KeyStack SE

Руководство по установке

Содержание

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 по установке
Версия документа	Версия 1.0
Идентификация ОО	Программное обеспечение «Облачная платформа KeyStack SE v.1». (ПО «KeyStack SE»)

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 УСТАНОВКЕ

3.1 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.

3.1.1 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.

3.1.2 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.

3.1.3 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).

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

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.

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"
```

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.

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.

[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.

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
```

3.1.4 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
```

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.

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.

[6] Дополнить параметрами при необходимости

- 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».

3.1.5 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.

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.

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.

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.

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.

3.1.6 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/</RestURL>
  </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.

3.1.7 Подключение служебных сервисов

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.

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.

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>
```

3.1.8 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 или портал самообслуживания AdminUI создайте несколько ВМ с различными флейворами и выполните их live-миграцию.