fqdn, должны использоваться новые IP-адреса. Обновление DNS-записей может занимать продолжительное время. Приступать к дальнейшим работам можно только после обновления DNS-кешей..

Развёртывание изменений на регионе

1. Ранее используемый в облаке сервис keepalived необходимо остановить и удалить на всех Controller-узлах. Для этого выполните команды:

```
# systemctl stop kolla-…keepalived
# systemctl disable kolla-…keepalived
# podman rm –force keepalived
```

2. Откройте веб-интерфейс GitLab.
3. Перейдите в репозиторий региона project_k > deployments > <имя региона>.
4. Создайте новый пайплайн: Build > Pipelines > Run Pipeline.
5. В открывшемся окне добавьте значение переменной KOLLA_ARGS равное -t bird,exabgp,haproxy.
6. Запустите пайплайн, нажав кнопку Run pipeline.
7. Дождитесь завершения задач на этапе **setup**.
8. Запустите задачу **deploy** на этапе **deploy** и дождитесь её завершения.

Проверка конфигурации BGP

Для проверки корректности работы BGP, выполните следующие действия на каждом Control-узле:

1. Выполните команду:

```
podman exec -ti bird birdc show protocols -s /var/lib/kolla/bird/bird.sock
```

Убедитесь, что протокол BGP запущен и находится в состоянии Established.

2. Выполните команду:

```
podman exec -ti bird birdc show route -s /var/lib/kolla/bird/bird.sock
```

Убедитесь, что нужные маршруты были добавлены в таблицу маршрутизации протоколом BGP.

3. Выполните команду:

   podman exec -ti bird birdc show bfd sessions -s /var/lib/kolla/bird/bird.sock

Убедитесь, что все сконфигурированные BFD-сессии находятся в ожидаемом состоянии.

Включение сервисов DRS и HA

Для включения DRS:

1. Зайдите в веб-интерфейс Портала администратора.
2. Перейдите в раздел DRS > Jobs.
3. Включите задачи DRS, которые были выключены перед применением новой конфигурации.

Для включения HA:

1. Зайдите на каждый Control-узел по SSH и выполните команду:

systemctl start kolla-consul-container.service

Дополнительная настройка узлов перед развёртыванием

Выполните дополнительную конфигурацию узлов перед развёртыванием региона. Для этого используйте конфигурационный файл host_config/host-config.yml. Следуйте инструкциям по редактированию файла host-config.yml в разделе Использование host-config для настройки региона, не запуская пайплайн. Все выполненные вами конфигурации будут применены ниже при запуске задачи bootstrap-servers пайплайна развёртывания региона.

Использование host-config для настройки региона

host-config может использоваться для внесения специфических настроек в конфигурацию ОС узлов. Это можно делать как перед развёртыванием региона, так и после. Если вы выполняете конфигурацию перед развёртыванием региона, вы можете не запускать пайплайн после внесения изменений в host-config.yml, они будут применены при последующем запуске основного пайплайна по развёртыванию региона.

Файл host-config находится в репозитории региона по пути host_config/host-config.yml. Если он отсутствует, вы можете создать его самостоятельно.

Настройка NTP в регионе

Для выполнения настройки протокола синхронизации времени NTP на узлах региона выполните следующие действия:

1. В файл host_config/host-config.yml в репозитории региона добавьте строки с указанием временной зоны и IP-адреса NTP-сервера:

   ntp_enabled: true

```
ntp_timezone: Europe/Moscow
ntp_servers:
  - 10.224.38.254
```

2. Создайте и запустите новый пайплайн: Build > Pipelines > Run Pipeline.
3. Дождитесь завершения выполнения задач на шаге **setup**.
4. Запустите задачу **bootstrap-servers**. Дождитесь завершения задачи **host-config**.

Для проверки настройки NTP:

1. Зайдите на любой узел региона по протоколу SSH.
2. Выполните команду timedatectl:

```
$ timedatectl
               Local time: Tue 2024-09-17 10:36:10 MSK
           Universal time: Tue 2024-09-17 07:36:10 UTC
                 RTC time: Tue 2024-09-17 07:36:10
                Time zone: Europe/Moscow (MSK, +0300)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
```

Убедитесь, что включен параметр System clock synchronized, отображаются правильные время и зона.

Настройка DNS в регионе

DNS (Domain Name System) – Система, которая переводит имена доменов, удобные для людей, в IP-адреса, которые используются компьютерами для идентификации узлов в сети.

Для выполнения настройки DNS на узлах региона выполните следующие действия:

1. В файл host_config/host-config.yml в репозитории региона добавьте строки конфигурации DNS:

```
dns_enabled: true
dns_servers:
- 10.220.32.42
- 10.220.32.43
dns_dnssec: false
dns_domains: ["cloud.itkey.com"]
```

2. Создайте и запустите новый пайплайн: Build > Pipelines > Run Pipeline.
3. Дождитесь завершения выполнения задач на шаге **setup**.
4. Запустите задачу **bootstrap-servers**. Дождитесь завершения задачи **host-config**.

Для проверки настройки DNS:

1. Зайдите на любой узел региона по протоколу SSH.
2. Проверьте конфигурацию DNS на узле, выполнив команду cat /etc/resolv.conf:

```
$ cat /etc/resolv.conf
```

```
nameserver 10.220.32.42
nameserver 10.220.32.43
search cloud.itkey.com
```

Отключение истории команд

Отключение истории команд позволяет исключить хранение учетных записей и паролей в истории команд пользователей в открытом виде. Для отключения истории команд выполните следующие действия:

1. В файл host_config/host-config.yml в репозитории региона добавьте строки конфигурации:

```
security_hardening: true
disable_history: true
```

2. Создайте и запустите новый пайплайн: Build > Pipelines > Run Pipeline.
3. Дождитесь завершения выполнения задач на шаге **setup**.
4. Запустите задачу **bootstrap-servers**. Дождитесь завершения задачи **host-config**.

Для проверки отключения истории команд:

1. Зайдите на любой узел региона по протоколу SSH.
2. Введите команду history и убедитесь, что вывод отсутствует.

Отключение внешних накопителей

Отключение поддержки внешних накопителей позволяет запретить использование внешних USB-накопителей. Для настройки выполните следующие действия:

1. В файл host_config/host-config.yml в репозитории региона добавьте строки конфигурации:

```
security_hardening: true
disable_usb_storage: true
```

2. Создайте и запустите новый пайплайн: Build > Pipelines > Run Pipeline.
3. Дождитесь завершения выполнения задач на шаге **setup**.
4. Запустите задачу **bootstrap-servers**. Дождитесь завершения задачи **host-config**.

Для проверки отключения внешних накопителей:

1. Зайдите на любой узел региона по протоколу SSH.
2. Введите команду lsmod | grep usb_storage, чтобы убедиться, что в выводе отсутствует значение usb_storage.

Удаление группы wheel из файла /etc/sudoers

Для удаления группы wheel из списка привилегированных пользователей выполните следующие действия:

1. В файл host_config/host-config.yml в репозитории региона добавьте строки конфигурации:

```
security_hardening: true
remove_wheel_from_sudoers: true
```

2. Создайте и запустите новый пайплайн: Build > Pipelines > Run Pipeline.
3. Дождитесь завершения выполнения задач на шаге **setup**.
4. Запустите задачу **bootstrap-servers**. Дождитесь завершения задачи **host-config**.

Для проверки отключения внешних накопителей:

1. Зайдите на любой узел региона по протоколу SSH.
2. Введите команду cat /etc/sudoers и убедитесь, что строка # %wheel ALL=(ALL) ALL закомментирована (начинается с #).

Установка привилегий доступа к файлам

Данная настройка поддерживается только для Sberlinux.

Для установки привилегий доступа к файлам выполните следующие действия:

1. В файл host_config/host-config.yml в репозитории региона добавьте строки конфигурации:

```
security_hardening: true
setting_file_permissions: true
```

2. Создайте и запустите новый пайплайн: Build > Pipelines > Run Pipeline.
3. Дождитесь завершения выполнения задач на шаге **setup**.
4. Запустите задачу **bootstrap-servers**. Дождитесь завершения задачи **host-config**.

Настройка текстового редактора

Для настройки текстового редактора выполните следующие действия:

1. В файл host_config/host-config.yml в репозитории региона добавьте строки конфигурации:

```
security_hardening: true
copy_pam_env: true
```

2. Создайте и запустите новый пайплайн: Build > Pipelines > Run Pipeline.
3. Дождитесь завершения выполнения задач на шаге **setup**.
4. Запустите задачу **bootstrap-servers**. Дождитесь завершения задачи **host-config**.

Настройка соответствующих модулей ядра

Для настройки соответствующих модулей ядра выполните следующие действия:

1. В файл host_config/host-config.yml в репозитории региона добавьте строку конфигурации:
2. sysctl_hardening_enabled: true
3. Создайте и запустите новый пайплайн: Build > Pipelines > Run Pipeline.
4. Дождитесь завершения выполнения задач на шаге **setup**.
5. Запустите задачу **bootstrap-servers**. Дождитесь завершения задачи **host-config**.

Настройка GRUB

Для настройки GRUB выполните следующие действия:

1. В файл host_config/host-config.yml в репозитории региона добавьте строки конфигурации:

```
grub_config_enabled: true
grub_cmdline_options:
  - "processor.max_cstate=1" # cpu performance option
  - "intel_idle.max_cstate=1" # cpu performance option
  - "idle=poll" # cpu performance option
  - "hugepagesz=1G" # hugepages option
  - "audit=1" # paranoic security option
  - "ipv6.disable=1" # paranoic security option
  - "tsx=auto" # paranoic security option
  - "slab_nomerge" # paranoic security option
  - "randomize_kstack_offset=1" # paranoic security option
  # - "custom_option"
```

2. Создайте и запустите новый пайплайн: Build > Pipelines > Run Pipeline.
3. Дождитесь завершения выполнения задач на шаге **setup**.
4. Запустите задачу **bootstrap-servers**. Дождитесь завершения задачи **host-config**.

Настройка аудита на серверах Linux -подобных систем

Для настройки аудита выполните следующие действия:

1. В файл host_config/host-config.yml в репозитории региона добавьте строку конфигурации:
2. auditd_enabled: true
3. Создайте и запустите новый пайплайн: Build > Pipelines > Run Pipeline.
4. Дождитесь завершения выполнения задач на шаге **setup**.
5. Запустите задачу **bootstrap-servers**. Дождитесь завершения задачи **host-config**.

Настройка завершения сеансов подключения к серверам

Для настройки завершения сеансов подключения к серверам выполните следующие действия:

1. В файл host_config/host-config.yml в репозитории региона добавьте строку конфигурации:

```
sshd_enabled: true
client_alive_interval: 15
client_alive_count_max: 20 #5 min
channel_timeout: 60 #1 min
```

2. Создайте и запустите новый пайплайн: Build > Pipelines > Run Pipeline.
3. Дождитесь завершения выполнения задач на шаге **setup**.
4. Запустите задачу **bootstrap-servers**. Дождитесь завершения задачи **host-config**.

Запуск пайплайна развёртывания региона

1. Запустите пайплайн по развёртыванию региона:
    1. Откройте веб-интерфейс развёрнутого GitLab.
    2. Откройте проект **project_k / deployments / <имя региона>**.
    3. Создайте новый пайплайн: Build > Pipelines > New Pipeline.
    4. В открывшемся окне при необходимости укажите значения переменных:
        - KOLLA_ARGS — дополнительные теги:
            - --limit <имя узла> — выполнить пайплайн для отдельного узла платформы;
            - --tags <имя компонента> — выполнить пайплайн для отдельного компонента платформы.
        - KOLLA_ANSIBLE_DEPLOY_ACTION — список задач для kolla-ansible, должно быть указано deploy;
        - KEYSTACK_RELEASE — версия KeyStack для развёртывания.
    5. Запустите пайплайн Run pipeline.
    6. Дождитесь завершения задачи **setup**.
    7. Запустите задачу **bootstrap-servers** и дождитесь её завершения. Если после её завершения автоматически запустятся связанные задачи, дождитесь их завершения.
    8. Запустите задачу **deploy** и дождитесь её завершения.

Общий список задач в пайплайне:

- **setup** — подготовка паролей и настроек для развёртывания платформы;
- **setup-castellan** — синхронизация общего хранилища паролей с хранилищем, уникальным для каждого компонента продукта;
- **backup-db** — создание резервной копии базы данных MariaDB;
- **bootstrap-servers** — установка необходимых пакетов и настройка компонентов ОС;
- **deploy** — развёртывание и запуск сервисов KeyStack на узлах;
- **states** — применение конфигурации к узлам платформы;
- **rally** — валидация компонентов платформы;
- **tempest** — тестирование платформы на соответствие конфигурации.

По завершению работы пайплайна регион будет создан. Вы можете переходить к настройке сетевого окружения региона.

4.2.5.4 Развёртывание сети региона

После создания региона необходимо развернуть и сконфигурировать сеть региона. Это необходимо для дальнейшей настройки виртуального сетевого окружения: сети, подсети, группы безопасности и прочее.

Включение поддержки тегирования

Для включения поддержки тегирования VLAN выполните действия:

1. Откройте веб-интерфейс развёрнутого GitLab.
2. Перейдите в репозиторий созданного региона.
3. Найдите файл globals.d/REGION.yml и добавьте в него строки. В параметре global_physnet_mtu укажите допустимое вашей сетевой инфраструктурой значение MTU:

```
enable_neutron_provider_networks: "yes"
global_physnet_mtu: "9000"
```

4. Создайте файл config/neutron/neutron.conf с содержимым:

```
[DEFAULT]
global_physnet_mtu = {{ global_physnet_mtu }}
```

5. Создайте файл config/neutron/ml2_conf.ini со следующим содержимым. В параметре network_vlan_ranges укажите допустимый вашей сетевой инфраструктурой диапазон номеров VLAN:

```
[ml2_type_vlan]
network_vlan_ranges = physnet1:1000:2000
```

6. Создайте новый пайплайн: Build > Pipelines > Run Pipeline.
7. В переменной KOLLA_ARGS укажите значение -t neutron. Это обеспечит выполнение пайплайна только для сервиса Neutron.
8. Запустите пайплайн, нажав кнопку Run pipeline.
9. Дождитесь завершения задач на этапе **setup**.
10. Запустите задачу **deploy** на этапе **deploy** и дождитесь её завершения.

Создание сети для региона

Создайте административную провайдерскую сеть для вашего региона:

1. Перейдите в веб-интерфейс портала самообслуживания. Подробнее в разделе Подключение к интерфейсу управления портала самообслуживания.
2. Перейдите в раздел Admin > Network > Networks.
3. Нажмите Create Network.
4. При создании сети укажите параметры:
    1. **Provider Network Type**: VLAN;
    2. **Physical Network**: physnet1;
    3. **Segmentation ID**: ID VLAN в соответствии с физической сетевой инфраструктурой.

После выполнения этих действий вы можете создавать виртуальные сети и подсети для тенантов облака.

4.2.6 Подключение и настройка СХД

Платформа KeyStack SE поддерживает подключение дополнительных систем хранения данных следующих типов:

- iSCSI,
- FC,
- NFS.

Также существует возможность включить LVM для проверки работоспособности систем. При подключении СХД необходимо использовать СХД, имеющие сертификат соответствия ФСТЭК России и руководствоваться эксплуатационной документацией на одключаемую СХД. Функция безопасности по удалению объектов файловой системы, используемых ОО, путем перезаписи уничтожаемых (стираемых) объектов файловой системы случайной битовой последовательностью должно выполняться ОО только при подключении систем хранения по типу iSCSI.

Включение поддержки LVM

По умолчанию поддержка LVM выключена. LVM использует локальные диски Control-узлов для хранения. Эта опция может использоваться только для тестирования систем и не рекомендуется для производственной эксплуатации.

Не используйте LVM для производственной эксплуатации.

1. Откройте веб-интерфейс развернутого GitLab.
2. Откройте проект **project_k / deployments / <имя региона>**.
3. Добавьте строки в файл globals.d/REGION.yml:
4. enable_cinder_backend_lvm: "yes"
5. Создайте новый пайплайн: Build > Pipelines > Run Pipeline.
6. В открывшемся окне добавьте параметры:
    - KOLLA_ANSIBLE_DEPLOY_ACTION — deploy;
    - KOLLA_ARGS — -t cinder,iscsi,nova-cell.
7. Запустите пайплайн: Run pipeline.
8. Дождитесь завершения задач на этапе **setup**.
9. Запустите задачу **deploy** на этапе **deploy**.
10. Дождитесь завершения выполнения операции.

Включение поддержки iSCSI

Для включения поддержки протокола iSCSI выполните следующие действия:

1. Откройте веб-интерфейс развернутого GitLab.
2. Откройте проект **project_k / deployments / <имя региона>**.
3. Добавьте строки в файл globals.d/REGION.yml:
4. enable_cinder_backend_iscsi: "yes"
5. enable_cinder_backend_lvm: "no"
6. Создайте новый пайплайн: Build > Pipelines > Run Pipeline.

7. В открывшемся окне добавьте параметры:
    - KOLLA_ANSIBLE_DEPLOY_ACTION — deploy;
    - KOLLA_ARGS — -t cinder.
8. Запустите пайплайн: Run pipeline.
9. Дождитесь завершения задач на этапе **setup**.
10. Запустите задачу **deploy** на этапе **deploy**.
11. Дождитесь завершения выполнения операции.

Включение и конфигурация Fibre Channel

В разделе приведен пример подключения Huawei Dorado с поддержкой FC. Шаги по подключению систем хранения других вендоров могут отличаться.

Настройте Cinder:

1. Создайте новый проект internal_cinder и пользователя internal_cinder_user.
2. Назначьте пользователю internal_cinder_user права member в созданном проекте.
3. Сохраните идентификаторы проекта и пользователя. Далее для примера будут использоваться abc11111111111111111111111111111 и def22222222222222222222222222222 для проекта и пользователя соответственно.
4. (Опционально) Увеличьте квоты в проекте internal_cinder для дисков.

Создайте и примените конфигурацию Fibre Channel:

1. Откройте веб-интерфейс развернутого GitLab.
2. Откройте проект **project_k / deployments / <имя региона>**.
3. Добавьте строки в файл globals.d/REGION.yml:

```
enable_multipathd: "yes"
enable_cinder_backend_lvm: "no"
default_volume_type: "huawei_storage"
cinder_internal_tenant_project_id: "abc11111111111111111111111111111"
cinder_internal_tenant_user_id: "def22222222222222222222222222222"
```

4. Создайте файл config/cinder/cinder-volume/vendor-configs/dorado.xml с содержимым:

```
SberLinux

<?xml version='1.0' encoding='UTF-8'?>
<config>
  <Storage>
    <Product>Dorado</Product>
    <Protocol>FC</Protocol>
    <UserName>user</UserName>
    <UserPassword>{{ lookup('hashi_vault', 'secret={{ vault_engine }}/data/{{ vault_prefix }}/{{ OPENSTACK_ENV | lower }}/huawei')['data']['UserPassword'] }}</UserPassword>
```

```
        <RestURL>https://<IP-
адрес>:8088/deviceManager/rest/;https://IP_ADDRESS2:8088/deviceManager/rest/</Re
stURL>
      </Storage>
      <LUN>
        <LUNCopySpeed>3</LUNCopySpeed>
        <StoragePool>NAME_POOL</StoragePool>
        <LUNType>Thin</LUNType>
      </LUN>
      <FC>
        <Initiator HostName=".*" ALUA="1" FAILOVERMODE="3"
SPECIALMODETYPE="0" PATHTYPE="0" />
      </FC>
    </config>
```

5. Создайте файл config/cinder.conf с содержимым:

```
[DEFAULT]
default_volume_type = {{ default_volume_type }}
enabled_backends = huawei_storage
enforce_multipath_for_image_xfer = true
use_multipath_for_image_xfer = true

[huawei_storage_high]
allowed_direct_url_schemes = cinder
cinder_huawei_conf_file = /etc/cinder/vendor-configs/dorado.xml
enforce_multipath_for_image_xfer = true
use_multipath_for_image_xfer = true
volume_backend_name = huawei_storage
volume_driver = cinder.volume.drivers.huawei.huawei_driver.HuaweiFCDriver
backend_host = huawei

image_volume_cache_enabled = true
```

6. Создайте файл config/multipath.conf с содержимым:

```
defaults {
  user_friendly_names no
  find_multipaths yes
}

blacklist {
}

devices {
  device {
    vendor "HUAWEI"
    product "XSG1"
    path_grouping_policy "multibus"
```

```
        path_selector "service-time 0"
        path_checker "tur"
        prio "const"
        failback "immediate"
        dev_loss_tmo 30
        fast_io_fail_tmo 5
        no_path_retry 15
    }
}
```

7. Создайте новый пайплайн: Build > Pipelines > Run Pipeline.
8. В открывшемся окне добавьте параметры:
    - KOLLA_ANSIBLE_DEPLOY_ACTION — deploy;
    - KOLLA_ARGS — -t cinder, multipath.
9. Запустите пайплайн: Run pipeline.
10. Дождитесь завершения задач на этапе **setup**.
11. Запустите задачу **deploy** на этапе **deploy**.
12. Дождитесь завершения выполнения операции.

Включение и конфигурация NFS

1. Откройте веб-интерфейс развернутого GitLab.
2. Откройте проект **project_k / deployments / <имя региона>**.
3. Добавьте строку в файл globals.d/REGION.yml:

```
enable_cinder_backend_nfs: "yes"
```

4. Создайте файл config/nfs_shares с записями о каждом Storage-узле:

```
storage01:/kolla_nfs
storage02:/kolla_nfs
```

5. На каждом Storage-узле:
    1. Создайте файл /etc/exports с информацией о монтировании хранилища, пример записи:

    ```
    /kolla_nfs 192.168.5.0/24(rw,sync,no_root_squash)
    ```

    2. Запустите сервис nfsd с помощью команды systemctl start nfs.
6. Создайте новый пайплайн: Build / Pipelines / Run Pipeline.
7. В открывшемся окне добавьте параметры:
    - KOLLA_ANSIBLE_DEPLOY_ACTION — deploy;
    - KOLLA_ARGS — -t cinder.
8. Запустите пайплайн: Run pipeline.
9. Дождитесь завершения задач на этапе **setup**.
10. Запустите задачу **deploy** на этапе **deploy**.
11. Дождитесь завершения выполнения операции.

4.2.7 Подключение служебных сервисов

4.2.7.1 Подключение Портала администратирования

При стандартной установке Портал администратора устанавливается и включается автоматически. Если этого не было сделано при развёртывании региона, а также при необходимости использования Портала администратора с несколькими регионами, обратитесь к разделу: Установка и настройка портала администратора.

Установка и настройка портала администриования

Варианты установки однорегионального и мультирегионального портала отличаются.

Установка и настройка однорегионального портала администратора

Чтобы установить портал администратора с одним регионом:

1. Откройте веб-интерфейс развернутого GitLab.
2. Откройте проект **project_k / deployments / <имя региона>**.
3. Найдите файл globals.d/REGION.yml и убедитесь, что в нём есть строки:

```
enable_adminui: "yes"
adminui_gitlab_username: "root"
```

4. Создайте новый пайплайн: Build > Pipelines > Run Pipeline.
5. В открывшемся окне добавьте параметр KOLLA_ARGS со значением -t adminui.
6. Запустите пайплайн: Run pipeline.
7. Дождитесь завершения выполнения операции.

Установка и настройка мультирегионального портала администратора

Чтобы подключить дополнительный регион к порталу администратора:

1. Зайдите на любой Control-узел подключаемого региона по SSH.
2. Откройте файл /etc/kolla/adminui-backend/adminui-backend-regions.conf. Информация из этого файла понадобится позже.
3. Откройте веб-интерфейс развернутого GitLab целевого региона.
4. Откройте проект **project_k / deployments / <имя региона>**.
5. Создайте файл config/adminui/adminui-backend-regions.conf и добавьте в него раздел [DEFAULT] с перечислением регионов, а также раздел для подключаемого региона, заменив параметры соответствующими значениями. Не указывайте в файле динамические параметры, в том числе пароли, в явном виде:

```
[DEFAULT]
region_names=<имя целевого региона>, <имя подключаемого региона>

[<имя подключаемого региона>]
type=keystack
prometheus_url=https://<VIP_АДРЕС_PROMETHEUS>:9091
```

gitlab_project_name=<имя подключаемого региона>
gitlab_repo_address=https://ks-lcm.cloud.itkey.com/project_k/devregion
grafana_url=https://<VIP адрес Grafana>:3000
opensearch_url=https://<VIP адрес OpenSearch>:5601
adminui_url=https://<VIP адрес Adminui>
gitlab_username={{ adminui_gitlab_username }}
gitlab_password={{ lookup('hashi_vault', 'secret={{ vault_engine }}/data/{{ vault_prefix }}/<имя подключаемого региона>/passwords_yml:adminui_gitlab_password') }}
vault_url={{ vault_addr }}/ui/vault/secrets/{{ vault_engine }}/list/{{ vault_prefix }}/<имя подключаемого региона>
os_region_name=<имя подключаемого региона>
os_auth_url=https://<VIP адрес портала администратора>:5000
os_interface=internal
os_endpoint_type=internalURL
os_username=admin
os_password={{ lookup('hashi_vault', 'secret={{ vault_engine }}/data/{{ vault_prefix }}/<имя подключаемого региона>/passwords_yml:keystone_admin_password') }}
os_project_name=admin
os_project_domain_name=Default
os_user_domain_name=Default
drs_endpoint = https://<VIP адрес DRS>:12998

6. Создайте новый пайплайн: Build > Pipelines > Run Pipeline.
7. В открывшемся окне добавьте параметр KOLLA_ARGS со значением -t adminui.
8. Запустите пайплайн: Run pipeline.
9. Дождитесь завершения выполнения операции.

4.2.7.2 Подключение сервисов мониторинга

CADF (Cloud Auditing Data Federation) – Стандарт для унификации и стандартизации сообщений аудита в облачных вычислениях. Он предоставляет формат для обмена информацией об аудите между различными облачными сервисами и платформами, что облегчает мониторинг, управление и отчетность в многопользовательских и мультиоблачных средах. CADF помогает организациям отслеживать действия и события в облаке, обеспечивая прозрачность и улучшая безопасность.

Мониторинг в KeyStack SE реализован на основе сервисов OpenSearch, Prometheus и Grafana. Вы можете получить доступ к к веб-интерфейсам этих сервисов по следующим ссылкам:

- OpenSearch — https://cloud.itkey.com:5601/;
- Prometheus — https://cloud.itkey.com:9091/;
- Grafana — https://cloud.itkey.com:3000/.

По умолчанию все перечисленные компоненты разворачиваются для каждого региона. Если этого не произошло или требуется дополнительная настройка, обратитесь к разделу Установка и настройка сервисов мониторинга.

Установка и настройка сервисов мониторинга

По умолчанию все компоненты мониторинга (OpenSearch, Prometheus, Alertmanager, Grafana) разворачиваются для каждого региона. Если это не было выполнено, выполните запуск сервисов мониторинга вручную. Для этого выполните следующие шаги:

1. Откройте веб-интерфейс развернутого GitLab.
2. Откройте проект **project_k / deployments / <имя региона>**.
3. Откройте файл globals.d/REGION.yml и установите значения yes для необходимых компонентов:

```
enable_grafana: "yes"
enable_prometheus: "yes"
enable_prometheus_alertmanager: "yes"
enable_opensearch: "yes"
enable_cadf_audit: "yes"
```

4. Создайте новый пайплайн: Build > Pipelines > Run Pipeline.
5. В открывшемся окне добавьте параметры:
   - KOLLA_ANSIBLE_DEPLOY_ACTION — deploy;
   - KOLLA_ARGS — укажите параметр -t с тегом нужного компонента. Можно перечислить несколько компонентов через запятую, например -t grafana,prometheus,opensearch.

   Примечание

   CADF является настройкой сервисов, поэтому для его включения необходимо запускать развёртывание без тегов.

6. Запустите пайплайн: Run pipeline.
7. Дождитесь завершения выполнения операции.

Настройка глубины хранения метрик в Prometheus

Чтобы настроить глубину хранения метрик в Prometheus, выполните перечисленные действия:

1. Откройте веб-интерфейс развернутого GitLab.
2. Откройте проект **project_k / deployments / <имя региона>**.
3. Внесите изменения в файл globals.d/REGION.yml:

```
prometheus_cmdline_extras: "--storage.tsdb.retention.time=60d --storage.tsdb.retention.size=500GB"
```

4. Создайте новый пайплайн: Build > Pipelines > Run Pipeline.
5. В открывшемся окне добавьте параметры:
   - KOLLA_ANSIBLE_DEPLOY_ACTION — deploy;
   - KOLLA_ARGS — -t prometheus.
6. Запустите пайплайн: Run pipeline.
7. Дождитесь завершения выполнения операции.

Создание шаблона индекса в OpenSearch

Чтобы создать шаблон индекса в OpenSearch:

1. Откройте веб-интерфейс развернутого OpenSearch. При первом входе вы будете перенаправлены на страницу создания шаблонов.
2. Нажмите кнопку Create index pattern.
3. В открывшемся окне в поле **Index pattern name** укажите значение flog*.
4. Нажмите кнопку Next Step.
5. В поле **Time field** выберите вариант @timestamp.
6. Нажмите кнопку Create index pattern.

Использование сервиса хранения метрик VictoriaMetrics

По умолчанию, в качестве сервиса хранения метрик используется Prometheus server. Вместо него или дополнительно к нему вы можете использовать сервис VictoriaMetrics. При одновременном включении Prometheus server и VictoriaMetrics оба сервиса будут работать независимо.

Для работы с VictoriaMetrics необходимо будет в Grafana вручную создать соответствующий data source.

Для развёртывания VictoriaMetrics в качестве сервиса хранения метрик, выполните следующие действия:

1. Откройте веб-интерфейс развернутого GitLab.
2. Откройте проект **project_k / deployments / <имя региона>**.
3. Внесите изменения в файл globals.d/REGION.yml:

enable_victoriametrics: "yes"
victoriametrics_vmstorage_cmdline_extras: "-retentionPeriod 15d -storage.minFreeDiskSpaceBytes 5000000"

где victoriametrics_vmstorage_cmdline_extras — необязательный параметр дополнительных опций запуска VictoriaMetrics, в котором:

   - -retentionPeriod — длительность хранения данных, по умолчанию 1 месяц;
   - -storage.minFreeDiskSpaceBytes — минимальное свободное место, после которого сервис перестаёт принимать новые данные, по умолчанию 10000000.
4. Создайте новый пайплайн: Build > Pipelines > Run Pipeline.
5. В открывшемся окне добавьте параметры:
   - KOLLA_ANSIBLE_DEPLOY_ACTION — deploy;
   - KOLLA_ARGS — -t victoriametrics.
6. Запустите пайплайн: Run pipeline.
7. Дождитесь завершения выполнения операции.

4.2.7.3 Подключение сервисов аудита

Для вывода логов в удаленный сервис системного журнала (syslog) используется fluent-plugin-remote_syslog — плагин Fluentd.

Настройка плагина по умолчанию:

```
<match **>
@type remote_syslog
host 10.120.120.125
port 514
protocol tcp
</match>
```

Чтобы изменить настройку плагина:

1. Откройте веб-интерфейс развернутого GitLab.
2. Откройте проект **project_k / deployments / <имя региона>**.
3. Перейдите в директорию config. Если такой директории нет, создайте её.
4. Создайте файл fluentd/output/fluent-plugin-remote_syslog.conf со следующим содержимым:

```
<match <регулярное выражение для фильтрации записей>>
  @type copy
  <store>
    @type opensearch
    host {{ opensearch_address }}
    port {{ opensearch_port }}
    scheme {{ fluentd_opensearch_scheme }}
{% if fluentd_opensearch_path != '' %}
    path {{ fluentd_opensearch_path }}
{% endif %}
{% if fluentd_opensearch_scheme == 'https' %}
    ssl_version {{ fluentd_opensearch_ssl_version }}
    ssl_verify {{ fluentd_opensearch_ssl_verify }}
{% if fluentd_opensearch_cacert | length > 0 %}
    ca_file {{ fluentd_opensearch_cacert }}
{% endif %}
{% endif %}
{% if fluentd_opensearch_user != '' and fluentd_opensearch_password != ''%}
    user {{ fluentd_opensearch_user }}
    password {{ fluentd_opensearch_password }}
{% endif %}
    logstash_format true
    logstash_prefix {{ opensearch_log_index_prefix }}
    reconnect_on_error true
    request_timeout {{ fluentd_opensearch_request_timeout }}
    suppress_type_name true
    <buffer>
    @type file
    path /var/lib/fluentd/data/opensearch.buffer/openstack.*
    flush_interval 15s
    </buffer>
  </store>
  <store>
    @type remote_syslog
    host <IP-адрес плагина>
    port <порт плагина>
```

```
      protocol <протокол для взаимодействия с плагином>
    </store>
  </match>
```

5. Создайте новый пайплайн: Build > Pipelines > Run Pipeline.
6. В открывшемся окне добавьте параметры:
   - KOLLA_ANSIBLE_DEPLOY_ACTION — deploy;
   - KOLLA_ARGS — -t opensearch.
7. Запустите пайплайн: Run pipeline.
8. Дождитесь завершения выполнения операции.
9. (Опционально) Убедитесь, что конфигурационный файл Fluentd td-agent.conf содержит добавленные данные.

Ниже приведён пример конфигурационного файла OpenSearch для отправки всех данных в сервис syslog по протоколу UDP на IP-адрес 10.10.140.100 и порт 7710:

```
<match **>
  @type copy
  <store>
    @type opensearch
    host {{ opensearch_address }}
    port {{ opensearch_port }}
    scheme {{ fluentd_opensearch_scheme }}
{% if fluentd_opensearch_path != '' %}
    path {{ fluentd_opensearch_path }}
{% endif %}
{% if fluentd_opensearch_scheme == 'https' %}
    ssl_version {{ fluentd_opensearch_ssl_version }}
    ssl_verify {{ fluentd_opensearch_ssl_verify }}
{% if fluentd_opensearch_cacert | length > 0 %}
    ca_file {{ fluentd_opensearch_cacert }}
{% endif %}
{% endif %}
{% if fluentd_opensearch_user != '' and fluentd_opensearch_password != ''%}
    user {{ fluentd_opensearch_user }}
    password {{ fluentd_opensearch_password }}
{% endif %}
    logstash_format true
    logstash_prefix {{ opensearch_log_index_prefix }}
    reconnect_on_error true
    request_timeout {{ fluentd_opensearch_request_timeout }}
    suppress_type_name true
    <buffer>
    @type file
    path /var/lib/fluentd/data/opensearch.buffer/openstack.*
    flush_interval 15s
    </buffer>
  </store>
  <store>
    @type remote_syslog
    host 10.10.140.100
```

```
    port 7710
    protocol udp
  </store>
</match>
```

4.2.8 Проверка работоспособности системы

Проверка работоспособности региона позволяет убедиться, что виртуальные машины создаются и доступны по сети, что подтверждает работоспособность цепочки сервисов (MariaDB, HAProxy, Cinder, Nova, Neutron, Glance).

- Подключитесь к интерфейсу OpenStack CLI.
- Проверьте сетевую доступность всех узлов облака с помощью команды `ping`.
- Выполните команду `openstack compute service list` для проверки состояния вычислительных сервисов. Убедитесь, что все сервисы находятся в состоянии `up`.
- Выполните команду `openstack volume service list` для проверки состояния службы томов. Убедитесь, что все сервисы находятся в состоянии `up`.
- Выполните команду `openstack server list` для проверки состояния виртуальных машин.
- Используя OpenStack CLI, портал самообслуживания AdminUI или Портал администратора, создайте несколько ВМ с различными флейворами и выполните их live-миграцию.

4.2.9 Создание резервной копии LCM-сервера

Для создания резервной копии выполните следующие шаги:

1. Подключитесь к LCM-серверу от имени пользователя root. Скопируйте содержимое скрипта `backupLCM.sh` (см. примечание 2, раздел «Устранение неисправностей при создании/восстановлении резервной копии LCM-сервера») в каталог, содержащий распакованный инсталлятор.
2. Перейдите в каталог с инсталлятором и установите права доступа на файл скрипта командой:

```
chmod 750 backupLCM.sh
```

3. Загрузите переменные, необходимые для работы скрипта, выполнив команду:

```
source /installer/config/settings
```

4. Запустите процесс создания резервной копии командой:

```
./backupLCM.sh
```

5. Дождитесь завершения работы скрипта. Скопируйте созданный файл резервной копии из каталога `/installer/backup` в надёжное хранилище.
6. Сохраните копию каталога с инсталлятором в надёжное хранилище.

4.2.10 Восстановление LCM-сервера из резервной копии

Для восстановления LCM-сервера из резервной копии используйте инсталлятор той же версии, что применялся при создании копии. Выполните следующие шаги:

1. Подключитесь к серверу LCM от имени пользователя root. Скопируйте содержимое скрипта `restoreLCM.sh` (см. примечание 3, раздел «Устранение неисправностей при

создании/восстановлении резервной копии LCM-сервера») в каталог с распакованным инсталлятором.

2. Перейдите в каталог с инсталлятором и установите права доступа на файл скрипта командой:

```
chmod 750 restoreLCM.sh
```

3. Поместите файл резервной копии в тот же каталог, где находится скрипт `restoreLCM.sh`.

4. Загрузите переменные, необходимые для работы скрипта, выполнив команду:

```
source /installer/config/settings
```

5. Если каталог установленным LCM-севрером отсутствует, выполните установку LCM-сервера в соответствии с инструкцией по установке, затем вернитесь к шагу 1 данной процедуры.

6. Убедитесь, что контейнеры LCM-сервера остановлены. При необходимости остановите их (см. примечание 1, раздел «Устранение неисправностей при создании/восстановлении резервной копии LCM-сервера»).

7. Запустите процесс восстановления командой:

```
./restoreLCM.sh
```

В процессе восстановления потребуется ввести пароли от восстанавливаемых сервисов.

8. Если после восстановления некоторые сервисы не запустились, обратитесь к разделу «Устранение неисправностей» или свяжитесь с представителем компании производителя ПО ОП «KS».

4.2.11 Настройка аудита логов в OpenSearch Dashboards

1. Настройка аудита логов при авторизации пользователей

Войдите в OpenSearch Dashboards под учетной записью администратора (например, admin). Затем перейдите в меню Management → Security → Audit logs.

Назначение страницы Audit logs: Эта страница предназначена для настройки аудита – отслеживания и записи действий пользователей в кластере OpenSearch. Аудит логирование позволяет фиксировать попытки доступа, успешные и неуспешные входы, запросы без необходимых прав и другие события безопасности. На странице Audit logs можно включить/выключить аудит и задать, какие категории событий логируются, а какие игнорируются, не прибегая к редактированию конфигурационных файлов вручную.

Конфигурация категорий аудита: В разделе General settings нажмите Configure. Найдите поле REST disabled categories (метка может отображаться как audit:disabled_rest_categories). В этом многострочном поле уже выбраны следующие категории событий, исключённые из аудита REST API: GRANTED_PRIVILEGES, BAD_HEADERS, MISSING_PRIVILEGES, SSL_EXCEPTION. Это означает, что события этих типов не будут записываться при взаимодействиях через REST-интерфейс OpenSearch. Если щёлкнуть по данному полю, раскроется список всех доступных категорий аудита – среди них вы увидите также FAILED_LOGIN и AUTHENTICATED. По умолчанию эти категории не добавлены в список отключенных, то есть события неудачного входа и успешной аутентификации пользователей логируются в аудите (они изначально включены для REST-слоя). При необходимости администратор может добавить их в поле отключённых категорий (выбрав из выпадающего списка) – после сохранения настройки эти события будут исключены из логирования. Однако в нашем случае FAILED_LOGIN и AUTHENTICATED оставляют активными, чтобы аудит фиксировал как неудачные попытки входа, так и успешные логины пользователей Opensearch.

Игнорирование системных пользователей: На данной же странице в разделе Ignore settings найдите поле Ignored users. Здесь указываются учетные записи, чьи действия не

должны фиксироваться в аудит-логах, чтобы отфильтровать лишние системные события. Пользователи, перечисленные в этом списке, будут проигнорированы механизмом аудита. Как правило, сюда включают служебные учетные записи. В нашем случае в списке игнорируемых указаны:

- kibanaserver – встроенный сервисный пользователь OpenSearch Dashboards (по умолчанию уже исключается из аудита).
- fluentd – учетная запись, под которой Fluentd подключается на каждой ноде для отправки логов в OpenSearch.
- dashboards – служебная учетная запись, которую OpenSearch Dashboards использует для подключения к самому OpenSearch.

Исключение этих пользователей из аудита позволяет не захламлять журнал частыми системными запросами и сосредоточиться на действиях реальных пользователей. Таким образом, на странице Audit logs настроен аудит логирования внутри самого OpenSearch (категории событий и игнорируемых пользователей) в соответствии с требованиями.

2. Поиск и просмотр логов аудита аутентификации пользователей

После включения и настройки аудита необходимо убедиться, что события авторизации действительно записываются, и научиться их просматривать через интерфейс OpenSearch Dashboards.

Создание шаблона индекса для аудит-логов: В меню Management (управление) перейдите в раздел Index Patterns (шаблоны индексов). Здесь отображается список уже созданных шаблонов. Например, в системе может уже существовать шаблон flog* (для других логов). Теперь создадим новый шаблон для индексированных аудит-логов безопасности. Нажмите Create index pattern (Создать шаблон индекса). В поле имени шаблона введите security-auditlog*. Этот шаблон покроет все индексы, в которых хранятся события аудита (по умолчанию OpenSearch создает индексы аудита с префиксом security-auditlog- и датой, например security-auditlog-2025.12.06). На шаге выбора временного поля укажите @timestamp (это поле присутствует в документах аудит-логов и содержит метку времени события). Затем сохраните новый шаблон.

Просмотр событий в Discover: Перейдите в раздел Discover. В верхней части окна Discover, в выпадающем списке выбора данных (рядом с заголовком), теперь доступны два варианта индексов: ранее существовавший flog* и созданный нами security-auditlog*. Выберите индексный шаблон security-auditlog*. После этого в области результатов появятся документы из аудит-логов безопасности OpenSearch. Каждый документ представляет одно событие аудита – например, успешная аутентификация или неудачный логин. Вы можете увидеть поля каждого события: @timestamp (время), audit_category (категория события), audit_request_effective_user (имя пользователя, от которого выполнен запрос), audit_request_remote_address (IP-адрес источника), и другие детали. Обращайте внимание на поле audit_category – в нем указано, к какому типу относится событие. Например, для попыток входа без правильных учетных данных будет категория FAILED_LOGIN, а для успешных входов – AUTHENTICATED. Используя фильтры Discover, можно отфильтровать события по конкретным категориям (например, показать только неудачные логины) или по пользователям/IP.

Таким образом, через интерфейс Discover производится демонстрируем, что настроенный аудит работает: журнал security-auditlog* содержит записи обо всех попытках аутентификации пользователей внутри OpenSearch

4.3 ПОДГОТОВКА К РАБОТЕ

5.1 Проверка соответствия требованиям

Перед началом эксплуатации ПО необходимо проверить его соответствие требованиям к комплектности, упаковке и маркировке, отраженным в документе «Программное обеспечение «Облачная платформа KeyStack SE v.1». Формуляр», 1087746411378.62.01.29.000.001 ФО в зависимости от вариантов исполнения ПО.

При проверке комплектности проводится проверка целостности и качества записи дистрибутива программного обеспечения ПО на оптическом носителе информации посредством подсчета контрольной суммы дистрибутива и сравнения их с контрольной суммой, отраженной в документе «Программное обеспечение «Облачная платформа KeyStack SE v.1». Формуляр», 1087746411378.62.01.29.000.001 ФО. Необходимо произвести резервное копирование конфигурации ПО (не менее 2-х копий) для обновления (при необходимости) и проверки целостности ПО.

5.2 Подготовка ПО «KeyStack SE» к работе

Перечень программного обеспечения, включающий системное, базовое и прикладное ПО, приведен в Пояснительной записке.

Системный администратор использует в работе следующие интерфейсы Платформы, доступные по представленным далее ссылкам:

Портал Администрирования;
Opensearch;
Grafana;
Prometheus
GitLab
Hashicorp Vault
NetBox

Основным рабочим веб-интерфейсом является Портал Администрирования.

Помимо веб-интерфейса, в работе может быть исользована утилита Openstack CLI.

5.3 Подключение к Порталу администратора

Используйте Портал администратора для управления вычислительными ресурсами облака.

Портал администратора может использоваться для управления как одним, так и несколькими регионами.

Подключение к интерфейсу управления Портала администратора производится с помощью веб-браузера. Используйте браузеры актуальных версий для доступа к Порталу:

- Google Chrome 72 и выше;
- Mozilla Firefox 63 и выше.

При запросе логина и пароля введите учётные данные.

Подключение к интерфейсу управления NetBox

NetBox — компонент в составе инсталлятора, размещенный на LCM-узле. Чтобы подключиться к пользовательскому веб-интерфейсу, перейдите по адресу https://netbox.cloud.itkey.com.

Стартовая страница отображает разделы, выбранные и настроенные самостоятельно.

Подключение к интерфейсу управления Openstack CLI

Интерфейс управления OpenStack CLI позволяет выполнять команды OpenStack в интерфейсе командной строки. OpenStack CLI может быть установлен на узел LCM или любой другой внешний узел.

Для установки и настройки OpenStack CLI обратитесь к инструкции: Подключение к интерфейсу командной строки OpenStack.

Для получения справки по командам OpenStack CLI обратитесь к документации OpenStack: https://docs.openstack.org/ocata/user-guide/cli-cheat-sheet.html.

Подключение к интерфейсу командной строки OpenStack

Для выполнения команд OpenStack необходимо подключение к интерфейсу командной строки (OpenStack CLI). Для управления сервисом DRS используется интерфейс командной строки drsclient. Эти клиенты уже входят в поставку LCM-узла KeyStack, но их также можно установить на рабочее место администратора. Ниже приведен порядок установки интерфейсов командной строки для различных ОС.

Установка клиентов CLI под Linux

Установка выполняется с помощью менеджера пакетов Python **pip**. Чтобы установить необходимые пакеты, выполните команды:

$ sudo pip install drsclient
$ sudo pip install python-cinderclient
$ sudo pip install python-novaclient
$ sudo pip install python-openstackclient

Получение файла openrc

Для подключения к CLI необходимо загрузить исходный файл OpenStack RC**.** Этот файл устанавливает переменные окружения, необходимые для подключения и авторизации клиента OpenStack CLI. Файл `openrc` можно получить разными способами.

Получение файла openrc из Vault

При получении файла `openrc` этим способом в нём явно указывается пароль пользователя admin. Будьте осторожны при обращении с файлом.
Зайдите на LCM-узел через SSH.

Выполните команду экспорта файла `openrc` :

```
# podman exec -ti vault vault read secret_v2/data/deployments/<ks-lcm.cloud.itkey.com>/<имя региона>/openrc -format=json | jq -r '.data.data.value' > openrc-<имя региона>.sh
```

Применение файла openrc

Если вы используете собственный сертификат CA, убедитесь, что цепочка CA-сертификатов `chain-cert.pem` сохранена в директории /installer/data/ca/cert/ сервиса Vault.
В импортированный файл добавьте строку:
export OS_CACERT=/installer/data/ca/cert/chain-ca.pem

Скопируйте файл `openrc` на компьютер с установленным клиентом OpenStack CLI.

Выполните команду для вашего файла для установки переменных окружения:

$ source admin-openrc.sh

Проверка работоспособности OpenStack CLI

Для проверки корректности настроек выполните команду, которая возвращает список виртуальных машин:

$ openstack server list

Полный список команд и другая справочная информация по OpenStack CLI: https://docs.openstack.org/python-openstackclient/latest/index.html

Перечень команд доступных утилите и совместимых с ОО:

access rule delete Delete access rule(s)
access rule list List access rules
access rule show Display access rule details
access token create Create an access token
address group create Create a new Address Group
address group delete Delete address group(s)
address group list List address groups
address group set Set address group properties
address group show Display address group details
address group unset Unset address group properties
address scope create Create a new Address Scope
address scope delete Delete address scope(s)
address scope list List address scopes
address scope set Set address scope properties
address scope show Display address scope details
aggregate add host Add host to aggregate
aggregate cache image Request image caching for aggregate
aggregate create Create a new aggregate
aggregate delete Delete existing aggregate(s)
aggregate list List all aggregates
aggregate remove host Remove host from aggregate
aggregate set Set aggregate properties
aggregate show Display aggregate details
aggregate unset Unset aggregate properties
application credential create Create new application credential
application credential delete Delete application credentials(s)
application credential list List application credentials
application credential show Display application credential details
availability zone list List availability zones and their status
bgp dragent add speaker Add a BGP speaker to a dynamic routing agent (python-neutronclient)
bgp dragent list List dynamic routing agents (python-neutronclient)
bgp dragent remove speaker Removes a BGP speaker from a dynamic routing agent (python-neutronclient)
bgp peer create Create a BGP peer (python-neutronclient)
bgp peer delete Delete a BGP peer (python-neutronclient)
bgp peer list List BGP peers (python-neutronclient)
bgp peer set Update a BGP peer (python-neutronclient)
bgp peer show Show information for a BGP peer (python-neutronclient)
bgp speaker add network Add a network to a BGP speaker (python-neutronclient)
bgp speaker add peer Add a peer to a BGP speaker (python-neutronclient)
bgp speaker create Create a BGP speaker (python-neutronclient)
bgp speaker delete Delete a BGP speaker (python-neutronclient)
bgp speaker list List BGP speakers (python-neutronclient)
bgp speaker list advertised routes List routes advertised (python-neutronclient)
bgp speaker remove network Remove a network from a BGP speaker (python-neutronclient)
bgp speaker remove peer Remove a peer from a BGP speaker (python-neutronclient)

bgp speaker set Set BGP speaker properties (python-neutronclient)
bgp speaker show Show a BGP speaker (python-neutronclient)
bgpvpn create Create BGP VPN resource (python-neutronclient)
bgpvpn delete Delete BGP VPN resource(s) (python-neutronclient)
bgpvpn list List BGP VPN resources (python-neutronclient)
bgpvpn network association create Create a BGP VPN network association (python-neutronclient)
bgpvpn network association delete Delete a BGP VPN network association(s) for a given BGP VPN (python-neutronclient)
bgpvpn network association list List BGP VPN network associations for a given BGP VPN (python-neutronclient)
bgpvpn network association show Show information of a given BGP VPN network association (python-neutronclient)
bgpvpn port association create Create a BGP VPN port association (python-neutronclient)
bgpvpn port association delete Delete a BGP VPN port association(s) for a given BGP VPN (python-neutronclient)
bgpvpn port association list List BGP VPN port associations for a given BGP VPN (python-neutronclient)
bgpvpn port association set Set BGP VPN port association properties (python-neutronclient)
bgpvpn port association show Show information of a given BGP VPN port association (python-neutronclient)
bgpvpn port association unset Unset BGP VPN port association properties (python-neutronclient)
bgpvpn router association create Create a BGP VPN router association (python-neutronclient)
bgpvpn router association delete Delete a BGP VPN router association(s) for a given BGP VPN (python-neutronclient)
bgpvpn router association list List BGP VPN router associations for a given BGP VPN (python-neutronclient)
bgpvpn router association set Set BGP VPN router association properties (python-neutronclient)
bgpvpn router association show Show information of a given BGP VPN router association (python-neutronclient)
bgpvpn router association unset Unset BGP VPN router association properties (python-neutronclient)
bgpvpn set Set BGP VPN properties (python-neutronclient)
bgpvpn show Show information of a given BGP VPN (python-neutronclient)
bgpvpn unset Unset BGP VPN properties (python-neutronclient)
block storage cleanup Do block storage cleanup
block storage cluster list List block storage clusters
block storage cluster set Set block storage cluster properties
block storage cluster show Show detailed information for a block storage cluster
block storage log level list List log levels of block storage service
block storage log level set Set log level of block storage service
block storage resource filter list List block storage resource filters
block storage resource filter show Show filters for a block storage resource type
block storage snapshot manageable list List manageable snapshots
block storage volume manageable list List manageable volumes
cached image clear Clear all images from cache, queue or both
cached image delete Delete image(s) from cache
cached image list Get Cache State
cached image queue Queue image(s) for caching
catalog list List services in the service catalog
catalog show Display service catalog details
command list List recognized commands by group
complete print bash completion command (cliff)
compute agent create Create compute agent
compute agent delete Delete compute agent(s)

compute agent list List compute agents
compute agent set Set compute agent properties
compute service delete Delete compute service(s)
compute service list List compute services
compute service set Set compute service properties
configuration show Display configuration details
consistency group add volume Add volume(s) to consistency group
consistency group create Create new consistency group
consistency group delete Delete consistency group(s)
consistency group list List consistency groups
consistency group remove volume Remove volume(s) from consistency group
consistency group set Set consistency group properties
consistency group show Display consistency group details
consistency group snapshot create Create new consistency group snapshot
consistency group snapshot delete Delete consistency group snapshot(s)
consistency group snapshot list List consistency group snapshots
consistency group snapshot show Display consistency group snapshot details
console log show Show server's console output
console url show Show server's remote console URL
consumer create Create new consumer
consumer delete Delete consumer(s)
consumer list List consumers
consumer set Set consumer properties
consumer show Display consumer details
container create Create new container
container delete Delete container
container list List containers
container save Save container contents locally
container set Set container properties
container show Display container details
container unset Unset container properties
credential create Create new credential
credential delete Delete credential(s)
credential list List credentials
credential set Set credential properties
credential show Display credential details
default security group rule create Add a new security group rule to the default security group template
default security group rule delete Remove security group rule(s) from the default security group template
default security group rule list List security group rules used for new default security groups
default security group rule show Show a security group rule used for new default security groups
domain create Create new domain
domain delete Delete domain(s)
domain list List domains
domain set Set domain properties
domain show Display domain details
ec2 credentials create Create EC2 credentials
ec2 credentials delete Delete EC2 credentials
ec2 credentials list List EC2 credentials
ec2 credentials show Display EC2 credentials details
endpoint add project Associate a project to an endpoint
endpoint create Create new endpoint
endpoint delete Delete endpoint(s)
endpoint group add project Add a project to an endpoint group

endpoint group create Create new endpoint group
endpoint group delete Delete endpoint group(s)
endpoint group list List endpoint groups
endpoint group remove project Remove project from endpoint group
endpoint group set Set endpoint group properties
endpoint group show Display endpoint group details
endpoint list List endpoints
endpoint remove project Dissociate a project from an endpoint
endpoint set Set endpoint properties
endpoint show Display endpoint details
extension list List API extensions
extension show Show API extension
federation domain list List accessible domains
federation project list List accessible projects
federation protocol create Create new federation protocol
federation protocol delete Delete federation protocol(s)
federation protocol list List federation protocols
federation protocol set Set federation protocol properties
federation protocol show Display federation protocol details
firewall group create Create a new firewall group (python-neutronclient)
firewall group delete Delete firewall group(s) (python-neutronclient)
firewall group list List firewall groups (python-neutronclient)
firewall group policy add rule Insert a rule into a given firewall policy (python-neutronclient)
firewall group policy create Create a new firewall policy (python-neutronclient)
firewall group policy delete Delete firewall policy(s) (python-neutronclient)
firewall group policy list List firewall policies (python-neutronclient)
firewall group policy remove rule Remove a rule from a given firewall policy (python-neutronclient)
firewall group policy set Set firewall policy properties (python-neutronclient)
firewall group policy show Display firewall policy details (python-neutronclient)
firewall group policy unset Unset firewall policy properties (python-neutronclient)
firewall group rule create Create a new firewall rule (python-neutronclient)
firewall group rule delete Delete firewall rule(s) (python-neutronclient)
firewall group rule list List firewall rules that belong to a given tenant (python-neutronclient)
firewall group rule set Set firewall rule properties (python-neutronclient)
firewall group rule show Display firewall rule details (python-neutronclient)
firewall group rule unset Unset firewall rule properties (python-neutronclient)
firewall group set Set firewall group properties (python-neutronclient)
firewall group show Display firewall group details (python-neutronclient)
firewall group unset Unset firewall group properties (python-neutronclient)
flavor create Create new flavor
flavor delete Delete flavor(s)
flavor list List flavors
flavor set Set flavor properties
flavor show Display flavor details
flavor unset Unset flavor properties
floating ip create Create floating IP
floating ip delete Delete floating IP(s)
floating ip list List floating IP(s)
floating ip pool list List pools of floating IP addresses
floating ip port forwarding create Create floating IP port forwarding
floating ip port forwarding delete Delete floating IP port forwarding
floating ip port forwarding list List floating IP port forwarding
floating ip port forwarding set Set floating IP Port Forwarding Properties
floating ip port forwarding show Display floating IP Port Forwarding details
floating ip set Set floating IP Properties

floating ip show Display floating IP details
floating ip unset Unset floating IP Properties
group add user Add user to group
group contains user Check user membership in group
group create Create new group
group delete Delete group(s)
group list List groups
group remove user Remove user from group
group set Set group properties
group show Display group details
help print detailed help for another command (cliff)
host list DEPRECATED: List hosts
host set Set host properties
host show DEPRECATED: Display host details
hypervisor list List hypervisors
hypervisor show Display hypervisor details
hypervisor stats show Display hypervisor stats details
identity provider create Create new identity provider
identity provider delete Delete identity provider(s)
identity provider list List identity providers
identity provider set Set identity provider properties
identity provider show Display identity provider details
image add project Associate project with image
image create Create/upload an image
image delete Delete image(s)
image import Initiate the image import process
image import info Show available import methods
image list List available images
image member get Show a particular project associated with image
image member list List projects associated with image
image metadef namespace create Create a metadef namespace
image metadef namespace delete Delete metadef namespace
image metadef namespace list List metadef namespaces
image metadef namespace set Set metadef namespace properties
image metadef namespace show Show a metadef namespace
image metadef object create Create a metadef object
image metadef object delete Delete metadata definitions object(s)
image metadef object list List metadef objects inside a specific namespace
image metadef object show Show a particular metadef object
image metadef object update Update a metadef object
image metadef property create Create a metadef property
image metadef property delete Delete metadef propert(ies)
image metadef property list List metadef properties
image metadef property set Update metadef namespace property
image metadef property show Show a particular metadef property
image metadef resource type list List metadef resource types
image remove project Disassociate project with image
image save Save an image locally
image set Set image properties
image show Display image details
image stage Upload data for a specific image to staging
image stores list Get available backends (only valid with Multi-Backend support)
image task list List tasks
image task show Display task details
image unset Unset image tags and properties
implied role create Creates an association between prior and implied roles

implied role delete Deletes an association between prior and implied roles
implied role list List implied roles
ip availability list List IP availability for network
ip availability show Show network IP availability details
keypair create Create new public or private key for server ssh access
keypair delete Delete public or private key(s)
keypair list List key fingerprints
keypair show Display key details
limit create Create a limit
limit delete Delete a limit
limit list List limits
limit set Update information about a limit
limit show Display limit details
limits show Show compute and block storage limits
local ip association create Create Local IP Association
local ip association delete Delete Local IP association(s)
local ip association list List Local IP Associations
local ip create Create Local IP
local ip delete Delete local IP(s)
local ip list List local IPs
local ip set Set local ip properties
local ip show Display local IP details
mapping create Create new mapping
mapping delete Delete mapping(s)
mapping list List mappings
mapping set Set mapping properties
mapping show Display mapping details
module list List module versions
network agent add network Add network to an agent
network agent add router Add router to an agent
network agent delete Delete network agent(s)
network agent list List network agents
network agent remove network Remove network from an agent
network agent remove router Remove router from an agent
network agent set Set network agent properties
network agent show Display network agent details
network auto allocated topology create Create the auto allocated topology for project
network auto allocated topology delete Delete auto allocated topology for project
network create Create new network
network delete Delete network(s)
network flavor add profile Add a service profile to a network flavor
network flavor create Create new network flavor
network flavor delete Delete network flavors
network flavor list List network flavors
network flavor profile create Create new network flavor profile
network flavor profile delete Delete network flavor profile
network flavor profile list List network flavor profile(s)
network flavor profile set Set network flavor profile properties
network flavor profile show Display network flavor profile details
network flavor remove profile Remove service profile from network flavor
network flavor set Set network flavor properties
network flavor show Display network flavor details
network l3 conntrack helper create Create a new L3 conntrack helper
network l3 conntrack helper delete Delete L3 conntrack helper
network l3 conntrack helper list List L3 conntrack helpers
network l3 conntrack helper set Set L3 conntrack helper properties

network l3 conntrack helper show Display L3 conntrack helper details
network list List networks
network log create Create a new network log (python-neutronclient)
network log delete Delete network log(s) (python-neutronclient)
network log list List network logs (python-neutronclient)
network log set Set network log properties (python-neutronclient)
network log show Display network log details (python-neutronclient)
network loggable resources list List supported loggable resources (python-neutronclient)
network meter create Create network meter
network meter delete Delete network meter
network meter list List network meters
network meter rule create Create a new meter rule
network meter rule delete Delete meter rule(s)
network meter rule list List meter rules
network meter rule show Display meter rules details
network meter show Show network meter
network onboard subnets Onboard network subnets into a subnet pool (python-neutronclient)
network qos policy create Create a QoS policy
network qos policy delete Delete Qos Policy(s)
network qos policy list List QoS policies
network qos policy set Set QoS policy properties
network qos policy show Display QoS policy details
network qos rule create Create new Network QoS rule
network qos rule delete Delete Network QoS rule
network qos rule list List Network QoS rules
network qos rule set Set Network QoS rule properties
network qos rule show Display Network QoS rule details
network qos rule type list List QoS rule types
network qos rule type show Show details about supported QoS rule type
network rbac create Create network RBAC policy
network rbac delete Delete network RBAC policy(s)
network rbac list List network RBAC policies
network rbac set Set network RBAC policy properties
network rbac show Display network RBAC policy details
network segment create Create new network segment
network segment delete Delete network segment(s)
network segment list List network segments
network segment range create Create new network segment range
network segment range delete Delete network segment range(s)
network segment range list List network segment ranges
network segment range set Set network segment range properties
network segment range show Display network segment range details
network segment set Set network segment properties
network segment show Display network segment details
network service provider list List Service Providers
network set Set network properties
network show Show network details
network subport list List all subports for a given network trunk
network trunk create Create a network trunk for a given project
network trunk delete Delete a given network trunk
network trunk list List all network trunks
network trunk set Set network trunk properties
network trunk show Show information of a given network trunk
network trunk unset Unset subports from a given network trunk
network unset Unset network properties
object create Upload object to container

object delete Delete object from container
object list List objects
object save Save object locally
object set Set object properties
object show Display object details
object store account set Set account properties
object store account show Display account details
object store account unset Unset account properties
object unset Unset object properties
policy create Create new policy
policy delete Delete policy(s)
policy list List policies
policy set Set policy properties
policy show Display policy details
port create Create a new port
port delete Delete port(s)
port list List ports
port set Set port properties
port show Display port details
port unset Unset port properties
project cleanup Clean resources associated with a project
project create Create new project
project delete Delete project(s)
project list List projects
project set Set project properties
project show Display project details
quota delete Delete configured quota for a project and revert to defaults
quota list List quotas for all projects with non-default quota values or list detailed quota information for requested project
quota set Set quotas for project or class
quota show Show quotas for project or class
region create Create new region
region delete Delete region(s)
region list List regions
region set Set region properties
region show Display region details
registered limit create Create a registered limit
registered limit delete Delete a registered limit
registered limit list List registered limits
registered limit set Update information about a registered limit
registered limit show Display registered limit details
request token authorize Authorize a request token
request token create Create a request token
role add Adds a role assignment to a user or group on the system, a domain, or a project
role assignment list List role assignments
role create Create new role
role delete Delete role(s)
role list List roles
role remove Removes a role assignment from system/domain/project : user/group
role set Set role properties
role show Display role details
router add port Add a port to a router
router add route Add extra static routes to a router's routing table
router add subnet Add a subnet to a router
router create Create a new router
router delete Delete router(s)

router list List routers
router ndp proxy create Create NDP proxy
router ndp proxy delete Delete NDP proxy
router ndp proxy list List NDP proxies
router ndp proxy set Set NDP proxy properties
router ndp proxy show Display NDP proxy details
router remove port Remove a port from a router
router remove route Remove extra static routes from a router's routing table
router remove subnet Remove a subnet from a router
router set Set router properties
router show Display router details
router unset Unset router properties
security group create Create a new security group
security group delete Delete security group(s)
security group list List security groups
security group rule create Create a new security group rule
security group rule delete Delete security group rule(s)
security group rule list List security group rules
security group rule show Display security group rule details
security group set Set security group properties
security group show Display security group details
security group unset Unset security group properties
server add fixed ip Add fixed IP address to server
server add floating ip Add floating IP address to server
server add network Add network to server
server add port Add port to server
server add security group Add security group to server
server add volume Add volume to server
server backup create Create a server backup image
server create Create a new server
server delete Delete server(s)
server dump create Create a dump file in server(s)
server evacuate Evacuate a server to a different host
server event list List recent events of a server
server event show Show server event details
server group create Create a new server group
server group delete Delete existing server group(s)
server group list List all server groups
server group show Display server group details
server image create Create a new server disk image from an existing server
server list List servers
server lock Lock server(s)
server migrate Migrate server to different host
server migrate confirm DEPRECATED: Use 'server migration confirm' instead
server migrate revert DEPRECATED: Use 'server migration revert' instead
server migration abort Cancel an ongoing live migration
server migration confirm Confirm server migration
server migration force complete Force an ongoing live migration to complete
server migration list List server migrations
server migration revert Revert server migration
server migration show Show an in-progress live migration for a given server
server pause Pause server(s)
server reboot Perform a hard or soft server reboot
server rebuild Rebuild server
server remove fixed ip Remove fixed IP address from server
server remove floating ip Remove floating IP address from server

server remove network Remove all ports of a network from server
server remove port Remove port from server
server remove security group Remove security group from server
server remove volume Remove volume from server
server rescue Put server in rescue mode
server resize Scale server to a new flavor
server resize confirm Confirm server resize
server resize revert Revert server resize
server restore Restore server(s)
server resume Resume server(s)
server set Set server properties
server shelve Shelve and optionally offload server(s)
server show Show server details
server ssh SSH to server
server start Start server(s)
server stop Stop server(s)
server suspend Suspend server(s)
server unlock Unlock server(s)
server unpause Unpause server(s)
server unrescue Restore server from rescue mode
server unset Unset server properties and tags
server unshelve Unshelve server(s)
server volume list List all the volumes attached to a server
server volume set Update a volume attachment on the server
server volume update DEPRECATED: Use 'server volume set' instead
service create Create new service
service delete Delete service(s)
service list List services
service provider create Create new service provider
service provider delete Delete service provider(s)
service provider list List service providers
service provider set Set service provider properties
service provider show Display service provider details
service set Set service properties
service show Display service details
sfc flow classifier create Create a flow classifier (python-neutronclient)
sfc flow classifier delete Delete a given flow classifier (python-neutronclient)
sfc flow classifier list List flow classifiers (python-neutronclient)
sfc flow classifier set Set flow classifier properties (python-neutronclient)
sfc flow classifier show Display flow classifier details (python-neutronclient)
sfc port chain create Create a port chain (python-neutronclient)
sfc port chain delete Delete a given port chain (python-neutronclient)
sfc port chain list List port chains (python-neutronclient)
sfc port chain set Set port chain properties (python-neutronclient)
sfc port chain show Display port chain details (python-neutronclient)
sfc port chain unset Unset port chain properties (python-neutronclient)
sfc port pair create Create a port pair (python-neutronclient)
sfc port pair delete Delete a given port pair (python-neutronclient)
sfc port pair group create Create a port pair group (python-neutronclient)
sfc port pair group delete Delete a given port pair group (python-neutronclient)
sfc port pair group list List port pair group (python-neutronclient)
sfc port pair group set Set port pair group properties (python-neutronclient)
sfc port pair group show Display port pair group details (python-neutronclient)
sfc port pair group unset Unset port pairs from port pair group (python-neutronclient)
sfc port pair list List port pairs (python-neutronclient)
sfc port pair set Set port pair properties (python-neutronclient)

sfc port pair show Display port pair details (python-neutronclient)
sfc service graph create Create a service graph (python-neutronclient)
sfc service graph delete Delete a given service graph (python-neutronclient)
sfc service graph list List service graphs (python-neutronclient)
sfc service graph set Set service graph properties (python-neutronclient)
sfc service graph show Show information of a given service graph (python-neutronclient)
subnet create Create a subnet
subnet delete Delete subnet(s)
subnet list List subnets
subnet pool create Create subnet pool
subnet pool delete Delete subnet pool(s)
subnet pool list List subnet pools
subnet pool set Set subnet pool properties
subnet pool show Display subnet pool details
subnet pool unset Unset subnet pool properties
subnet set Set subnet properties
subnet show Display subnet details
subnet unset Unset subnet properties
token issue Issue new token
token revoke Revoke existing token
trust create Create new trust
trust delete Delete trust(s)
trust list List trusts
trust show Display trust details
usage list List resource usage per project
usage show Show resource usage for a single project
user create Create new user
user delete Delete user(s)
user list List users
user password set Change current user password
user set Set user properties
user show Display user details
versions show Show available versions of services
volume attachment complete Complete an attachment for a volume
volume attachment create Create an attachment for a volume
volume attachment delete Delete an attachment for a volume
volume attachment list Lists all volume attachments
volume attachment set Update an attachment for a volume
volume attachment show Show detailed information for a volume attachment
volume backend capability show Show capability command
volume backend pool list List pool command
volume backup create Create new volume backup
volume backup delete Delete volume backup(s)
volume backup list List volume backups
volume backup record export Export volume backup details
volume backup record import Import volume backup details
volume backup restore Restore volume backup
volume backup set Set volume backup properties
volume backup show Display volume backup details
volume backup unset Unset volume backup properties
volume create Create new volume
volume delete Delete volume(s)
volume group create Create a volume group
volume group delete Delete a volume group
volume group failover Failover replication for a volume group
volume group list Lists all volume groups

volume group set Update a volume group
volume group show Show detailed information for a volume group
volume group snapshot create Create a volume group snapshot
volume group snapshot delete Delete a volume group snapshot
volume group snapshot list Lists all volume group snapshot
volume group snapshot show Show detailed information for a volume group snapshot
volume group type create Create a volume group type
volume group type delete Delete a volume group type
volume group type list Lists all volume group types
volume group type set Update a volume group type
volume group type show Show detailed information for a volume group type
volume host set Set volume host properties
volume list List volumes
volume message delete Delete a volume failure message
volume message list List volume failure messages
volume message show Show a volume failure message
volume migrate Migrate volume to a new host
volume qos associate Associate a QoS specification to a volume type
volume qos create Create new QoS specification
volume qos delete Delete QoS specification
volume qos disassociate Disassociate a QoS specification from a volume type
volume qos list List QoS specifications
volume qos set Set QoS specification properties
volume qos show Display QoS specification details
volume qos unset Unset QoS specification properties
volume revert Revert a volume to a snapshot
volume service list List service command
volume service set Set volume service properties
volume set Set volume properties
volume show Display volume details
volume snapshot create Create new volume snapshot
volume snapshot delete Delete volume snapshot(s)
volume snapshot list List volume snapshots
volume snapshot set Set volume snapshot properties
volume snapshot show Display volume snapshot details
volume snapshot unset Unset volume snapshot properties
volume summary Show a summary of all volumes in this deployment
volume transfer request accept Accept volume transfer request
volume transfer request create Create volume transfer request
volume transfer request delete Delete volume transfer request(s)
volume transfer request list Lists all volume transfer requests
volume transfer request show Show volume transfer request details
volume type create Create new volume type
volume type delete Delete volume type(s)
volume type list List volume types
volume type set Set volume type properties
volume type show Display volume type details
volume type unset Unset volume type properties
volume unset Unset volume properties
vpn endpoint group create Create an endpoint group (python-neutronclient)
vpn endpoint group delete Delete endpoint group(s) (python-neutronclient)
vpn endpoint group list List endpoint groups that belong to a given project (python-neutronclient)
vpn endpoint group set Set endpoint group properties (python-neutronclient)
vpn endpoint group show Display endpoint group details (python-neutronclient)
vpn ike policy create Create an IKE policy (python-neutronclient)

vpn ike policy delete Delete IKE policy (policies) (python-neutronclient)
vpn ike policy list List IKE policies that belong to a given project (python-neutronclient)
vpn ike policy set Set IKE policy properties (python-neutronclient)
vpn ike policy show Display IKE policy details (python-neutronclient)
vpn ipsec policy create Create an IPsec policy (python-neutronclient)
vpn ipsec policy delete Delete IPsec policy(policies) (python-neutronclient)
vpn ipsec policy list List IPsec policies that belong to a given project (python-neutronclient)
vpn ipsec policy set Set IPsec policy properties (python-neutronclient)
vpn ipsec policy show Display IPsec policy details (python-neutronclient)
vpn ipsec site connection create Create an IPsec site connection (python-neutronclient)
vpn ipsec site connection delete Delete IPsec site connection(s) (python-neutronclient)
vpn ipsec site connection list List IPsec site connections that belong to a given project (python-neutronclient)
vpn ipsec site connection set Set IPsec site connection properties (python-neutronclient)
vpn ipsec site connection show Show information of a given IPsec site connection (python-neutronclient)
vpn service create Create an VPN service (python-neutronclient)
vpn service delete Delete VPN service(s) (python-neutronclient)
vpn service list List VPN services that belong to a given project (python-neutronclient)
vpn service set Set VPN service properties (python-neutronclient)
vpn service show Display VPN service details (python-neutronclient)

5. ПРИНЦИПЫ БЕЗОПАСНОЙ РАБОТЫ ПО «KeyStack SE»

6.1 Защита ПО «KeyStack SE» от несанкционированного доступа

Обеспечение собственной защиты ПО «KeyStack SE» от несанкционированного доступа осуществляется за счет:

- наличия ответственного лица (администратора среды функционирования – операционной системы), отвечающего за установку компонентов логической сущности LCM;
- настройка компонентов LCM должна осуществляться в соответствии с эксплуатационной документацией (руководством администратора ПО «KeyStack SE»);
- выполнения установки, конфигурирования и управления ПО «KeyStack SE» в соответствии с эксплуатационной документацией;
- обеспечения физической сохранности аппаратной платформы с установленным ПО «KeyStack SE» и исключением возможности доступа к ней посторонних лиц;
- выделением сетевых сегментов для работы. Требования к сегментации сети при настройке ПО «KeyStack SE» приведены в пункте 2.4 настоящего документа.
- обеспечения предотвращения несанкционированного доступа к идентификаторам и паролям привилегированных пользователей (администратора);
- предоставления пользователям прав доступа к объектам доступа информационной системы обеспечивается, основываясь на задачах, решаемых пользователями в ПО «KeyStack SE» и взаимодействующими с ней информационными системами;
- разделения полномочий (ролей) пользователей, администраторов и лиц, обеспечивающих функционирование ПО «KeyStack SE»;
- осуществления доступа к объектам доступа ПО «KeyStack SE» с учетом минимально необходимых прав и привилегий;
- предоставления прав и привилегий по доступу к функциям безопасности (параметрам настройки) ПО «KeyStack SE» исключительно администратору, наделенному полномочиями в соответствии с требованиями к функциям безопасности;
- параметры и настройки ПО «KeyStack SE» хранятся на уровне файловой системы и базы данных, эти ресурсы недоступны пользователям ПО «KeyStack SE» через предоставляемые для них программные интерфейсы. Доступ к ним имеет только администратор средства виртуализации ПО «KeyStack SE», который производит установку и настройку ПО;
- определения правил и процедур управления взаимодействием с внешними информационными системами в организационно-распределительных документах по защите информации;
- предоставления доступа к ПО «KeyStack SE» из внешней информационной системы только авторизованным (уполномоченным) пользователям;
- определения порядка обработки, хранения и передачи информации с использованием внешних информационных систем;
- пересмотра перечня событий безопасности, подлежащих регистрации, не менее чем один раз в год, а также по результатам контроля (мониторинга) за обеспечением уровня защищенности информации, содержащейся в ПО «KeyStack SE»;

- обеспечения срока хранения информации о зарегистрированных событиях безопасности не менее трех месяцев, если иное не установлено требованиями законодательства Российской Федерации, при этом осуществляется хранение записей о выявленных событиях безопасности и записей системных журналов, которые послужили основанием для регистрации события безопасности;
- в процессе функционирования ПО «KeyStack SE» с заданной периодичность осуществляется контроль целостности исполняемых файлов и конфигурации ПО;
- ПО «KeyStack SE» должен контролировать целостность исполняемых и конфигурационных файлов средства виртуализации с использованием программной среды (ОС Platform V SberLinux OS Server).
- определения в ПО «KeyStack SE» источника надежных меток времени (с использованием функции NTP в операционной системе);
- выполнения в ПО «KeyStack SE» синхронизации системного времени с периодичностью, определенной оператором.

6. АВАРИЙНЫЕ СИТУАЦИИ

7.1 Сбои и ошибки эксплуатации

В ходе эксплуатации изделия могут возникнуть сбои и допускаться ошибки эксплуатации.

Сбои в работе ПО «KeyStack SE» могут возникнуть вследствие:

- отказа оборудования;
- сбоев в работе программного обеспечения;
- внесения некорректных изменений в настройки конфигурации программного продукта и реализуемых им функций безопасности;
- нарушений порядка работы с ПО «KeyStack SE» и режима эксплуатации, определенных эксплуатационной документацией на изделие.

7.2 Признаки наличия сбоев

Наличие сбоев и неполадок в работе изделия может быть идентифицировано по следующим признакам:

- сообщения об ошибках (текстовые сообщения), отображаемые в веб-интерфейсе пользователя;
- сообщения средств мониторинга защищаемой сети об отсутствии отклика от ПО «KeyStack SE»;
- программыне сообщения об отсутствии отклика от компонентов ПО «KeyStack SE».

7.3 Действия при наличии сбоев

Порядок проверки сервисов при аварийном переключении (failover)

Failover — это процесс переключения на резервный узел при отказе активного узла.
На узле, где был произведен failover, выполните команду:

```
podman ps -a | grep -v Up
```

Вывод команды должен быть либо пустым, либо содержать только пользовательские контейнеры.

Особое внимание обратить в случае присутствия контейнеров со статусом **Restarting**.

Проверка состояния кластера RabbitMQ

На управляющем узле, где был произведен failover, выполните команду:

```
podman exec -it rabbitmq rabbitmqctl cluster_status
```

В выводе команды должно быть три узла кластера. Все узлы должны быть как в списке nodes.disc, так и в списке running_nodes.

Пример:

```
podman exec -it rabbitmq rabbitmqctl cluster_status
warning: the VM is running with native name encoding of latin1 which may cause Elixir to malfunction as it expects utf8. Please ensure your locale is set to UTF-8 (which can be verified by running "locale" in your shell)
Cluster status of node rabbit@host1 ...
[{nodes,[{disc,['rabbit@host1','rabbit@host2',
'rabbit@host3']}]},
{running_nodes,['rabbit@host13','rabbit@host2',
'rabbit@host3']},
{cluster_name,<<"rabbit@host1">>},
{partitions,[]},
{alarms,[{'host3',[]},
{'rabbit@host2',[]},
{'rabbit@host1',[]}]}]
```

Пайплайн по зачистке RabbitMQ и рестарту сервисов

Пайплайн по зачистке RabbitMQ (RMQ) и рестарту сервисов можно применять в случаях, перечисленных далее (по мониторингу или при соответствующих проверках через CLI).

При постоянном росте на протяжении 5 минут следующих метрик:

- “Messages ready to be delivered to consumers”
- “Messages pending consumer acknowledgement”
- “Unroutable messages dropped”
- “Unacknowledged messages”

При постоянном снижении на протяжении 5 минут следующих метрик:

- “Total queues”
- “Total channels”
- “Total connections”

После применения пайплайна перечисленные выше метрики должны стабилизироваться.

Если вы столкнулись с описанными выше проблемами с RabbitMQ, свяжитесь со службой поддержки.

Проверка состояния кластера MariaDB

Зайдите vault в хранилище секретов соответствующего региона и в passwords_yml найдите “database_password”.

После этого на управляющем узле, пережившем failover, выполните:

```
mysql -uroot  -pPASSWORD -s -s --
execute="SHOW GLOBAL STATUS LIKE 'wsrep_local_state_comment';"
```

где PASSWORD — пароль, прочитанный из Vault.

Пример:

```
ssh control1
mysql -uroot  -pPassW0rd  -s -s --execute="SHOW GLOBAL STATUS LIKE
'wsrep_local_state_comment';"
wsrep_local_state_comment       Synced
```

Состояние узла MariaDB должно быть “Synced”.

На этом же узле необходимо проверить количество активных узлов в кластере MariaDB:

Пример:

```
mysql -uroot  -pPassW0rd -s -s --execute="SHOW GLOBAL STATUS LIKE
'wsrep_cluster_size';"
wsrep_cluster_size      3
```

В кластере Wsrep должно быть 3 активных узла.

Проверка вспомогательных компонентов мониторинга и логирования

На узле, пережившем failover, выполните:

```
podman ps -a | egrep 'opensearch|prometheus|fulentd|grafana'
```

Все контейнеры должны быть в запущенном (“Up”) состоянии.

Проверьте журналы сервисов по пути /var/log/kolla/<service.name>/*.log.

Получить список состояний компонентов сервиса nova в OpenStack

```
openstack compute service list | grep $(HOSTNAME)
```

где $(HOSTNAME) — имя целевого узла, пережившего failover.

Вывод должен показать состояние всех компонентов nova на узле, пережившем failover как “Up”.

Пример:

```
openstack  compute service list | grep control1
|  3 | nova-conductor   | control1   | internal | enabled | up    | 2023-08-08T13:38:00.000000
|
| 33 | nova-scheduler   | control1   | internal | enabled | up    | 2023-08-
08T13:38:03.000000 |
| 42 | nova-consoleauth | control1   | internal | enabled | up    | 2023-08-
08T13:37:57.000000 |
```

Получите список состояний компонентов сервиса Cinder в OpenStack:

```
openstack volume service list | grep $(HOSTNAME)
```

где $(HOSTNAME) — имя целевого узла.

Вывод должен показать состояние всех компонентов Cinder на узле, пережившем failover как “Up”.

Пример:

```
openstack  volume service  list | grep control1
| cinder-scheduler | control1 | nova | enabled | up    | 2023-08-08T13:38:25.000000 |
| cinder-backup    | control1 | nova | enabled | up    | 2023-08-08T13:38:27.000000 |
```

Проверка журналов компонентов neutron на предмет актуальных ошибок

```
ls -t /var/log/kolla/neutron/ | egrep -v 'log..' | grep neutron | xargs -l1 -I {} tail -
n 200 /var/log/kolla/neutron/{} | grep -i error
```

Вывод данной команды должен быть пустым.

Далее нужно зайти по ssh на любой узел с Openstack клиентом и, используя файл admin-openrc.sh, обратиться к API OpenStack, чтобы получить статус компонентов сервиса neutron.

Пример:

```
openstack network agent list | grep network1
| 02480c02-1827-4007-b31a-08c4cb599826 | Metadata agent           | network1    | None
| True  | UP    | neutron-metadata-agent    |
| 67dd92dc-465e-4c28-a52d-9e76b85f66bb | Open vSwitch agent       | network1    |
None            | True  | UP    | neutron-openvswitch-agent |
| a0411328-04d9-46b3-a0a2-0c4e23b8eb06 | L3 agent                 | network1    | nova
| True  | UP    | neutron-l3-agent          |
| bf2b46eb-74a5-40b6-bc40-e0e6fb035dd4 | DHCP agent               | network1    | nova
| True  | UP    | neutron-dhcp-agent        |
```

Failover одного управляющего узла

1. На сервере, пережившем failover, нужно проверить журналы Keystone и MariaDB:

```
grep -P -ri '(error|trace|\=5\d{2})' /var/log/kolla/keystone/\*.log
grep -P -ri '(error|trace|\=5\d{2})' /var/log/kolla/mariadb/\*.log
```

В журналах не должно быть записей 5xx, Traceback, Error.

2. Если ошибки 5xx, Traceback, Error встречаются только в Keystone — перезапустите контейнеры Keystone:

```
podman restart keystone
```

При возникновении проблем в MariaDB требуется понять и устранить причину в логах.

Failover двух управляющих узлов

Если управляющие узлы выключались штатно, то нужно запомнить последовательность выключения серверов и включить их в обратной последовательности с задержкой в 5 минут.

Обычно 5 минут достаточно, чтобы узлы galera добавились в кластер и синхронизировали свое состояние.

При этом крайне желательно после загрузки сервера (в данном примере — controller1) выполнить проверку статуса MariaDB.

Пример:

```
ssh controller2
mysql -uroot  -pPassW0rd  -s -s --execute="SHOW GLOBAL STATUS LIKE
'wsrep_local_state_comment';"
wsrep_local_state_comment       Synced

ssh controller3
mysql -uroot  -pPassW0rd  -s -s --execute="SHOW GLOBAL STATUS LIKE
'wsrep_local_state_comment';"
wsrep_local_state_comment       Synced
```

Failover всех управляющих узлов

1. Восстановите Galera cluster.
2. Откройте страницу `https://<LCM_URL>/project_k/deployments/<REGION>/-/pipelines` → Run Pipeline.

   `KOLLA_ANSIBLE_DEPLOY_ACTION – mariadb_recovery`

Дождитесь выполнения пайплайна и убедитесь, что работа сервисов восстановлена.

3. Перезапустите все контейнеры, кроме MariaDB и RabbitMQ.
4. Перезапустите контейнеры Openstack на узлах Compute.

Failover вычислительного узла

На сервере, пережившем failover, выполните следующие проверки:

1. Проверьте состояние контейнеров nova_compute на вычислительном узле. Выполните команду на вычислительном узле:

   `podman ps -a | grep -i nova_compute`

Убедитесь, что контейнер nova_compute находится в состоянии `Up`.

2. На вычислительном узле проверьте лог `/var/log/kolla/nova/nova-compute.log` на отсутствие событий `ERROR` или `Traceback`.

3. При их наличии убедитесь, что на вычислительном узле нет запущенных ВМ. Для этого на LCM-узле (или любом узле с установленным клиентом OpenStack CLI) выполните команду :

```
openstack server list --all --long | grep $(имя вычислительного узла)
```

4. Если команда вернула информацию о каких-либо виртуальных машинах, тогда следует удалить их в контейнере `nova_libvirt` на вычислительном узле. Для этого для каждой ВМ:

- Получите имя ВМ в среде libvirt, для этого выполните команду:

```
openstack server show ${vmuuid} -c "OS-EXT-SRV-ATTR:instance_name" -f value
```

- Зайдите на вычислительный узел и выполните команду:

```
podman exec nova_libvirt virsh undefine ${vm_name}
```

5. Включите вычислительный узел. Для этого на LCM-узле (или любом узле с установленным клиентом OpenStack CLI) выполните команду:

```
openstack compute service set --enable ${имя_вычислительного_узла} nova-compute
```

Успешное выполнение приведенного выше сценария зависит от следующих факторов:

- Выход вычислительного узла из строя не связан с выходом из строя SDS (Ceph).
- Образы ВМ, работавших на этом вычислительном узле, находятся в консистентном состоянии.
- Образы ВМ, работавших на этом вычислительном узле, корректно эвакуировались с вычислительного узла.

В случае, если приведенные выше действия не привели к устранению сбоев, следует обратиться в службу технической поддержки ПО «KeyStack SE»

Устранение неисправностей при создании/восстановлении резервной копии LCM-сервера

1. Проблема: при попытке доступа к LCM-серверу Nginx возвращает ошибку 302.
Решение: перезапустите контейнер Nginx. Если проблема сохраняется, перезапустите контейнер GitLab командой:

```
podman compose restart gitlab
```

Дождитесь полной загрузки GitLab, затем повторно перезапустите контейнер Nginx.

2. Проблема: Сервис NetBox не запускается.
Решение: возможно, во время создания резервной копии контейнер `netbox-pangolin` не был остановлен.
Признак: в логах контейнера NetBox наблюдаются циклические попытки подключения к базе данных. Проверьте логи контейнера `netbox-pangolin` и выполните диагностику с использованием инструментов Pangolin/PostgreSQL (см. документацию соответствующего ПО).

3. Проблема: Контейнеры не запускаются, появляются ошибки `permission denied`. Решение: Проверьте корректность прав доступа к каталогам в `/installer/data` и при необходимости установите соответствующие разрешения.

Примечания

1. Остановка контейнеров LCM: перед созданием резервной копии или восстановлением убедитесь, что все контейнеры LCM остановлены, чтобы избежать повреждения данных

```
su - root
cd /installer/config
source settings
podman compose down
podman compose -f netbox-compose.yml down
```

2. Скрипт `backupLCM.sh`: убедитесь, что содержимое скрипта соответствует версии используемого инсталлятора.

```
#!/bin/bash

script_path=$(realpath "$0")
script_dir=$(dirname "$script_path")

# check for elevated privileges
if ((${EUID:-0} || "$(id -u)")); then
  echo This script has to be run as root or sudo.
  exit 1
fi

source $script_dir/settings

cd ~ || exit

mkdir -p ~/backupLCM$INSTALL_HOME/data/gitlab/data/backups

# Backup KeyStack config files
cd ~ || exit
cp -pr $INSTALL_HOME/{config,repo}  ~/backupLCM$INSTALL_HOME
cp -pr $INSTALL_HOME/data/{ca,gitlab-runner,vault,netbox,nginx}
~/backupLCM$INSTALL_HOME/data
cp -p ~/.config/containers/auth.json
~/backupLCM$INSTALL_HOME/config/podman_auth.json

# Backup KeyStack Gitlab settings
cp -pr $INSTALL_HOME/data/gitlab/config
~/backupLCM$INSTALL_HOME/data/gitlab/
# fallback:
# BKP_FILE=$(ls -lat /installer/data/gitlab/data/backups/*_gitlab_backup.tar
| head -n 1 | awk '{print $NF}')
podman exec gitlab bundle exec rake --trace gitlab:backup:create
RAILS_ENV=production SKIP=remote,registry BACKUP=dump STRATEGY=copy
cp -pr $INSTALL_HOME/data/gitlab/data/backups
~/backupLCM$INSTALL_HOME/data/gitlab/data/
```

```
cd ~/backupLCM && tar -czpf $BACKUP_HOME/backupLCM-$(date +%d-%m-%Y-
%s).tar.gz * && cd ~
rm -rf ~/backupLCM
rm -f $INSTALL_HOME/data/gitlab/data/backups/*

echo "Upload the backup $BACKUP_HOME/backupLCM.tar.gz to a safe place."
```
3. Скрипт `restoreLCM.sh`: Используйте скрипт, совместимый с версией инсталлятора и резервной копии.
```
#!/bin/bash

DOCKER_COMPOSE_COMMAND='podman compose'

# check os release
os=unknown
[[ -f /etc/os-release ]] && os=$({ . /etc/os-release; echo ${ID,,}; })

script_path=$(realpath "$0")
script_dir=$(dirname "$script_path")

# check for elevated privileges
if ((${EUID:-0} || "$(id -u)")); then
  echo This script has to be run as root or sudo.
  exit 1
fi

if [[ -z "${KS_netbox_admin_password}" ]]; then
  unset netbox_admin_password
  read -rp "Enter the Netbox admin password: " netbox_admin_password
else
  netbox_admin_password=$KS_netbox_admin_password
fi

if [[ -z "${KS_netbox_db_password}" ]]; then
  unset netbox_db_password
  read -rp "Enter the Netbox postgres password: " netbox_db_password
else
  netbox_db_password=$KS_netbox_db_password
fi

if [[ -z "${KS_netbox_redis_password}" ]]; then
  unset netbox_redis_password
  read -rp "Enter the Netbox redis password: " netbox_redis_password
else
  netbox_redis_password=$KS_netbox_redis_password
fi

if [[ -z "${KS_netbox_redis_cache_password}" ]]; then
  unset netbox_redis_cache_password
  read -rp "Enter the Netbox redis cache password: "
netbox_redis_cache_password
else
  netbox_redis_cache_password=$KS_netbox_redis_cache_password
fi

if [[ -z "${KS_Root_Token}" ]]; then
  unset Root_Token
  read -rp "Enter the LCM Vault Initial Root Token: " Root_Token
else
```

```
  Root_Token=$KS_Root_Token
fi

if [[ -z "${KS_Unseal_Key}" ]]; then
  unset Unseal_Key
  read -rp "Enter the LCM Vault Unseal Key 1: " Unseal_Key
else
  Unseal_Key=$KS_Unseal_Key
fi

if [[ -z "${KS_LDAP_BIND_PASSWORD_NETBOX}" ]]; then
  unset LDAP_BIND_PASSWORD_NETBOX
  while [ -z ${LDAP_BIND_PASSWORD_NETBOX} ]; do
    read -rp "Enter the LDAP BIND Password for Netbox: "
LDAP_BIND_PASSWORD_NETBOX
  done
else
  LDAP_BIND_PASSWORD_NETBOX=$KS_LDAP_BIND_PASSWORD_NETBOX
fi

FILE=$(find backupLCM-* -type f)
tar -C / -xpzf $FILE

source /installer/config/settings

############################
# * LCM not Internet * #
############################
# download images and packages
if [ "$os" == "ubuntu" ] && [ -f lcmpackages-ub22034.gz ]; then
  echo "LCM packages exist => untar and install"
  tar -xf lcmpackages-ub22034.gz
  dpkg -i  packages/*.deb
fi

if [ "$os" == "sberlinux" ] && [ -f lcmpackages-sberlinux.gz ]; then
  echo "LCM packages exist => untar and install"
  tar -xf lcmpackages-sberlinux.gz
  yum install -y packages/*rpm
  systemctl start podman
fi

if [ -f nginx-$RELEASE.tar ]; then
  echo "Nginx image exist => loading"
  podman load -i nginx-$RELEASE.tar
  podman tag repo.itkey.com/project_k/lcm/nginx:$RELEASE
$GITLAB_FQDN/project_k/lcm/nginx:$RELEASE
fi

###################################
# SberLinux root cert installation #
###################################
[[ "$os" == "sberlinux" ]] && { cp $CA_HOME/root/ca.crt  /etc/pki/ca-
trust/source/anchors/;  update-ca-trust; }

#################################
# Ubuntu root cert installation #
#################################
```

```
[[ "$os" == "ubuntu" ]] && { cp $CA_HOME/root/ca.crt /usr/local/share/ca-
certificates; update-ca-certificates; }
################################


mkdir -p $INSTALL_HOME/{backup,update} ~/.docker
mv $CFG_HOME/podman_auth.json ~/.config/containers/auth.json
chmod 600 ~/.config/containers/auth.json

sed -i "s|DB_PASSWORD=.*|DB_PASSWORD=$netbox_db_password|"
$NETBOX_HOME/env/netbox.env
sed -i
"s|REDIS_CACHE_PASSWORD=.*|REDIS_CACHE_PASSWORD=$netbox_redis_cache_password|
" $NETBOX_HOME/env/netbox.env
sed -i "s|REDIS_PASSWORD=.*|REDIS_PASSWORD=$netbox_redis_password|"
$NETBOX_HOME/env/netbox.env
sed -i "s|SUPERUSER_PASSWORD=.*|SUPERUSER_PASSWORD=$netbox_admin_password|"
$NETBOX_HOME/env/netbox.env
sed -i "s|POSTGRES_PASSWORD=.*|POSTGRES_PASSWORD=$netbox_db_password|"
$NETBOX_HOME/env/postgres.env
sed -i "s|REDIS_PASSWORD=.*|REDIS_PASSWORD=$netbox_redis_password|"
$NETBOX_HOME/env/redis.env
sed -i "s|REDIS_PASSWORD=.*|REDIS_PASSWORD=$netbox_redis_cache_password|"
$NETBOX_HOME/env/redis-cache.env
sed -i "s|LDAP-BIND-PASSWORD|$LDAP_BIND_PASSWORD_NETBOX|"
$NETBOX_HOME/env/netbox.env
sed -i "/netbox.dump/d" $CFG_HOME/netbox-compose.yml

$DOCKER_COMPOSE_COMMAND -f $CFG_HOME/compose.yaml up -d nginx

# starting the services
$DOCKER_COMPOSE_COMMAND -f $CFG_HOME/compose.yaml up -d
##project_k netbox start
$DOCKER_COMPOSE_COMMAND -f $CFG_HOME/netbox-compose.yml up -d
$DOCKER_COMPOSE_COMMAND -f $CFG_HOME/compose.yaml restart nginx

# check for gitlab readiness
echo -n Waiting for GitLab readiness
while [ "$(podman exec -it gitlab curl -sf http://localhost:8080/-/liveness |
jq -r .status)"  != "ok" ];
do
  echo -n .; sleep 5
done
echo .
sleep 120
echo GitLab is Ready!

podman exec -it gitlab chown -R git /var/opt/gitlab/backups
podman exec -it gitlab supervisorctl stop gitlab:puma
podman exec -it gitlab supervisorctl stop gitlab:gitlab-workhorse
podman exec -it gitlab supervisorctl stop sidekiq
podman exec -it gitlab supervisorctl status
podman exec -it gitlab bundle exec rake gitlab:backup:restore
GITLAB_ASSUME_YES=1 BACKUP=dump force=yes
sleep 30
$DOCKER_COMPOSE_COMMAND -f $CFG_HOME/compose.yaml restart gitlab

echo -n Waiting for GitLab readiness
```

```
while [ "$(podman exec -it gitlab curl -sf http://localhost:8080/-/liveness |
jq -r .status)"  != "ok" ];
do
  echo -n .; sleep 5
done
echo .
sleep 120
echo GitLab is Ready!

$DOCKER_COMPOSE_COMMAND -f $CFG_HOME/compose.yaml restart nginx
podman exec -it gitlab bundle exec rake gitlab:check SANITIZE=true

cat <<END >> /etc/ssh/ssh_config
Host $GITLAB_NAME.$DOMAIN
  PreferredAuthentications publickey
  IdentityFile $CFG_HOME/gitlab_key
END
cat <<END >> /etc/ssh/ssh_config
Host $GITLAB_NAME
  PreferredAuthentications publickey
  IdentityFile $CFG_HOME/gitlab_key
END

echo $(<$CFG_HOME/gitlab_ssh_key) >> /etc/ssh/ssh_known_hosts

# configure Git
git config --system user.email "root@gitlab"
git config --system user.name "ITKey KeyStack"
git config --system --add safe.directory "$REPO_HOME/*"

rm -f $INSTALL_HOME/data/gitlab/data/backups/*

# Vault unseal
podman compose -f $CFG_HOME/compose.yaml exec vault vault operator unseal
$Unseal_Key
podman compose -f $CFG_HOME/compose.yaml exec vault vault login -no-print
$Root_Token


# remove unneeded env variables
unset SAN

printf "\n\n\n################################################################\n"
echo "#                 YOUR INSTALLATION IS READY                   #"
printf "################################################################\n\n\n"

echo "gitlab_runner_token" > $CFG_HOME/gitlab_runner_token
sed -i "s|DB_PASSWORD=.*|DB_PASSWORD=netbox_db_password|"
$NETBOX_HOME/env/netbox.env
sed -i
"s|REDIS_CACHE_PASSWORD=.*|REDIS_CACHE_PASSWORD=netbox_redis_cache_password|"
$NETBOX_HOME/env/netbox.env
sed -i "s|REDIS_PASSWORD=.*|REDIS_PASSWORD=netbox_redis_password|"
$NETBOX_HOME/env/netbox.env
sed -i "s|SUPERUSER_PASSWORD=.*|SUPERUSER_PASSWORD=netbox_admin_password|"
$NETBOX_HOME/env/netbox.env
sed -i "s|AUTH_LDAP_BIND_PASSWORD: .*|AUTH_LDAP_BIND_PASSWORD: \"LDAP-BIND-
PASSWORD\"|" $NETBOX_HOME/env/netbox.env
```

```
sed -i "s|POSTGRES_PASSWORD=.*|POSTGRES_PASSWORD=netbox_db_password|"
$NETBOX_HOME/env/postgres.env
sed -i "s|REDIS_PASSWORD=.*|REDIS_PASSWORD=netbox_redis_password|"
$NETBOX_HOME/env/redis.env
sed -i "s|REDIS_PASSWORD=.*|REDIS_PASSWORD=netbox_redis_cache_password|"
$NETBOX_HOME/env/redis-cache.env
```

РЕКОМЕНДАЦИИ ПО ОСВОЕНИЮ

Для освоения ПО «KeyStack SE» рекомендуем ознакомиться с настоящим документом «Программное обеспечение «Облачная платформа KeyStack SE v.1». Руководство администратора», 1087746411378.62.01.29.000.001 91.

Описание Openstack CLI доступно по ссылке: https://docs.openstack.org/python-openstackclient/latest/index.html

Описание Интеграции с LDAP доступно по ссылке: https://docs.openstack.org/keystone/pike/admin/identity-integrate-with-ldap.html