Глобус

Защищённая облачная платформа

Руководство по базовой установке

ООО «Лаборатория 50»

e-mail: team@lab50.net

2016

Защищённая облачная платформа «Глобус»: Руководство по базовой установке

Настоящий документ является руководством по разворачиванию защищённой облачной платформы «Глобус» в начальной (тестовой) конфигурации. Тестовая конфигурация включает все основные компоненты и предоставляет необходимый минимум функций платформы. Документ сделан на основе перевода оригинала «OpenStack Basic Installation Guide», предоставляемого организацией OpenStack Foundation.

Дата публикации: 6 апреля 2016 г.

ООО «Лаборатория 50»
198259, г. Санкт-Петербург, ул. Пограничника Гарькавого, дом 36, корп. 1, литер А, пом. 10Н

Содержание

Предисловие

Настоящий документ является руководством по установке и настройке упрощённой конфигурации защищённой облачной платформы «Глобус» для операционной системы Astra Linux Special Edition. Используйте настоящий документ для установки тестовой инфраструктуры. Мы будем производить установку на три узла: узел контроллера, сетевой и вычислительный узлы. Разумеется, вы можете устанавливать столько вычислительных узлов, сколько захотите.

Конфигурационные файлы Глобуса содержат закомментированные параметры. Эти параметры определяют стандартные (штатные) настройки. Если вы хотите изменить стандартные настройки достаточно их раскомментировать. К тому же, те настройки, которые будут приводиться в данном руководстве, будут отличаться от стандартных.

Наконец, имейте ввиду, что «password» в качестве пароля используется в данном руководстве только для простоты и в тестовых целях. Настоятельно рекомендуем использовать надлежащие пароли при настройке ЗОП «Глобус» в рабочем окружении.

Настоящее руководство не содержит инструкций по настройке ряда возможностей облачной платформы, например плавающих IP-адресах.

Принятые обозначения

В данном руководстве принят ряд обозначений.

В документе могут встречаться следующие типы замечаний:

Примечание

Примечание или полезная подсказка.

Важно

Существенные нюансы, которые вы должны принять во внимание, прежде чем начнёте какое-либо действие.

Предупреждение

Важная информация о проблемах безопасности или риске потери данных.

В документе используются два варианта приглашения командного интерпретатора.

$ команда Команды, начинающиеся с приглашения `$`, может выполнять любой пользователь, в том числе `root`.

 # команда Команды, начинающиеся с приглашения `#`, должен выполнять пользователь `root`. Но вы можете использовать утилиту **sudo** чтобы запускать их под другой учётной записью.

Предисловие к переводу

Этот документ создан на основе перевода оригинальной документации, предоставляемой OpenStack Foundation. В процессе перевода мы столкнулись с рядом трудностей, связанных с переводом терминов, принятых в проекте OpenStack. Авторы OpenStack выделяют используемые сущности с помощью специфических терминов, обычно не используемых в области программного обеспечения. Поэтому мы постарались найти такие переводы, которые бы отражали смысл сущности в русском языке и аналогично оригинальной задумке чётко обособляли их.

API	ППИ (прикладной программный интерфейс)
cloud controller	облачный контроллер
compute node	вычислительный узел
credentials	учётные данные
emphemeral	временный (накопитель)
endpoint	точка доступа
flavor	артикул
floating IP	плавающий IP
host aggregate	агрегат
instance	виртуальная машина, экземпляр
keypair	криптопара (ключей), криптоключи
live migration	бесшовная миграция
logging	журналирование
middleware	прослойка
security group	группа безопасности
snapshot	снимок
tenant	проект, участник
volume	том (не путать с накопителем)

Естественно, представленная терминология используется как в документации, так и в программном обеспечении.

История изменений документа

История переиздания

10/03/2015

Описаны необходимые действия по включению мандатной защиты в Менеджере виртуальных машин.

05/02/2015

Использование пакета rabbitmq-server-parsec вместо rabbitmq-server.

23/12/2014

Уточнены настройки синхронизации времени. Настройка сервера NTP на облачном контроллере. Первоначальная синхронизация времени на сетевом узле и сервере виртуализации. Добавлена процедура настройки кэширования в Инструментальной панели.

1 Архитектура

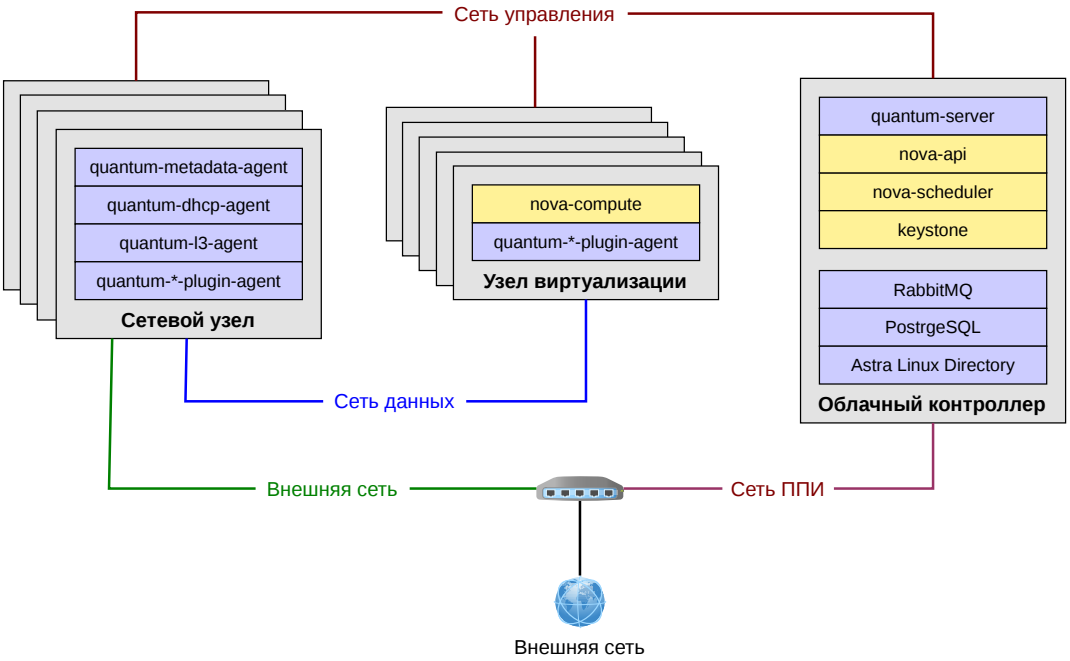

Рис. 1.1: Физическая структура сети

Настоящее руководство описывает создание следующих типовых серверов Глобуса:

Облачный контроллер

Обеспечивает все функциональные возможности облака, за исключением реального управления виртуальными машинами или обеспечения сетевых сервисов. Описания этих функций содержится в главах «Вычислительный узел» и «Сетевой контроллер». На сервере работают: сервис Реестра образов, сервис Оператора блочных устройств, сервис Службы безопасности и Инструментальная панель. На сервере также запускаются элементы Менеджера виртуальных машин (ВМ), а именно: сервер ППИ (прикладной программный интерфейс, API), планировщик, кондуктор, аутентификация для консолей и сервис удалённого рабочего стола (VNC). Наконец, на нем расположены сервис предоставления ППИ (endpoint) Сетевого интегратора Глобуса.

Сетевой контроллер

Обеспечивает большую часть сетевых сервисов Глобуса: DHCP, коммутацию 2-го уровня, маршрутизацию 3-го уровня, плавающие IP (данное руководство не рассматривает их настройку) и сервис метаданных.

Вычислительный узел

Запускает сервис Менеджера виртуальных машин Глобуса и агент Сетевого интегратора (в данном случае — модуль агента Open vSwitch). Этот сервер также управляет совместимым с Глобусом гипервизором (KVM). На этом же сервере будут размещены и имеющиеся виртуальные машины.

Примечание

Глобус очень удобна в управлении отдельными сервисами. Например, сервисы, работающие на сетевом контроллере, можно легко установить на облачный контроллер. Или, например, сервис реестра образов можно установить на свой собственный сервер (или на несколько серверов).

Что касается облачных сетей, стандартная установка сети Глобуса может иметь до четырёх отдельных физических сетей ЦОД. Обратите внимание, что такие сети можно комбинировать и использовать несколько раз. Например, сети «Управление», «Данные» и «ППИ» — это часто одна и та же сеть. Для упрощения, так и будет сделано в данном руководстве.

Сеть управления

Используется для внутренней связи между компонентами Глобуса. Доступ к IP-адресам в этой сети возможен только в центре обработки данных.

Сеть данных

Используется виртуальными машинами для обмена данными в пределах облака. Требования к IP-адресам этой сети зависят от используемого модуля Сетевого интегратора.

Внешняя сеть

Обеспечивает ВМ доступом во внешнюю сеть в некоторых инсталляциях Глобуса. IP-адреса в этой сети должны быть доступны любому пользователю из внешней сети.

Сеть ППИ

Предоставляет участникам все ППИ Глобуса, включая сетевые. IP-адреса этой сети должны быть доступны любому пользователю внешней сети. Это может быть та же сеть, что и внешняя, поскольку можно создать подсеть во внешней сети, использующую выделенный диапазон IP-адресов, меньший, чем полный диапазон адресов в блоке IP.

2 Установка

2.1 Требования

Вам потребуется минимум три виртуальных или реальных машины с установленной ОС Astra Linux Special Edition версии 1.4.

Таблица 2.1. Архитектура и параметры узлов

Параметр	Узлы		
	Облачный контроллер	**Сетевой**	**Вычислительный**
Имя узла Сервисы	cloud PostgreSQL, ALD, RabbitMQ, Nova, Cinder, Glance, Keystone, Quantum	network Quantum-L3-agent, Quantum-DHCP-agent, Quantum Agent с OpenvSwitch	c01 nova-compute, KVM, libvirt, Quantum Agent с OpenvSwitch
Минимальное количество дисков	**2**	1	1
Внешняя сеть Внутренняя сеть	10.0.0.10/24 10.10.10.10/24	10.0.0.9/24 10.10.10.9/24	— 10.10.10.11/24
Общее число сетевых портов	**2**	**2**	1

2.2 Облачный контроллер

2.2.1 Введение

Контроллер обеспечивает следующие сервисы:

— СУБД (PostgreSQL);

— домен со службами Kerberos и LDAP (Astra Linux Directory);

— Сервер очередей AMQP (RabbitMQ);

— Службу безопасности (Keystone);

— Реестр образов (Glance);

— Менеджер виртуальных машин (Nova, без nova-compute);

— Оператор томов (Cinder);

— Сетевой интегратор (Quantum, с плагином OpenvSwitch);

— Инструментальная панель (Horizon).

2.2.2 Общие сервисы

2.2.2.1 Операционная система

1. Установите ОС СН Astra Linux Special Edition. Подробная процедура установки в настоящем руководстве не рассматривается, при установке обратите внимание на следующие настройки:
   — Временная зона: UTC.
   — Имя узла: cloud.
   — Роль: сервер ALD.
   — Пакеты: OpenSSH-Server.

   По завершении установки сервер перезагрузится.
2. Подключите дистрибутив Глобуса в APT. Вы можете использовать компакт-диск Глобуса или скопировать его в общее хранилище. После подключения дистрибутива обновите индексы с помощью команды

   ```
   # aptitude update
   ```

   и установите пакет с ключами дистрибутива Глобуса

   ```
   # aptitude install lab50-archive-keyring
   ```
3. Отключите ограничения на размер файлов (soft, hard) в файле `/etc/security/limits.conf`:

   ```
   # sed -e '/fsize/ s/^#*/#/' -i /etc/security/limits.conf
   ```
4. Выполните настройку сети.
   4.1. Отключите демон wicd:

   ```
   # service wicd stop
   # insserv -r wicd
   ```

   4.2. Настройте сетевые интерфейсы с файле `/etc/network/interfaces`:

   ```
   auto lo
   iface lo inet loopback

   # Внутренняя сеть
   auto eth0
   iface eth0 inet static
       address 10.10.10.10
       netmask 255.255.255.0

   # Внешняя сеть
   auto eth1
   iface eth1 inet static
       address 10.0.0.10
       netmask 255.255.255.0
       gateway 10.0.0.1
       dns-nameservers 10.0.0.1
   ```

   4.3. В файле `/etc/hosts` добавьте имена хостов `cloud`, `network`, и `c01` с правильными IP.

   ```
   127.0.0.1       localhost
   10.10.10.10     cloud.lab50 cloud
   10.10.10.9      network.lab50 network
   10.10.10.11     c01.lab50 c01
   ```

Примечание

Если при работе с простой или тестовой средой записи хостов можно выполнить вручную, то для рабочих инсталляций настоятельно рекомендуется использовать DNS, или, как минимум, использовать средство управления конфигурацией, такое как Ansible.

4.4. Перезагрузитесь.

5. Настройка NTP (сервер протокола времени).

5.1. Установите NTP. NTP будет контролировать точность времени на сервере. Сервис необходим как для компонентов Глобуса, так и для функционирования домена Astra Linux Directory.

```
# aptitude -y install ntp
```

5.2. Облачный контроллер будет серверов NTP. С ним клиенты будут синхронизировать своё время. Для того, чтобы сервер NTP использовал свое время как источник для синхронизации клиентов, добавьте следующие строки в файл конфигурации `/etc/ntp.conf`:

```
server 127.127.1.1
fudge 127.127.1.1 stratum 12
```

5.3. Включите разрешение для запросов с других узлов. Для этого отредактируйте файл настройки NTP `/etc/ntp.conf`:

```
restrict -4 default kod notrap nomodify nopeer
restrict -6 default kod notrap nomodify nopeer
```

5.4. Включите автозапуск демона NTP:

```
# insserv ntp
```

2.2.2.2 Настройка домена Astra Linux Directory

1. Установите пакеты ald-client-common, ald-admin-common, ald-server-common:

```
# aptitude -y install ald-client-common ald-admin-common ald-server-common
```

Примечание

Если вы выбрали роль «сервер ALD» при установке то все необходимые пакеты будут уже установлены.

2. Настройте параметры домена в файле `/etc/ald/ald.conf`:

```
DOMAIN=.lab50
SERVER=cloud.lab50
SERVER_ON=1
CLIENT_ON=1
```

После чего обновите настройки домена и запустите сервисы ALD командой

```
# ald-init init
```

Примечание

Для проверки работы механизмов ALD можно воспользоваться командой test-integrity утилиты ald-admin.

3. Создание пользователей. Через две минуты после инициализации пользователей создайте учётную запись для работы сервисов Глобуса:

```
# ald-admin user-add globus-service
```

И пользователя demo для администрирования Глобуса:

```
# ald-admin user-add demo
```

4. Разрешите пользователю demo заходить на controller:

```
# ald-admin user-ald-cap demo --add-hosts --host cloud.lab50
```

5. Установите утилиту для администрирования Глобуса.

```
# aptitude -y install ald-admin-globus
```

6. Получите билеты администратора ALD (admin/admin) для работы утилиты:

```
# export KRB5CCNAME=/tmp/krb5cc_ald_admin
# kinit admin/admin
```

7. Инициализируйте ALD для работы со Службой безопасности Глобуса:

```
# globus commit-config
```

8. И назначьте пользователя demo администратором:

```
# globus admin-add demo
```

9. Создайте keytab-файл для работы сервисов:

```
# kadmin.local -q 'ktadd -k /root/globus-service.keytab -norandkey globus-service'
```

10. Удалите полученные билеты администратора ALD (admin/admin):

```
# kdestroy
```

2.2.2.3 Сервис базы данных PostgreSQL

Различные компоненты Глобуса хранят постоянные данные в реляционной базе данных. Глобус использует СУБД PostgreSQL, которая входит в состав дистрибутива Astra Linux Special Edition.

1. Установите пакеты:

```
# aptitude -y install python-psycopg2 postgresql-9.2
```

2. В соответствии со стандартными настройками, PostgreSQL принимает запросы только от собственного узла (localhost), на котором работает пользователь. Необходимо изменить настройки таким образом, чтобы узлы Глобуса имели доступ к СУБД. Сервис Мене-

джера ВМ делает запросы к базе данных через сервис `nova-conductor`. В файле `/etc/postgresql/9.2/main/pg_hba.conf` добавьте строку

```
host all all 10.10.10.0/24 md5
```

В файле `/etc/postgresql/9.2/main/postgresql.conf` разрешите подключение к серверу с других узлов с помощью параметра `listen_addresses`:

```
listen_addresses = '*'
```

3. При работе с большим количеством запросов или виртуальных машин могут возникать проблемы с нехваткой количества одновременных соединений к СУБД. Поэтому даже для тестовых инсталляций мы рекомендуем увеличить лимит подключений к PostgreSQL. В файле `/etc/postgresql/9.2/main/postgresql.conf` измените значения параметра `max_connections`:

```
max_connections = 200
```

После этого необходимо также увеличить объем разрешённой разделяемой памяти для процесса. Создайте или отредактируйте файл `/etc/sysctl.d/30-postgresql-shm.conf`:

```
# Maximum size of shared memory segment in bytes
kernel.shmmax = 50331648
# Maximum total size of shared memory in pages (normally 4096 bytes)
kernel.shmall = 12288
```

И введите в действие эти параметры:

```
# sysctl --system
```

4. Перезапустите сервис:

```
# service postgresql restart
```

и включите его автозапуск командой

```
# insserv postgresql
```

5. Необходимо создать различные БД для сервисов Глобуса, а также учётные записи для доступа к ним (в данном руководстве используется пароль «password» для БД):

```
# sudo -u postgres createuser -SRDP glance
# sudo -u postgres createuser -SRDP keystone
# sudo -u postgres createuser -SRDP nova
# sudo -u postgres createuser -SRDP quantum
# sudo -u postgres createuser -SRDP cinder
# sudo -u postgres createdb -E UTF-8 -O glance glance
# sudo -u postgres createdb -E UTF-8 -O keystone keystone
# sudo -u postgres createdb -E UTF-8 -O nova nova
# sudo -u postgres createdb -E UTF-8 -O quantum quantum
# sudo -u postgres createdb -E UTF-8 -O cinder cinder
```

2.2.2.4 Сервис сообщений RabbitMQ

Связь между компонентами Глобуса также обеспечивается с помощью сервиса передачи сообщений AMQP. Например, облачный контроллер размещает в очереди запрос на запуск виртуальной машины. Вычислительный узел получает запрос и запускает машину. Гло-

бус может работать с рядом различных сервисов передачи сообщений, но мы будем использовать RabbitMQ, входящий в дистрибутив Astra Linux Special Edition.

1. Установите сервис сообщений.

Важно

> В связи с наличием ошибки (#56) в скрипте запуска демона RabbitMQ, поставляемого с дистрибутивом Astra Linux Special Edition, просто так поставить этот пакет не удастся. Для корректной установки сервера RabbitMQ используйте пакет rabbitmq-server-parsec из дистрибутива Глобуса.

Установите пакеты RabbitMQ:

```
# aptitude -y install rabbitmq-server-parsec
```

2. Для тестовых инсталляций с небольшим дисковым пространством рекомендуем уменьшить необходимый объем дискового пространства для RabbitMQ. Создайте файл `/etc/rabbitmq/rabbitmq.config`:

```
[
  {rabbit, [{disk_free_limit, 104857600}]}
].
```

3. Включите автозапуск сервиса сообщений:

```
# insserv rabbitmq-server-parsec
```

4. Перезапустите сервис сообщений:

```
# service rabbitmq-server-parsec restart
```

2.2.3 Служба безопасности

Служба безопасности (сервис идентификации) Глобуса обеспечивает облачную инфраструктуру системой авторизации. В этой системе пользователи являются частью одного или нескольких проектов (участников). В каждом участнике у них соответствующие права доступа.

Для настройки службы безопасности выполните следующее:

1. Установите пакеты:

```
# aptitude -y install keystone python-keystoneclient python-globus-keystone
```

2. Зарегистрируйте сервис Службы безопасности в службе ALD:

```
# ald-admin service-add keystone/cloud
```

3. Обновите системный файл ключей Kerberos:

```
# ald-client update-svc-keytab keystone/cloud --ktfile=/etc/keystone/krb5.keytab
# chown keystone:keystone /etc/keystone/krb5.keytab
# chmod 0400 /etc/keystone/krb5.keytab
```

4. Отредактируйте `/etc/keystone/keystone.conf`:

```
[sql]
connection = postgresql://keystone:password@cloud/keystone
```

```
[identity]
driver = globus.keystone.identity.backends.ldap.Identity
default_domain_id = default

[token]
driver = keystone.token.backends.sql.Token

[signing]
token_format = UUID

[ldap]
auth = gssapi
url = ldap://cloud
use_dumb_member = True
dumb_member = cn=dumb,dc=lab50
user_tree_dn = ou=users,dc=lab50
user_objectclass = x-ald-user
user_id_attribute = uid
user_allow_create = False
user_allow_update = False
user_allow_delete = False

tenant_tree_dn = ou=tenants,ou=globus,dc=lab50
tenant_id_attribute = cn
tenant_member_attribute = member
tenant_name_attribute = cn
tenant_attribute_ignore = 'enabled'

role_tree_dn = ou=roles,ou=globus,dc=lab50
role_id_attribute = cn
role_name_attribute = cn

domain_tree_dn = ou=domains,ou=globus,dc=lab50
domain_object_class = organizationalRole
domain_attribute_ignore = 'enabled'

[auth]
methods = password,token
password = keystone.auth.plugins.password.Password
token = keystone.auth.plugins.token.Token

[globus]
admin_group_dn = cn=globus-admin,ou=globus,dc=lab50
globus_service_ou = cn=services,ou=globus,dc=lab50

[filter:krb_auth]
paste.filter_factory = globus.common.auth.middleware:filter_factory
kerberos_service = keystone

[filter:ldap_admin_auth]
paste.filter_factory = globus.keystone.middleware.admin:AdminLdapAuthMiddleware.factory

[filter:debug]
paste.filter_factory = keystone.common.wsgi:Debug.factory

[filter:token_auth]
paste.filter_factory = keystone.middleware:TokenAuthMiddleware.factory

[filter:admin_token_auth]
paste.filter_factory = keystone.middleware:AdminTokenAuthMiddleware.factory

[filter:xml_body]
paste.filter_factory = keystone.middleware:XmlBodyMiddleware.factory

[filter:json_body]
paste.filter_factory = keystone.middleware:JsonBodyMiddleware.factory
```

```
[filter:user_crud_extension]
paste.filter_factory = keystone.contrib.user_crud:CrudExtension.factory

[filter:crud_extension]
paste.filter_factory = keystone.contrib.admin_crud:CrudExtension.factory

[filter:ec2_extension]
paste.filter_factory = keystone.contrib.ec2:Ec2Extension.factory

[filter:s3_extension]
paste.filter_factory = keystone.contrib.s3:S3Extension.factory

[filter:url_normalize]
paste.filter_factory = keystone.middleware:NormalizingFilter.factory

[filter:sizelimit]
paste.filter_factory = keystone.middleware:RequestBodySizeLimiter.factory

[filter:stats_monitoring]
paste.filter_factory = keystone.contrib.stats:StatsMiddleware.factory

[filter:stats_reporting]
paste.filter_factory = keystone.contrib.stats:StatsExtension.factory

[filter:access_log]
paste.filter_factory = keystone.contrib.access:AccessLogMiddleware.factory

[app:public_service]
paste.app_factory = keystone.service:public_app_factory

[app:service_v3]
paste.app_factory = keystone.service:v3_app_factory

[app:admin_service]
paste.app_factory = keystone.service:admin_app_factory

[pipeline:public_api]
pipeline = access_log sizelimit stats_monitoring url_normalize krb_auth token_auth xml_body ldap_admin

[pipeline:admin_api]
pipeline = access_log sizelimit stats_monitoring url_normalize krb_auth token_auth xml_body ldap_admin

[pipeline:api_v3]
pipeline = access_log sizelimit stats_monitoring url_normalize krb_auth token_auth xml_body ldap_admin

[app:public_version_service]
paste.app_factory = keystone.service:public_version_app_factory

[app:admin_version_service]
paste.app_factory = keystone.service:admin_version_app_factory

[pipeline:public_version_api]
pipeline = access_log sizelimit stats_monitoring url_normalize xml_body public_version_service

[pipeline:admin_version_api]
pipeline = access_log sizelimit stats_monitoring url_normalize xml_body admin_version_service

[composite:main]
use = egg:Paste#urlmap
/v2.0 = public_api
/v3 = api_v3
/ = public_version_api

[composite:admin]
use = egg:Paste#urlmap
/v2.0 = admin_api
/v3 = api_v3
```

```
/ = admin_version_api
```

5. Создайте структуру базы данных с помощью команды

```
# keystone-manage db_sync
```

6. Перезапустите сервис:

```
# service keystone restart
```

Примечание

Проверьте файл `/var/log/keystone/keystone.log` на наличие ошибок, которые могут препятствовать успешному запуску сервиса.

7. Войдите в систему с учётной записью demo.
8. Настройте переменные окружения для администрирования:

```
$ export OS_SERVICE_ENDPOINT=http://cloud:35357/v2.0
$ export OS_SERVICE_TOKEN=none
$ export LFT_KRB_KEYSTONE=keystone@cloud
```

9. Заполните службу идентификации начальными данными:

9.1. Зарегистрируйте сервис идентификации доступа:

```
$ keystone service-create --type identity --name 'Служба безопасности'
```

На консоли должен появиться следующий результат, в котором будет обозначен идентификатор созданного сервиса (id):

```
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |                                  |
|      id     | 834d939e41a64c9d9c1f72b72df3ceef |
|     name    |       Служба безопасности        |
|     type    |             identity             |
+-------------+----------------------------------+
```

9.2. Зарегистрируйте точку доступа Службы безопасности:

```
$ keystone endpoint-create --service-id <идентификатор сервиса> \
           --region Lab50 \
           --publicurl http://cloud:5000/v2.0 \
           --adminurl http://cloud:35357/v2.0 \
           --internalurl http://cloud:5000/v2.0
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
|   adminurl  |     http://cloud:35357/v2.0      |
|      id     | 5e601bb773e741e3b33897f665355636 |
| internalurl |      http://cloud:5000/v2.0      |
|  publicurl  |      http://cloud:5000/v2.0      |
|    region   |              Lab50               |
|  service_id | 26987853e2c9487d92e8d6b319b40391 |
+-------------+----------------------------------+
```

В качестве параметра <*идентификатор сервиса*> должен быть использован идентификатор сервиса (id) из шаг 9.1.

9.3. Создайте роли admin и member:

```
$ keystone role-create --name admin
$ keystone role-create --name member
```

9.4. Создайте участников service, project1 и project2:

```
$ keystone tenant-create --name service
$ keystone tenant-create --name project1
$ keystone tenant-create --name project2
```

9.5. Назначьте пользователю demo роль member в project1 и project2:

```
$ keystone user-role-add --role admin --tenant project1 --user demo
$ keystone user-role-add --role member --tenant project2 --user demo
```

2.2.4 Реестр образов

Сервис Реестр образов предоставляет облачной инфраструктуре каталог образов виртуальных машин для запуска.

Например, если реестр содержит образ Astra Linux Special Edition, пользователи облака могут запустить виртуальные машины Astra Linux Special Edition.

Выполните установку и настройку Реестра образов следующим образом.

1. Установите пакеты Реестра образов Глобуса следующим образом:

```
# aptitude -y install glance-api glance-registry
```

2. Настройте аутентификацию с использованием ALD:

2.1. Создайте принципалы ALD для сервисов `glance-api` и `glance-registry`:

```
# ald-admin service-add glance-api/cloud
# ald-admin service-add glance-registry/cloud
```

2.2. Добавьте принципалы сервисов в группу `mac`:

```
# ald-admin sgroup-svc-add glance-api/cloud --sgroup=mac
# ald-admin sgroup-svc-add glance-registry/cloud --sgroup=mac
```

2.3. Добавьте принципалам флаг requires-preauth:

```
# kadmin -p admin/admin -q 'modprinc +requires_preauth glance-api/cloud.lab50'
# kadmin -p admin/admin -q 'modprinc +requires_preauth glance-registry/cloud.lab50'
```

2.4. Добавьте ключи принципалов в файл ключей `/etc/glance/krb5.keytab`:

```
# ald-client update-svc-keytab glance-api/cloud --ktfile=/etc/glance/krb5.keytab
# ald-client update-svc-keytab glance-registry/cloud --ktfile=/etc/glance/krb5.keytab
# chown glance:glance /etc/glance/krb5.keytab
# chmod 0400 /etc/glance/krb5.keytab
```

3. Настройте реестр образов Глобуса следующим образом:

3.1. Реестр образов Глобуса состоит из двух сервисов `glance-api` и `glance-registry`. Для базовой установки они настроены одинаково, однако, они обеспечивают два отдельных сервиса.

Отредактируйте `/etc/glance/glance-api.conf` следующим образом:

```
[DEFAULT]
sql_connection = postgresql://glance:password@cloud/glance
log_file = /var/log/glance/api.log
filesystem_store_datadir = /var/lib/glance/images/

[keystone_authtoken]
admin_tenant_name = service
admin_user = glance-api/cloud
admin_password =
```

```
auth_version = v2.0
auth_host = cloud

[paste_deploy]
flavor = keystone

[globus]
keystone_service = keystone@cloud
kerberos_service = glance-api
glance_registry_service = glance-registry

[kerberos]
keytab = /etc/glance/krb5.keytab
keytab_principal = glance-api/cloud.lab50
cache = /var/lib/glance/krbcc
```

Отредактируйте `/etc/glance/glance-registry.conf` следующим образом:

```
[DEFAULT]
sql_connection = postgresql://glance:password@cloud/glance
log_file = /var/log/glance/registry.log

[keystone_authtoken]
admin_tenant_name = service
admin_user = glance-registry/cloud
admin_password =
auth_version = v2.0
auth_host = cloud

[paste_deploy]
flavor = keystone

[globus]
keystone_service = keystone@cloud
kerberos_service = glance-registry

[kerberos]
keytab = /etc/glance/krb5.keytab
keytab_principal = glance-registry/cloud.lab50
cache = /var/lib/glance/krbcc
```

3.2. Перезапустите оба сервиса:

```
# service glance-api restart && service glance-registry restart
```

Примечание

Проверьте файлы `/var/log/glance/*.log` на наличие ошибок, которые могут препятствовать успешному запуску сервиса реестра образов.

3.3. Инициализируйте базу данных Реестра образов Глобуса:

```
# glance-manage db_sync
```

4. Зарегистрируйте созданный сервис в Службе безопасности:

4.1. Войдите в систему с учётной записью demo и настройте переменные окружения:

```
$ export OS_SERVICE_ENDPOINT=http://cloud:35357/v2.0
$ export OS_SERVICE_TOKEN=none
$ export LFT_KRB_KEYSTONE=keystone@cloud
```

4.2. Зарегистрируйте принципалы glance-api/cloud и glance-registry/cloud в Службе Безопасности:

```
$ globus service-add glance-api/cloud
$ globus service-add glance-registry/cloud
```

4.3. Назначьте пользователям admin в участнике service:

```
$ keystone user-role-add --role admin --tenant service --user glance-api/cloud
$ keystone user-role-add --role admin --tenant service --user glance-registry/cloud
```

4.4. Зарегистрируйте сервис:

```
$ keystone service-create --type image --name 'Реестр образов'
```

На консоли должен появиться следующий результат, в котором будет обозначен идентификатор созданного сервиса (id):

```
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |                                  |
|      id     | 7f108ec2ef4c4502983fb55d09b7ba40 |
|     name    |       Реестр образов             |
|     type    |             image                |
+-------------+----------------------------------+
```

4.5. Зарегистрируйте точку доступа прикладного интерфейса Реестра образов:

```
$ keystone endpoint-create --service-id <идентификатор сервиса> \
           --region Lab50 \
           --publicurl http://cloud:9292/v2.0 \
           --adminurl http://cloud:9292/ \
           --internalurl http://cloud:9292/v2.0
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
|   adminurl  |        http://cloud:9292/        |
|      id     | 54b2a89ba2d04c77a6dd9f261f8535a6 |
| internalurl |      http://cloud:9292/v2.0      |
|  publicurl  |      http://cloud:9292/v2.0      |
|    region   |              Lab50               |
|  service_id | b59d14ffc6b441a0bb74a1646badfe62 |
+-------------+----------------------------------+
```

В качестве параметра *идентификатор сервиса* должен быть использован идентификатор сервиса (id) из шаг 4.4.

5. Залейте образ Astra Linux Special Edition.

5.1. Войдите в систему с учётной записью demo и настройте переменные окружения:

```
$ export OS_AUTH_URL=http://cloud:5000/v2.0
$ export OS_TENANT_NAME=project1
$ export LFT_KRB_KEYSTONE=keystone@cloud
$ export LFT_KRB_GLANCE=glance-api@cloud
```

5.2. Скачайте и импортируйте образ Astra Linux Special Edition:

```
$ glance image-create --name "Astra 1.4" \
         --is-public true --disk-format qcow2 \
         --container-format bare < smolensk-small.img
```

Скачайте и импортируйте образ CirrOS QCOW2:

```
$ glance image-create --name "CirrOS 0.3.3" \
         --is-public true --disk-format qcow2 --container-format bare \
         --copy-from http://download.cirros-cloud.net/0.3.3/cirros-0.3.3-x86_64-disk.img
```

5.3. Убедитесь, что образы появились в индексе:

```
$ glance image-list
+--------------------------------------+--------------+--------+-----------+-----------+--------+
| ID                                   | Name         | Disk   | Container | Size      | Status |
|                                      |              | Format | Format    |           |        |
+--------------------------------------+--------------+--------+-----------+-----------+--------+
| 2ac89c5e-ce81-4376-a48c-1f73575b60a2 | Astra 1.4    | qcow2  | bare      | 1008402432 | active |
| 0d6853a5-3b88-43ae-b637-6513844d9edf | CirrOS 0.3.2 | qcow2  | bare      | 13167616   | active |
+--------------------------------------+--------------+--------+-----------+-----------+--------+
```

2.2.5 Оператор блочных устройств

Несмотря на то, что Оператор блочных устройств содержит много различных драйверов устройств хранения, наиболее употребительной конфигурацией является использование LVM и iSCSI. Данное руководство показывает, как использовать один диск (`/dev/sdb`), в группе томов LVM. Группа томов будет называться `cinder`. Когда пользователь запрашивает новый том, в этой группе создаётся один логический том, который затем монтируется к виртуальной машине пользователя с помощью iSCSI.

Настройка оператора блочных устройств выполняется следующим образом.

1. Отключите правило `udisks-lvm-pv-export` в файле `/lib/udev/rules.d/80-udisks.rules`. Это необходимо для корректной работы компонента cinder-volume. Для этого в файле удалите или закомментируйте строку, содержащую «/lib/udev/udisks-lvm-pv-export»:

   ```
   # sed -e '/udisks-lvm-pv-export/ s/^#*/#/' -i /lib/udev/rules.d/80-udisks.rules
   ```

   и перезагрузите правила udev:

   ```
   # service udev reload
   ```

2. Установите пакеты Оператора блочных устройств:

   ```
   # aptitude -y install cinder-api cinder-scheduler cinder-volume python-cinderclient
   ```

3. Настройте аутентификацию с использованием ALD:

   3.1. Создайте принципал ALD для сервиса `cinder` и для пользователя `cinder`:

   ```
   # ald-admin service-add cinder/cloud
   ```

   3.2. Добавьте принципал сервиса в группу `mac`:

   ```
   # ald-admin sgroup-svc-add cinder/cloud --sgroup=mac
   ```

   3.3. Добавьте принципалам флаг requires-preauth:

   ```
   # kadmin -p admin/admin -q 'modprinc +requires_preauth cinder/cloud.lab50'
   ```

   3.4. Создайте новый файл ключей `/etc/cinder/krb5.keytab` с ключом принципала сервиса `cinder/cloud`:

   ```
   # ald-client update-svc-keytab cinder/cloud --ktfile=/etc/cinder/krb5.keytab
   # chown cinder:cinder /etc/cinder/krb5.keytab
   # chmod 0400 /etc/cinder/krb5.keytab
   ```

4. Настройте Оператор блочных устройств:

   4.1. Отредактируйте `/etc/cinder/cinder.conf`:

   ```
   [DEFAULT]
   ```

```
sql_connection = postgresql://cinder:password@cloud/cinder
sql_max_retries = -1
volumes_dir = /var/lib/cinder/volumes
volume_group = cinder-volumes
iscsi_helper = tgtadm
image_conversion_dir = /var/tmp
lock_path = /run/lock/cinder
auth_strategy = keystone
rabbit_host = cloud
glance_host = cloud
rootwrap_config = /etc/cinder/rootwrap.conf

[keystone_authtoken]
admin_tenant_name = service
admin_user = cinder/cloud
auth_version = v2.0
auth_host = cloud

[globus]
keystone_service = keystone@cloud
kerberos_service = cinder@cloud
glance_service = glance-api@cloud

[kerberos]
keytab = /etc/cinder/krb5.keytab
keytab_principal = cinder/cloud.lab50
cache = /var/lib/cinder/krb5_cc
```

4.2. Создайте физический том и группу LVM на устройстве. Это может быть `/dev/sdb` или `/dev/vdb`, если речь идёт о виртуальной инфраструктуре:

```
# pvcreate /dev/sdb
# vgcreate cinder-volumes /dev/sdb
```

4.3. Создайте таблицы в базе данных Оператора:

```
# cinder-manage db sync
```

4.4. Перезапустите сервисы:

```
# service cinder-api restart
# service cinder-scheduler restart
# service cinder-volume restart
```

5. Зарегистрируйте созданный сервис в Службе безопасности:

5.1. Войдите в систему с учётной записью demo и настройте переменные окружения:

```
$ export OS_SERVICE_ENDPOINT=http://cloud:35357/v2.0
$ export OS_SERVICE_TOKEN=none
$ export LFT_KRB_KEYSTONE=keystone@cloud
```

5.2. Зарегистрируйте принципала как пользователя-сервис в Службе Безопасности:

```
$ globus service-add cinder/cloud
```

5.3. Назначьте пользователю cinder/cloud роль admin в участнике service:

```
$ keystone user-role-add --role admin --tenant service --user cinder/cloud
```

5.4. Зарегистрируйте сервис:

```
$ keystone service-create --type volume --name 'Оператор блочных устройств'
```

На консоли должен появиться следующий результат, в котором будет обозначен идентификатор созданного сервиса (id):

```
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |                                  |
|      id     | 111422a81f0a4408a1b1c08d97fd98ee |
|     name    |    Оператор блочных устройств    |
|     type    |            volume                |
+-------------+----------------------------------+
```

5.5. Зарегистрируйте точку доступа прикладного интерфейса Оператора блочных устройств:

```
$ keystone endpoint-create --service-id <идентификатор сервиса> \
           --region Lab50 \
           --publicurl 'http://cloud:8776/v1/$(tenant_id)s' \
           --adminurl 'http://cloud:8776/v1/$(tenant_id)s' \
           --internalurl 'http://cloud:8776/v1/$(tenant_id)s'
+-------------+-------------------------------------+
|   Property  |               Value                 |
+-------------+-------------------------------------+
|   adminurl  | http://cloud:8776/v1/$(tenant_id)s |
|      id     |  ec4a0e915c084147841402c26365feb8  |
| internalurl | http://cloud:8776/v1/$(tenant_id)s |
|  publicurl  | http://cloud:8776/v1/$(tenant_id)s |
|    region   |                Lab50                |
|  service_id |  e8d24afae89f45a680f73e5ad3f9e8ac  |
+-------------+-------------------------------------+
```

В качестве параметра *идентификатор сервиса* должен быть использован идентификатор сервиса (id) из шаг 5.4.

6. Убедитесь, что сервис функционирует исправно:

6.1. Настройте переменные окружения:

```
$ export OS_AUTH_URL=http://cloud:5000/v2.0
$ export OS_TENANT_NAME=project1
$ export LFT_KRB_KEYSTONE=keystone@cloud
$ export LFT_KRB_CINDER=cinder@cloud
```

6.2. Создайте элементарный том:

```
$ cinder create 1
```

6.3. Убедитесь, что том появились в списке:

```
$ cinder list
+--------------------------------------+-----------+---------+------+--------+----------+----------+
|                  ID                  |   Status  | Display | Size | Volume | Bootable | Attached |
|                                      |           | Name    |      | Type   |          | to       |
+--------------------------------------+-----------+---------+------+--------+----------+----------+
| 8174c841-b4e0-4f85-8961-f5af5990b934 | available |  None   |  1   |  None  |  false   |          |
+--------------------------------------+-----------+---------+------+--------+----------+----------+
```

2.2.6 Сетевой интегратор

Служба Сетевого интегратора обеспечивает облако обширными и расширяемыми сетевыми сервисами. Вот некоторые из них: получение доступа виртуальной машины к внешней сети, за пределами облака; возможность каждому пользователю облака создавать свои собственные многочисленные внутренние подсети. Для настройки Сетевого интегратора выполните ряд операций.

1. Установите сервер Сетевого интегратора:

```
# aptitude -y install quantum-server
```

2. Настройте аутентификацию с использованием ALD:

   2.1. Создайте принципала ALD для сервиса `quantum` и для пользователя `quantum`:

```
# ald-admin service-add quantum/cloud
# ald-admin user-add --home-type=local quantum
```

   2.2. Добавьте принципал сервиса в группу `mac`:

```
# ald-admin sgroup-svc-add quantum/cloud --sgroup=mac
```

   2.3. Добавьте принципалам флаг requires-preauth:

```
# kadmin -p admin/admin -q 'modprinc +requires_preauth quantum/cloud.lab50'
```

   2.4. Создайте новый файл ключей `/etc/quantum/krb5.keytab` с ключом принципала `quantum/cloud`:

```
# ald-client update-svc-keytab quantum/cloud --ktfile=/etc/quantum/krb5.keytab
# chown quantum:quantum /etc/quantum/krb5.keytab
# chmod 0400 /etc/quantum/krb5.keytab
```

3. Выполните настройку сервиса.

   3.1. Укажите файл настроек используемого сетевого модуля в файле `/etc/default/quantum-server`:

```
QUANTUM_PLUGIN_CONFIG=/etc/quantum/plugins/openvswitch/ovs_quantum_plugin.ini
```

   3.2. Отредактируйте `/etc/quantum/quantum.conf`:

```
[DEFAULT]
core_plugin = quantum.plugins.openvswitch.ovs_quantum_plugin.OVSQuantumPluginV2

[keystone_authtoken]
admin_tenant_name = service
admin_user = quantum/cloud
auth_version = v2.0

[globus]
keystone_service = keystone@cloud
kerberos_service = quantum@cloud

[kerberos]
keytab = /etc/quantum/krb5.keytab
keytab_principal = quantum/cloud.lab50
cache = /var/lib/quantum/krb5_cc
```

   3.3. Отредактируйте `/etc/quantum/plugins/openvswitch/ovs_quantum_plugin.ini`:

```
[DATABASE]
sql_connection = postgres://quantum:password@cloud/quantum
sql_max_retries = -1

[OVS]
tenant_network_type = gre
tunnel_id_ranges = 1:1000
enable_tunneling = True
local_ip = 10.10.10.10
integration_bridge = br-int

[SECURITYGROUP]
firewall_driver = \
  quantum.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver
```

Примечание

Проще выбрать изоляцию сетей с помощью GRE (туннелирование) т.к. вам не придётся настраивать сетевое оборудование для работы VLAN.

4. Перезапустите сервис:

```
# service quantum-server restart
```

5. Зарегистрируйте созданный сервис в Службе безопасности:

5.1. Войдите в систему с учётной записью demo и настройте переменные окружения:

```
$ export OS_SERVICE_ENDPOINT=http://cloud:35357/v2.0
$ export OS_SERVICE_TOKEN=none
$ export LFT_KRB_KEYSTONE=keystone@cloud
```

5.2. Зарегистрируйте принципа quantum/cloud как пользователя-сервис в Службе Безопасности:

```
$ globus service-add quantum/cloud
```

5.3. Назначьте пользователю quantum/cloud роль admin в участнике service:

```
$ keystone user-role-add --role admin --tenant service --user quantum/cloud
```

5.4. Зарегистрируйте сервис:

```
$ keystone service-create --type network --name 'Сетевой интегратор'
```

На консоли должен появиться следующий результат, в котором будет обозначен идентификатор созданного сервиса (id):

```
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |                                  |
|      id     | adc4b6eabf5747fc8bfd19418b37c19b |
|     name    |        Сетевой интегратор        |
|     type    |             network              |
+-------------+----------------------------------+
```

5.5. Зарегистрируйте точку доступа прикладного интерфейса Сетевого интегратора:

```
$ keystone endpoint-create --service-id <идентификатор сервиса> \
           --region Lab50 \
           --publicurl 'http://cloud:9696/' \
           --adminurl 'http://cloud:9696/' \
           --internalurl 'http://cloud:9696/'
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
|   adminurl  |        http://cloud:9696/        |
|      id     | bf5d1eb33e524c819084a6ef65aa36a4 |
| internalurl |        http://cloud:9696/        |
|  publicurl  |        http://cloud:9696/        |
|    region   |              Lab50               |
|  service_id | ed6ee3784fac4ef495d29ba4e6201f8a |
+-------------+----------------------------------+
```

В качестве параметра <*идентификатор сервиса*> должен быть использован идентификатор сервиса (id) из шаг 5.4.

6. Убедитесь, что сервис функционирует исправно:

6.1. Настройте переменные окружения:

```
$ export OS_AUTH_URL=http://cloud:5000/v2.0
$ export OS_TENANT_NAME=project1
$ export LFT_KRB_KEYSTONE=keystone@cloud
$ export LFT_KRB_QUANTUM=quantum@cloud
```

6.2. Запросите список доступных модулей Сетевого интегратора:

```
$ quantum ext-list
+-----------------+-------------------+---------------------+-----------------------------------------------------------+--------
| alias           | description       | name                | namespace                                                 | updated
+-----------------+-------------------+---------------------+-----------------------------------------------------------+--------
| agent_scheduler | Schedule...       | Agent Schedulers    | http://docs.openstack.org/ext/agent_scheduler/api/v1.0    | 2013-02
| security-group  | The security...   | security-group      | http://docs.openstack.org/ext/securitygroups/api/v2.0     | 2012-10
| binding         | Expose port...    | Port Binding        | http://docs.openstack.org/ext/binding/api/v1.0            | 2012-11
| quotas          | Expose...         | Quota managementrt  | http://docs.openstack.org/network/ext/quotas-sets/api/v2.0| 2012-07
| agent           | The agent...      | agent               | http://docs.openstack.org/ext/agent/api/v2.0              | 2013-02
| provider        | Expose mapping... | Provider Network    | http://docs.openstack.org/ext/provider/api/v1.0           | 2012-09
| router          | Router...         | Quantum L3 Router   | http://docs.openstack.org/ext/quantum/router/api/v1.0     | 2012-07
| extraroute      | Extra routes...   | Quantum Extra Route | http://docs.openstack.org/ext/quantum/extraroutes/api/v1.0| 2013-02
+-----------------+-------------------+---------------------+-----------------------------------------------------------+--------
```

2.2.7 Менеджер виртуальных машин

Менеджер виртуальных машин предоставляет облачной среде возможность управлять расписанием, созданием и удалением ВМ.

Для настройки сервиса выполните следующие операции.

1. Установите пакеты для облачного контроллера:

```
# aptitude -y install python-globus-keystone python-globus-nova \
                      nova-api nova-conductor nova-consoleauth nova-console \
                      nova-novncproxy nova-scheduler nova-spiceproxy
```

2. Настройте аутентификацию с использованием ALD:

2.1. Создайте принципала ALD для сервиса `nova`:

```
# ald-admin service-add nova/cloud
```

2.2. Добавьте принципал в группу `mac`:

```
# ald-admin sgroup-svc-add nova/cloud --sgroup=mac
```

2.3. Добавьте принципалам флаг requires-preauth:

```
# kadmin -p admin/admin -q 'modprinc +requires_preauth nova/cloud.lab50'
```

2.4. Создайте новый файл ключей `/etc/nova/krb5.keytab` с ключом принципала сервиса `nova/cloud`:

```
# ald-client update-svc-keytab nova/cloud --ktfile=/etc/nova/krb5.keytab
# chown nova:nova /etc/nova/krb5.keytab
# chmod 0400 /etc/nova/krb5.keytab
```

3. Настройте Менеджер виртуальных машин.

3.1. Установите необходимые параметры в файле `/etc/nova/nova.conf`. nova.conf является главным файлом конфигурации сервиса. В этом файле имеется большое количество настроек. В данном руководстве рассматривается минимально необходимые для построения простой среды. Обратите внимание, что в файле nova.conf, при установке уже указаны некоторые значения. Оставьте их, как есть.

```
[DEFAULT]
auth_strategy = keystone
lock_path = /var/lock/nova
```

```
state_path = /var/lib/nova

compute_driver = globus.compute.driver.libvirt_driver.LibvirtParsecDriver
compute_manager = globus.compute.manager.ComputeManager
sql_connection = postgresql://nova:password@cloud/nova
rabbit_hosts = cloud:5672
glance_api_servers = cloud:9292
vnc_enabled = True
vncserver_proxyclient_address = cloud
novncproxy_host = cloud
novncproxy_port = 6080
novncproxy_base_url = http://cloud:6080/vnc_auto.html
vncserver_listen = 0.0.0.0

iscsi_helper = tgtadm

allow_resize_to_same_host = True

network_api_class = nova.network.quantumv2.api.API
quantum_url = http://cloud:9696/
quantum_auth_strategy = keystone
quantum_admin_tenant_name = service
quantum_admin_username = nova/cloud
quantum_admin_auth_url = http://cloud:35357/v2.0
security_group_api = quantum
service_quantum_metadata_proxy = true

firewall_driver = nova.virt.firewall.NoopFirewallDriver
quantum_metadata_proxy_shared_secret =

volume_api_class = nova.volume.cinder.API

[keystone_authtoken]
admin_password =
admin_tenant_name = service
admin_user = nova/cloud
auth_host = cloud
auth_port = 35357
auth_protocol = http
auth_version = v2.0

[globus]
kerberos_service = nova@cloud
cinder_service = cinder@cloud
glance_service = glance-api@cloud
keystone_service = keystone@cloud
quantum_service = quantum@cloud

[kerberos]
keytab = /etc/nova/krb5.keytab
keytab_principal = nova/cloud.lab50
cache = $state_path/krb5_cc
```

3.2. Создайте структуру базы данных с помощью команды:

```
# nova-manage db sync
```

3.3. Перезапустите необходимые демоны:

```
# service nova-api restart
# service nova-console restart
# service nova-consoleauth restart
# service nova-scheduler restart
# service nova-conductor restart
# service nova-novncproxy restart
```

Примечание

Проверьте файлы `/var/log/nova/nova-*` на наличие ошибок, которые могут препятствовать успешному запуску сервиса управления ВМ.

4. Зарегистрируйте созданный сервис в Службе безопасности:

   4.1. Войдите в систему с учётной записью demo и настройте переменные окружения:

```
$ export OS_SERVICE_ENDPOINT=http://cloud:35357/v2.0
$ export OS_SERVICE_TOKEN=none
$ export LFT_KRB_KEYSTONE=keystone@cloud
```

   4.2. Зарегистрируйте принципала как пользователя-сервис в Службе Безопасности:

```
$ globus service-add nova/cloud
```

   4.3. Назначьте пользователю nova/cloud роль admin в участнике service:

```
$ keystone user-role-add --role admin --tenant service --user nova/cloud
```

   4.4. Зарегистрируйте сервис:

```
$ keystone service-create --type compute --name 'Менеджер ВМ'
```

   На консоли должен появиться следующий результат, в котором будет обозначен идентификатор созданного сервиса (id):

```
+-------------+----------------------------------+
|   Property  |              Value               |
+-------------+----------------------------------+
| description |                                  |
|      id     | 293a3fda12d84be9a8c67828e518541d |
|     name    |           Менеджер ВМ            |
|     type    |             compute              |
+-------------+----------------------------------+
```

   4.5. Зарегистрируйте точку доступа прикладного интерфейса Менеджера виртуальных машин:

```
$ keystone endpoint-create --service-id <идентификатор сервиса> \
           --region Lab50 \
           --publicurl 'http://cloud:$(compute_port)s/v2/$(tenant_id)s' \
           --adminurl 'http://cloud:$(compute_port)s/v2/$(tenant_id)s' \
           --internalurl 'http://cloud:$(compute_port)s/v2/$(tenant_id)s'
+-------------+-------------------------------------------------+
|   Property  |                     Value                       |
+-------------+-------------------------------------------------+
|   adminurl  | http://cloud:$(compute_port)s/v2/$(tenant_id)s |
|      id     |        bf690522e92b407a906af88b667b9fa8         |
| internalurl | http://cloud:$(compute_port)s/v2/$(tenant_id)s |
|  publicurl  | http://cloud:$(compute_port)s/v2/$(tenant_id)s |
|    region   |                     Lab50                       |
|  service_id |        30dac5ce2ad14e44940843cfe9ad181a         |
+-------------+-------------------------------------------------+
```

   В качестве параметра <*идентификатор сервиса*> должен быть использован идентификатор сервиса (id) из шаг 4.4.

5. Убедитесь, что сервис функционирует исправно:

   5.1. Настройте переменные окружения:

```
$ export OS_AUTH_URL=http://cloud:5000/v2.0
$ export OS_TENANT_NAME=project1
$ export LFT_KRB_KEYSTONE=keystone@cloud
$ export LFT_KRB_NOVA=nova@cloud
```

5.2. Запросите список сервисов Nova:

```
$ nova service-list
+------------------+-------+----------+---------+-------+----------------------------+
| Binary           | Host  | Zone     | Status  | State | Updated_at                 |
+------------------+-------+----------+---------+-------+----------------------------+
| nova-conductor   | cloud | internal | enabled | up    | 2014-08-21T07:57:24.731150 |
| nova-console     | cloud | internal | enabled | up    | 2014-08-21T07:57:21.949159 |
| nova-consoleauth | cloud | internal | enabled | up    | 2014-08-21T07:57:19.494618 |
| nova-scheduler   | cloud | internal | enabled | up    | 2014-08-21T07:57:22.537319 |
+------------------+-------+----------+---------+-------+----------------------------+
```

2.2.8 Инструментальная панель

Инструментальная панель управления предоставляет пользователям облачной среды графический веб-интерфейс, вместо использования команд в командной строке. Для настройки панели управления выполните следующие шаги.

1. Для работы инструментальной панели используется защищённый веб-сервер Apache, входящий в состав Astra Linux Special Edition. Настройте его в соответствии с руководством администратора (РУСБ.10015-01 95 01):

   1.1. Установите необходимые пакеты:

   ```
   # aptitude -y install apache2 libapache2-mod-auth-kerb libapache2-mod-wsgi
   ```

   1.2. Создайте принципала ALD `HTTP`:

   ```
   # ald-admin service-add HTTP/cloud
   ```

   1.3. Добавьте принципал в группу `mac`:

   ```
   # ald-admin sgroup-svc-add HTTP/cloud --sgroup=mac
   ```

   1.4. Добавьте ключи сервисов в файл ключей `/etc/apache/krb.keytab`:

   ```
   # ald-client update-svc-keytab HTTP/cloud --ktfile=/etc/apache2/krb.keytab
   ```

   1.5. Установите на файл ключей правильные права доступа:

   ```
   # chown www-data:www-data /etc/apache2/krb.keytab
   # chmod 0644 /etc/apache2/krb.keytab
   ```

2. Чтобы запустить инструментальную панель, установите пакеты и выполните ряд настроек.

   2.1. Установите необходимые пакеты:

   ```
   # aptitude -y install openstack-dashboard-apache openstack-dashboard-globus-theme
   ```

   2.2. Настройте следующие переменные панели управления в файле `/etc/openstack-dashboard/local_settings.py`:

   ```
   OPENSTACK_HOST = "cloud"
   ```

   2.3. Перезагрузите веб-браузер:

   ```
   # service apache2 reload
   ```

3. Отзывчивость инструментальной панели можно серьёзно ускорить, если настроить кэширование данных. В качестве хранилища кэша доступны различные варианты. В данном руководстве используется PostgreSQL.

   3.1. Если вы планируете запускать виртуальные машины с ненулевой мандатной меткой, необходимо правильно настроить PostgreSQL. Процедура настройки приведена в «Руководстве администратора ОС СН Astra Linux Special Edition (см. подраздел 6.8.1.1). В настоящем руководстве приведём краткую выжимку необходимых команд:

```
# ald-admin service-add postgres/cloud.lab50
# ald-admin sgroup-svc-add postgres/cloud.lab50 --sgroup=mac
# ald-client update-svc-keytab postgres/cloud.lab50 --ktfile=/etc/postgresql/9.2/main/krb5.keytab
# chown postgres:postgres /etc/postgresql/9.2/main/krb5.keytab
```

   После выполнения команд отредактируйте конфигурационные файлы:

   `/etc/postgresql/9.2/main/postgresql.conf`:

```
krb_server_keyfile = '/etc/postgresql/9.2/main/krb5.keytab'
krb_srvname = 'postgres'
```

   `/etc/postgresql/9.2/main/pg_hba.conf`:

```
host horizon all cloud.lab50 gss
```

   3.2. Создайте пользователя `horizon`:

```
# useradd -r horizon
```

   3.3. Создайте пользователя в СУБД и базу данных для хранения кэша:

```
# sudo -u postgres createuser -SRDP horizon
# sudo -u postgres createdb -E UTF-8 -T template0 -O horizon horizon
```

   3.4. Настройте параметры кэширования в файле конфигурации веб-приложения `/etc/openstack-dashboard/local_settings.py`:

```
CACHES = {
    'default': {
        'BACKEND': 'django.core.cache.backends.db.DatabaseCache',
        'LOCATION': 'djangocache',
    }
}

SESSION_ENGINE = 'django.contrib.sessions.backends.cached_db'

DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.postgresql_psycopg2',
        'NAME': 'horizon',
        'USER': 'horizon',
        'PASSWORD': 'password', # пароль пользователя horizon в СУБД
        'HOST': 'cloud',
        'default-character-set': 'utf8'
    }
}
```

   3.5. Инициализируйте БД кэширования:

```
# yes no | /usr/share/openstack-dashboard/manage.py syncdb
# /usr/share/openstack-dashboard/manage.py createcachetable djangocache
```

   3.6. Перезагрузите веб-приложение:

```
# service apache2 reload
```

Примечание

Для корректной работы механизмов кэширования необходимо убедиться в правильной настройке часового пояса. Часовой пояс должен совпадать на всех машинах облачной инфраструктуры, включая клиенты. Но это ещё не всё. В соответствии с используемым системным часовым поясом должен быть установлен часовой пояс в веб-приложении (см. параметр `TIME_ZONE` в файле конфигурации `/etc/openstack-dashboard/local_settings.py`).

4. Инструментальная панель будет доступна по адресу http://cloud. Вы должны использовать учётную запись соответствующего пользователя для доступа к панели.

Примечание

Проверьте файл `/var/log/apache2/error.log` `/var/log/httpd/error_log` на наличие ошибок, которые могут препятствовать успешному запуску сервиса Apache или сервиса инструментальной панели.

5. Убедитесь, что панель функционирует исправно:
   5.1. Запустите веб-браузер. Зайдите в его настройки (введите в адресной строке адрес `about:config`) и отредактируйте два параметра `network.negotiate-auth.delegation-uris` и `network.negotiate-auth.trusted-uris`, поле `Value` должно иметь значение `http://`.
   5.2. Перейдите по адресу `http://cloud`.

2.3 Сетевой узел

2.3.1 Введение

Сетевой узел будет предоставлять следующие функции:

— программную коммутацию (OpenvSwitch + агент Сетевого интегратора) с туннелированием;

— сервер DHCP (агент DHCP);

— виртуальную маршрутизацию (агент L3).

Примечание

Все эти сервисы можно установить на облачный контроллер, если вы ограничены в ресурсах.

2.3.2 Общие сервисы

2.3.2.1 Операционная система

1. Установите ОС СН Astra Linux Special Edition. Подробная процедура установки в настоящем руководстве не рассматривается, при установке обратите внимание на следующие настройки:
   — Временная зона: UTC.
   — Имя узла: network.
   — Роль: клиент ALD.
   — Пакеты: OpenSSH-Server.

   По завершении установки сервер перезагрузится.
2. Подключите дистрибутив Глобуса в APT. Вы можете использовать компакт-диск Глобуса или скопировать его на общее хранилище. После подключения дистрибутива обновите индексы с помощью команды

   ```
   # aptitude update
   ```

   и установите пакет с ключами дистрибутива Глобуса:

   ```
   # aptitude -y install spkb-archive-keyring
   ```
3. Выполните настройку сети.
   3.1. Отключите демон wicd:

   ```
   # service wicd stop
   # insserv -r wicd
   ```

   3.2. Установите пакеты программного коммутатора Open vSwitch:

   ```
   # aptitude -y install openvswitch-switch \
                         openvswitch-datapath-module-3.2.0-27-generic
   ```

   3.3. Включите автозапуск демона openvswitch-switch:

   ```
   # insserv -r openvswitch-switch && insserv openvswitch-switch
   ```

   3.4. Создайте сетевые мосты Open vSwitch:

   ```
   # ovs-vsctl -- --may-exist add-br br-int
   # ovs-vsctl -- --may-exist add-br br-ext
   # ovs-vsctl -- --may-exist add-port br-ext eth1
   ```

   3.5. Отредактируйте `/etc/network/interfaces`:

   ```
   auto lo
   iface lo inet loopback

   auto eth0
   iface eth0 inet static
       address 10.10.10.9
       netmask 255.255.255.0

   auto eth1
   iface eth1 inet manual
       pre-up ifconfig $IFACE up
       post-down ifconfig $IFACE down

   allow-ovs br-int
   ```

```
iface br-int inet manual

allow-ovs br-ext
iface br-ext inet static
    pre-up ifup --allow=$IFACE eth1
    address 10.0.0.9
    netmask 255.255.255.0
    gateway 10.0.0.1
    dns-nameservers 10.0.0.1
```

3.6. Добавьте следующие параметры в файл `/etc/sysctl.conf`:

```
net.ipv4.ip_forward = 1
net.ipv4.conf.all.forwarding = 1
```

3.7. Перезагрузитесь.

3.8. Настройте внешний мост командой:

```
# ovs-vsctl --no-wait br-set-external-id br-ext bridge-id br-ext
```

3.9. В файле `/etc/hosts` добавьте имена хостов `cloud`, `network`, и `c01` с правильными IP.

```
127.0.0.1       localhost
10.10.10.10     cloud.lab50 cloud
10.10.10.9      network.lab50 network
10.10.10.11     c01.lab50 c01
```

Примечание

Если при работе с простой или тестовой средой записи хостов можно выполнить вручную, то для рабочих инсталляций настоятельно рекомендуется использовать DNS, или, как минимум, использовать средство управления конфигурацией, такое как Ansible.

4. Настройте синхронизацию времени в домене. Установите NTP (сервер протокола времени). Сервис необходим как для компонентов Глобуса, так и для функционирования домена Astra Linux Directory.

4.1. Установите необходимый пакет:

```
# aptitude -y install ntp ntpdate
```

4.2. Настройте демон NTP для синхронизации с сервером на узле cloud. Для этого добавьте строку в файл `/etc/ntp.conf`:

```
server cloud
```

4.3. Включите автозапуск демона NTP:

```
# insserv ntp
```

4.4. Перезапустите демон NTP:

```
# service ntp restart
```

4.5. Произведите первоначальную синхронизацию времени:

```
# ntpdate -u cloud
```

5. Установите и настройте домен *Astra Linux Directory*.

5.1. Установите пакеты:

```
# aptitude -y install ald-client-common ald-admin ald-admin-parsec-mac
```

5.2. Настройте параметры домена в файле `/etc/ald/ald.conf`:

```
DOMAIN=.lab50
SERVER=cloud.lab50
SERVER_ON=0
CLIENT_ON=1
```

5.3. Введите узел в домен ALD:

```
# ald-client commit-config
```

2.3.3 Сетевой интегратор

2.3.3.1 Настройка модуля Open vSwitch

Для настройки Open vSwitch:

1. Установите пакеты:

```
# aptitude -y install quantum-plugin-openvswitch-agent quantum-dhcp-agent \
    quantum-l3-agent quantum-metadata-agent
```

2. Включите простой сервис NAT (транслятор сетевых адресов), чтобы вычислительный узлы имели доступ во внешнюю сеть через облачный контроллер:

```
# iptables -A FORWARD -i eth0 -o br-ex -s 10.10.10.0/24 -m conntrack \
          --ctstate NEW -j ACCEPT
# iptables -A FORWARD -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
# iptables -A POSTROUTING -s 10.10.10.0/24 -t nat -j MASQUERADE
```

Примечание

Эти настройки будут потеряны при перезагрузке. Чтобы сохранить или восстановить эти настройки, используйте специальные инструменты.

2.3.3.2 Сетевые сервисы

Настройка сетевых агентов и модулей Сетевого интегратора.

1. Настройте аутентификацию с использованием ALD:

1.1. Создайте принципала ALD для сервиса `quantum`:

```
# ald-admin service-add quantum/network
```

1.2. Добавьте принципал в группу `mac`:

```
# ald-admin sgroup-svc-add quantum/network --sgroup=mac
```

1.3. Добавьте принципалам флаг requires-preauth:

```
# kadmin -p admin/admin -q 'modprinc +requires_preauth quantum/network.lab50'
```

1.4. Создайте новый файл ключей `/etc/quantum/krb5.keytab` с ключом принципала сервиса `quantum/network`:

```
# ald-client update-svc-keytab quantum/network --ktfile=/etc/quantum/krb5.keytab
# chown quantum:quantum /etc/quantum/krb5.keytab
# chmod 0400 /etc/quantum/krb5.keytab
```

1.5. Войдите в систему с учётной записью demo и настройте переменные окружения:

```
$ export OS_SERVICE_ENDPOINT=http://cloud:35357/v2.0
$ export OS_SERVICE_TOKEN=none
$ export LFT_KRB_KEYSTONE=keystone@cloud
```

1.6. Зарегистрируйте принципала как пользователя-сервис в Службе Безопасности:

```
$ globus service-add quantum/network
```

1.7. Назначьте пользователю quantum/network роль admin в участнике service:

```
$ keystone user-role-add --role admin --tenant service --user quantum/network
```

2. Отредактируйте `/etc/quantum/quantum.conf` следующим образом:

```
[DEFAULT]
core_plugin = quantum.plugins.openvswitch.ovs_quantum_plugin.OVSQuantumPluginV2
rabbit_hosts = cloud

[AGENT]
root_helper = sudo quantum-rootwrap /etc/quantum/rootwrap.conf

[keystone_authtoken]
admin_tenant_name = service
admin_user = quantum/network
auth_version = v2.0

[globus]
keystone_service = keystone@cloud
kerberos_service = quantum@cloud
```

3. Отредактируйте `/etc/quantum/plugins/openvswitch/ovs_quantum_plugin.ini` следующим образом:

```
[DATABASE]
sql_connection = postgres://quantum:password@cloud/quantum
sql_max_retries = -1

[OVS]
tenant_network_type = gre
tunnel_id_ranges = 1:1000
enable_tunneling = True
local_ip = 10.10.10.9
intergration_bridge = br-int

[SECURITYGROUP]
firewall_driver = quantum.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver
```

4. Отредактируйте `/etc/quantum/dhcp_agent.ini` следующим образом:

```
[DEFAULT]
interface_driver = quantum.agent.linux.interface.OVSInterfaceDriver
use_namespaces = True
enable_isolated_metadata = True
enable_metadata_network = True
```

5. Отредактируйте `/etc/quantum/l3_agent.ini` следующим образом:

```
[DEFAULT]
interface_driver = quantum.agent.linux.interface.OVSInterfaceDriver
use_namespaces = True
```

```
external_network_bridge = br-ext
```

6. Отредактируйте `/etc/quantum/metadata_agent.ini` следующим образом:

```
[DEFAULT]
auth_url = http://10.10.10.10:35357/v2.0
auth_region = Lab50
auth_version = v2.0
admin_tenant_name = service
admin_user = quantum/network
admin_password =
nova_metadata_ip = 10.10.10.10
metadata_proxy_shared_secret =

[kerberos]
keytab = /etc/quantum/krb5.keytab
keytab_principal = quantum/network.lab50
cache = /var/lib/quantum/krb5_cc
```

7. Перезапустите агенты и модули следующим образом:

```
# service quantum-plugin-openvswitch-agent start
# service quantum-dhcp-agent restart
# service quantum-metadata-agent restart
# service quantum-l3-agent restart
```

Примечание

Проверьте файлы `/var/log/quantum/*.log` на наличие ошибок, которые могут препятствовать успешному запуску сервиса Сетевого интегратора.

8. Убедитесь, что сервис функционирует исправно:

   8.1. Зайдите на узел облачного контроллера с учётной записью demo. Настройте переменные окружения:

```
$ export OS_AUTH_URL=http://cloud:5000/v2.0
$ export OS_TENANT_NAME=project1
$ export LFT_KRB_KEYSTONE=keystone@cloud
$ export LFT_KRB_QUANTUM=quantum@cloud
```

   8.2. Запросите список агентов:

```
$ quantum agent-list
+--------------------------------------+--------------------+--------------+-------+----------+
| id                                   | agent_type         | host         | alive | admin_   |
|                                      |                    |              |       | state_up |
+--------------------------------------+--------------------+--------------+-------+----------+
| 2972139e-e83e-45f2-b792-a50baf9b74b8 | L3 agent           | network.lab50 | :-)   | True     |
| 181a3dfd-8909-4cf4-93b5-536832da32b7 | DHCP agent         | network.lab50 | :-)   | True     |
| d3ab33ad-f31b-49d0-a9f5-7ad56e5762aa | Open vSwitch agent | network.lab50 | :-)   | True     |
+--------------------------------------+--------------------+--------------+-------+----------+
```

2.3.4 Виртуальные сети

2.3.4.1 Создание виртуальных сетей

1. Зайдите на узел облачного контроллера с учётной записью demo. Настройте переменные окружения:

```
$ export OS_AUTH_URL=http://cloud:5000/v2.0
$ export OS_TENANT_NAME=project1
$ export LFT_KRB_KEYSTONE=keystone@cloud
$ export LFT_KRB_QUANTUM=quantum@cloud
```

2. Создайте внутреннюю сеть

```
$ quantum net-create private-net --router:external=False
```

3. Создайте подсеть для внутренней сети:

```
$ quantum subnet-create \
        --name private-subnet private-net 10.11.0.0/24 \
        --gateway 10.11.0.1 \
        --allocation-pool start=10.11.0.200,end=10.11.0.250 \
        --enable-dhcp
```

4. Создайте роутер и подключите его к внутренней сети:

```
$ quantum router-create demo_router
$ quantum router-interface-add demo_router private-subnet
```

2.3.4.2 Настройка маршрутизации L3

Сервис Quantum L3 предоставляет доступ к внешней сети. Если этот сервис не настроен, Вы сможете обмениваться данными только между собой. Имейте ввиду, что эта настройка в значительной степени зависит от Вашей конфигурации. Например, обратите внимание на приведенную ниже команду subnet-create. Вам необходимо проверить сетевые настройки своей внешней подсети (в данном случае 10.0.0.0/24) и распределительного пула. Распределительный пул предоставляет всем проектам с любым IP-адресом доступ к внешней сети. Пул состоит из 50 IP и, следовательно, только 50 проектов смогут получить шлюзовые IP.

1. Создайте внешнюю сеть:

```
$ quantum net-create public-net --router:external=True
```

2. Создайте подсеть для внешней сети:

```
$ quantum subnet-create --ip_version 4 \
          --name public-subnet public-net 10.0.0.0/24 \
          --gateway 10.0.0.1 \
          --allocation-pool start=10.0.0.200,end=10.0.0.250 \
          --disable-dhcp
```

3. Настройте шлюз примера маршрутизатора общедоступной сети:

```
$ quantum router-gateway-set demo_router public-net
```

4. Убедитесь, что сети созданы правильно и соединены через роутер:

```
$ quantum net-list
+--------------------------------------+-------------+----------------------------------------------------+
| id                                   | name        | subnets                                            |
+--------------------------------------+-------------+----------------------------------------------------+
| 1758e621-6a76-418e-89ae-80b2b446611d | private-net | 9be55b18-2370-42e0-baf2-3600e8aa031b 10.11.0.0/24  |
| fdbc0d7c-f515-4693-b9ed-4f050a5e3ed7 | public-net  | f4d6f8c6-96cf-4592-83cb-ba8417565b57 192.168.50.0/24 |
+--------------------------------------+-------------+----------------------------------------------------+
$ quantum router-list
+--------------------------------------+-------------+-----------------------------------------------------+
| id                                   | name        | external_gateway_info                               |
+--------------------------------------+-------------+-----------------------------------------------------+
| b54515e7-e023-4bf3-bd95-6a7c3662b0e9 | demo_router | {"network_id": "fdbc0d7c-f515-4693-b9ed-4f050a5e3ed7"} |
+--------------------------------------+-------------+-----------------------------------------------------+
```

Так же можно посмотреть сетевую топологию через инструментальную панель (вкладка Network topology).

2.4 Вычислительный узел

2.4.1 Введение

На вычислительном узле (сервере виртуализации) функционируют следующие компоненты:

— гипервизор (KVM);
— клиент ALD;
— модуль nova-compute Менеджера ВМ Глобуса;
— агент Сетевого интегратора для работы с Open vSwitch.

2.4.2 Стандартные сервисы

2.4.2.1 Операционная система

1. Установите ОС СН Astra Linux Special Edition. Подробная процедура установки в настоящем руководстве не рассматривается, при установке обратите внимание на следующие настройки:
   — Временная зона: UTC.
   — Имя узла: c01.
   — Роль: клиент ALD.
   — Пакеты: OpenSSH-Server.

   По завершении установки сервер перезагрузится.
2. Подключите дистрибутив Глобуса в APT. Вы можете использовать компакт-диск Глобуса или скопировать его на общее хранилище. После подключения дистрибутива обновите индексы с помощью команды

   ```
   # aptitude update
   ```

   и установите пакет с ключами дистрибутива Глобуса:

   ```
   # aptitude -y install spkb-archive-keyring
   ```
3. Выполните настройку сети.
   3.1. Отключите демон wicd:

   ```
   # service wicd stop
   # insserv -r wicd
   ```

   3.2. Установите пакеты программного коммутатора Open vSwitch:

   ```
   # aptitude -y install openvswitch-switch \
                         openvswitch-datapath-module-3.2.0-27-generic
   ```

   3.3. Отредактируйте `/etc/network/interfaces`. Файл будет определять внутренний сетевой мост. На вычислительном узле внешний мост не создаётся, как и было написано во введении. Поэтому весь сетевой трафик виртуальных машин будет идти через сетевой контроллер.

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
    address 10.10.10.11
    netmask 255.255.255.0
    gateway 10.10.10.9
    dns-domain lab50

allow-ovs br-int
iface br-int inet manual
    pre-up ovs-vsctl -- --may-exist add-br $IFACE
```

3.4. В файле `/etc/hosts` добавьте имена хостов `cloud`, `network`, и `c01` с правильными IP.

```
127.0.0.1       localhost
10.10.10.10     cloud.lab50 cloud
10.10.10.9      network.lab50 network
10.10.10.11     c01.lab50 c01
```

Примечание

Если при работе с простой или тестовой средой записи хостов можно выполнить вручную, то для рабочих инсталляций настоятельно рекомендуется использовать DNS, или, как минимум, использовать средство управления конфигурацией, такое как Ansible.

3.5. Перезагрузитесь.

3.6. Отключите ограничения на размер файлов (soft, hard) в файле `/etc/security/limits.conf`:

```
# sed -e '/fsize/ s/^#*/#/' -i /etc/security/limits.conf
```

4. Установите NTP (сервер протокола времени). NTP будет контролировать точность времени на сервере. Сервис необходим как для компонентов Глобуса, так и для функционирования домена Astra Linux Directory.

4.1. Установите необходимый пакет:

```
# aptitude -y install ntp ntpdate
```

4.2. Настройте демон NTP для синхронизации с сервером на узле cloud. Для этого добавьте строку в файл `/etc/ntp.conf`:

```
server cloud
```

4.3. Включите автозапуск демона NTP:

```
# insserv ntp
```

4.4. Перезапустите демон NTP:

```
# service ntp restart
```

4.5. Произведите первоначальную синхронизацию времени:

```
# ntpdate -u cloud
```

5. Установите и настройте домен *Astra Linux Directory*.

5.1. Установите пакеты:

```
# aptitude -y install ald-client-common ald-admin ald-admin-parsec-mac
```

5.2. Настройте параметры домена в файле `/etc/ald/ald.conf`:

```
DOMAIN=.lab50
SERVER=cloud.lab50
SERVER_ON=0
CLIENT_ON=1
```

5.3. Введите узел в домен ALD:

```
# ald-client commit-config
```

2.4.3 Менеджер виртуальных машин

Как и в случае с облачным контроллером, сервис Менеджера виртуальных машин Глобуса устанавливается на вычислительный узел (сервер виртуализации). Однако, в данном случае устанавливается модуль nova-compute, который и управляет запускаемыми на узле виртуальными машинами.

1. Установите следующие пакеты:

```
# aptitude -y install nova-compute-qemu python-globus-nova python-psycopg2
```

2. Настройте аутентификацию с использованием ALD:

2.1. Создайте принципала ALD для сервиса `quantum`:

```
# ald-admin service-add nova/c01
```

2.2. Добавьте принципал в группу `mac`:

```
# ald-admin sgroup-svc-add nova/c01 --sgroup=mac
```

2.3. Добавьте принципалам флаг requires-preauth:

```
# kadmin -p admin/admin -q 'modprinc +requires_preauth nova/c01.lab50'
```

2.4. Создайте новый файл ключей `/etc/nova/krb5.keytab` с ключом принципала сервиса `nova/c01`:

```
# ald-client update-svc-keytab nova/c01 --ktfile=/etc/nova/krb5.keytab
# chown nova:nova /etc/nova/krb5.keytab
# chmod 0400 /etc/nova/krb5.keytab
```

2.5. Войдите в систему с учётной записью demo и настройте переменные окружения:

```
$ export OS_SERVICE_ENDPOINT=http://cloud:35357/v2.0
$ export OS_SERVICE_TOKEN=none
$ export LFT_KRB_KEYSTONE=keystone@cloud
```

2.6. Зарегистрируйте принципала как пользователя-сервис в Службе Безопасности:

```
$ globus service-add nova/c01
```

2.7. Назначьте пользователю quantum/network роль admin в участнике service:

```
$ keystone user-role-add --role admin --tenant service --user nova/c01
```

3. Отредактируйте файл параметров nova-compute `/etc/nova/nova.conf`:

```
[DEFAULT]
state_path = /var/lib/nova
```

```
auth_strategy = keystone
lock_path = /var/lock/nova
compute_driver = globus.compute.driver.libvirt_driver.LibvirtParsecDriver
sql_connection = postgresql://nova:password@cloud/nova
rabbit_hosts=cloud:5672
glance_api_servers=cloud:9292
vnc_enabled=True
novncproxy_host=cloud
novncproxy_port=6080
novncproxy_base_url=http://cloud:6080/vnc_auto.html
iscsi_helper=tgtadm

vncserver_listen=0.0.0.0
vncserver_proxyclient_address=c01
allow_resize_to_same_host=True
sql_max_retries=-1
sql_retry_interval=30

network_api_class = nova.network.quantumv2.api.API
quantum_url = http://cloud:9696/
quantum_auth_strategy = keystone
quantum_admin_tenant_name = service
quantum_admin_username = nova/c01
quantum_admin_auth_url = http://cloud:35357/v2.0
firewall_driver = nova.virt.firewall.NoopFirewallDriver
security_group_api = quantum
service_quantum_metadata_proxy = true
quantum_metadata_proxy_shared_secret =

volume_api_class = nova.volume.cinder.API

[paste_deploy]
flavor=keystone

[keystone_authtoken]
admin_tenant_name = service
admin_user = nova/c01
auth_version = v2.0

[globus]
keystone_service = keystone@cloud
kerberos_service =
glance_service = glance-api@cloud
glance_registry_service = glance-registry@cloud
quantum_service = quantum@cloud
cinder_service = cinder@cloud
nova_service = nova@cloud

[kerberos]
keytab = /etc/nova/krb5.keytab
keytab_principal = nova/c01.lab50
cache = $state_path/krb5_cc
```

Предупреждение

Если вы планируете запускать виртуальные машины с ненулевой мандатной меткой, необходимо добавить следующие параметры в файл конфигурации:

```
state_path=/var/lib/nova/$mac_label/$mac_category

#обязательно включить:

#регистрирует настройки и создает каталоги
compute_manager = globus.compute.manager.ComputeManager

#управляет форками
rpc_backend = globus.common.rpc.impl_kombu

#обязательно для impl_kombu
libvirt_nonblocking = False
```

4. Настройте NBD.

   4.1. Добавьте модуль ядра nbd в автозагрузку:

   ```
   # echo nbd >> /etc/modules
   ```

   4.2. Загрузите модуль nbd:

   ```
   # modprobe nbd
   ```

5. Для корректной работы Оператора блочных устройств с пакетом Open iSCSI необходимо создать каталог `/etc/iscsi/ifaces`:

   ```
   # mkdir -p /etc/iscsi/ifaces
   ```

   и запустите демон iscsid

   ```
   # service open-iscsi start
   ```

6. Перезапустите nova-compute:

   ```
   # service nova-compute restart
   ```

7. Убедитесь, что сервис функционирует исправно.

   7.1. Зайдите на узел облачного контроллера с учётной записью demo. Настройте переменные окружения:

   ```
   $ export OS_AUTH_URL=http://cloud:5000/v2.0
   $ export OS_SERVICE_TOKEN=none
   $ export LFT_KRB_KEYSTONE=keystone@cloud
   $ export LFT_KRB_NOVA=nova@cloud
   $ export OS_TENANT_NAME=project1
   ```

   7.2. Запросите список имеющихся сервисов:

```
$ nova service-list
+------------------+-------+----------+---------+-------+----------------------------+
| Binary           | Host  | Zone     | Status  | State | Updated_at                 |
+------------------+-------+----------+---------+-------+----------------------------+
| nova-compute     | c01   | nova     | enabled | up    | 2014-08-21T07:57:19.316879 |
| nova-conductor   | cloud | internal | enabled | up    | 2014-08-21T07:57:24.731150 |
| nova-console     | cloud | internal | enabled | up    | 2014-08-21T07:57:21.949159 |
| nova-consoleauth | cloud | internal | enabled | up    | 2014-08-21T07:57:19.494618 |
| nova-scheduler   | cloud | internal | enabled | up    | 2014-08-21T07:57:22.537319 |
+------------------+-------+----------+---------+-------+----------------------------+
```

2.4.4 Настройка сетевого интегратора

2.4.4.1 Агент для Open vSwitch

Компоненты программного коммутатора Open vSwitch мы установили и настроили в предыдущих шагах. Теперь пришло время настроить агент Сетевого интегратора для работы с Open vSwitch.

1. Установите пакеты модуля Open vSwitch:

```
# aptitude -y install quantum-plugin-openvswitch-agent
```

2. Отредактируйте файл настроек Сетевого интегратора `/etc/quantum/quantum.conf`. В нем содержаться параметры, используемые всеми компонентами Интегратора.

```
[DEFAULT]
core_plugin = quantum.plugins.openvswitch.ovs_quantum_plugin.OVSQuantumPluginV2
rabbit_hosts = cloud

[AGENT]
root_helper = sudo quantum-rootwrap /etc/quantum/rootwrap.conf

[keystone_authtoken]
admin_tenant_name = service
admin_user = quantum/c01
auth_version = v2.0

[globus]
keystone_service = keystone@cloud
kerberos_service = quantum@cloud
```

3. Отредактируйте файл настроек модуля Open vSwitch `/etc/quantum/plugins/openvswitch/ovs_quantum_plugin.ini`:

```
[DATABASE]
sql_connection = postgres://quantum:password@cloud/quantum
sql_max_retries = -1

[OVS]
tenant_network_type = gre
tunnel_id_ranges = 1:1000
enable_tunneling = True
local_ip = 10.10.10.11
intergration_bridge = br-int

[SECURITYGROUP]
firewall_driver=quantum.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver
```

4. Перезапустите агент:

```
# service quantum-plugin-openvswitch-agent restart
```

Примечание

Проверьте файл `/var/log/quantum/openvswitch-agent.log` на ошибки, которые могут препятствовать успешному старту сетевых сервисов.

3 Создание первой виртуальной машины

Вы можете использовать консольные клиенты или инструментальную панель Глобуса для управления своей облачной инфраструктурой. Если вы используете панель, в браузере перейдите по адресу http://cloud, находясь при этом в созданной в одном из предыдущих этапов учётной записи (demo).

Создание ключей и запуск виртуальной машины с помощью панели инструментов

1. Отредактируйте стандартную группу безопасности («default») для разрешения трафика ICMP и SSH.
2. Создайте персональную пару криптоключей `default_key`.

Примечание

> Если возникнет ошибка: невозможно создать криптоключи, пара ключей уже создана, возможно, необходимо создать ключи, используя консольные команды, как описано ниже.

3. Измените права доступа к файлу криптоключей `default_key`.

```
$ chmod 0400 default_key.pem
```

4. В панели перейдите в «Виртуальные машины» и выберите «Запустить машину», чтобы создать новую виртуальную машину.

Создание ключей и запуск виртуальной машины через консоль на облачном контроллере

Для успешной работы с сервисами необходимо настроить переменные окружения. Вы можете прописать их вручную или скачать через инструментальную панель rc-файл и запустить его.

1. Отредактируйте стандартную группу безопасности («default») для разрешения трафика ICMP и SSH:

```
$ nova secgroup-add-rule default tcp 22 22 0.0.0.0/24
$ nova secgroup-add-rule default icmp -1 -1 0.0.0.0/24
```

2. Создайте пару ключей SSH (`default_key`) и добавьте ее в Менеджер ВМ:

```
$ ssh-keygen -f ~/.ssh/id_rsa -t rsa -N ''
$ nova keypair-add --pub_key ~/.ssh/id_rsa.pub default_key
```

Примечание

Если возникнет ошибка: невозможно создать ключи, пара ключей уже создана, возможно, необходимо создать ключи, используя панель инструментов, как описано выше.

3. Скопируйте файл ключей `default_key`.

```
$ cp ~/.ssh/id_rsa.pub ~/default_key.pem
```

4. Запросите у Реестра образов список образов и отметьте идентификатор образа, на основе которого вы хотите запустить виртуальную машину:

```
$ nova image-list
```

5. Запустите виртуальную машину:

```
$ nova boot my_instance --image <image_id> --flavor 1 --key-name default_key
```

В качестве параметра *`image_id`* должен быть использован идентификатор образа виртуальной машины из списка образов, запрашиваемого выше.

6. Подождите несколько секунд и проверьте состояние вашей виртуальной машины:

```
$ nova show my_instance
```

7. Если статус виртуальной машины «ERROR», проанализируйте файл `nova-scheduler.log`:

```
$ tail /var/log/nova/nova-scheduler.log
```

8. Если виртуальная машина стала доступна, (состояние «Available»), проверьте состояние загрузки на консоли ВМ:

```
$ nova console-log my_instance
```

Доступ по SSH к загруженной виртуальной машине

После успешной загрузки виртуальной машины и после того, как на консоли отобразится приглашение, вы можете войти в свою машину по SSH. Для этого вы должны работать в особом сетевом пространстве на сетевом узле:

1. Распечатайте список всех сетевых пространств:

```
# ip netns
```

Вывод должен включать две строки: одна строка начинается с `qrouter`, а другая с `qdhcp`.

2. Запустите SSH внутри пространства `qdhcp`. Этот пример показывает, как получить доступ по SSH к виртуальной машине, сделанной на основе образа CirrOS ранее:

```
# ip netns exec qdhcp-c73d082f-d7ed-4b53-ac93-7a6a4c3fa3aa \
  ssh -i default_key.pem cirros@10.0.0.201
```

3. Войдя первый раз как пользователь root со своей парой ключей, вы можете создать дополнительных пользователей и входить с их правами. Если используется образ CirrOS, можно войти со стандартной учётной записью Cirros (имя: `cirros`, пароль `cubswin:)`) и не использовать криптоключи.

4 Заключение

Мы построили базовую архитектуру для расширенного тестирования. Такой вид архитектуры близок к рабочему, но без обеспечения высокой готовности и некоторых дополнительных сервисов, представленных в экосистеме OpenStack. Разумеется, вы можете добавлять столько вычислительных узлов, сколько пожелаете. Если потребуется более конкретная помощь, вы можете ознакомиться с официальной документацией Глобуса или соответствующей документацией, предоставляемой сообществом OpenStack.

Приложение А. Типичные проблемы и их решения

При работе с Глобуса часто возникают одни и те же ошибки, и чтобы запросто с ними справиться, воспользуйтесь этим приложением.

Ошибки, связанные с параметрами доступа (код 401, 403)

В случае, когда у вас возникают проблемы с подключением к ППИ Менеджера виртуальных машин, в первую очередь необходимо проверить параметры аутентификации и авторизации. Такие проблемы как правило сопровождаются следующими сообщениями об ошибках:

— Invalid OpenStack Identity credentials;
— Authorization Required;
— Authorization Failed;
— Ticket expired.

При появлении подобной проблемы необходимо, в первую очередь, проверить наличие и годность билетов Kerberos с помощью команды `klist -f`:

```
$ klist -f
Ticket cache: FILE:/tmp/krb5cc_2504_nW3875
Default principal: user1@LAB50

Valid starting       Expires              Service principal
21.10.2015 11:56:18  21.10.2015 21:56:18  krbtgt/LAB50@LAB50
        renew until 28.10.2015 11:56:18, Flags: FRIA
21.10.2015 11:56:19  21.10.2015 21:56:18  ldap/cloud.lab50@LAB50
        renew until 28.10.2015 11:56:18, Flags: FRAT
21.10.2015 11:56:19  21.10.2015 21:56:18  cifs/cloud.lab50@LAB50
        renew until 28.10.2015 11:56:18, Flags: FRAT
```

Если билеты отсутствуют или протухли, их следует получить заново с помощью команды

```
$ kinit -f
```

Примечание

Билеты должны иметь атрибут «F» — Forwardable.

Если при попытке запросить что-либо у одного из сервисов выдаётся ошибка следующего содержания

```
Authorization Failed: An unexpected error prevented the server from fulfilling your request.
{'info': 'SASL(-1): generic failure: GSSAPI Error: Unspecified GSS failure. Minor code may
provide more information (Ticket expired)', 'desc': 'Local error'} (HTTP 500)
```

Следует проверить билеты у пользователя root (команда `klist -f`). Если таковые имеются, необходимо их удалить с помощью команды

```
# kdestroy
```

Если билеты Kerberos в порядке, но проблемы с доступом продолжаются, необходимо проверить параметры авторизации.

Примечание

В отличии от оригинального ПО OpenStack, Глобус использует другой набор параметров аутентификации и авторизации.

Параметры `OS_USERNAME` (имя пользователя), `OS_PASSWORD` (пароль) не используются службами Глобуса. Но должны быть указаны принципалы Керберос служб Глобуса, в том числе Менеджера виртуальных машин. Для работы с ППИ Менеджера необходимо указать три принципала:

— LFT_KRB_KEYSTONE — Служба безопасности (например, `keystone@host`);
— LFT_KRB_NOVA — Менеджер виртуальных машин (например, `nova@host`);
— LFT_KRB_CINDER — Оператор блочных устройств (например, `cinder@host`).

Принципалы можно указать как через переменные окружения, так и в аргументах команды **nova**.

Отсутствующие или неправильные параметры доступа приводят к ошибкам 401 и 403. Правильные параметры доступа можно получить через Инструментальную панель.

Предупреждение

При первом подключении пользователя с другим мандатным уровнем необходимо перезапустить сервер PostgreSQL.

Общие вопросы

1. Клиент командной строки keystone выдаёт ошибку «**Invalid OpenStack Identity credentials.**» при каком-либо запросе.
    1) Проверьте билеты Kerberos (см. «Ошибки, связанные с параметрами доступа (код 401, 403)»).
    2) Проверьте, не находитесь ли вы на сервере от имени пользователя root.
    3) Проверьте параметры аутентификации в аргументах клиента или в переменных окружения.
2. Браузер выдаёт сообщение «**Authorization Required**» при попытке зайти на инструментальную панель.
    1) Проверьте билеты Kerberos (см. «Ошибки, связанные с параметрами доступа (код 401, 403)»).
    2) Проверьте настройку веб-браузера:
        1) перейдите на страницу браузера about:config;
        2) установить у параметра `network.negotiate-auth.delegation-uris` значение `http://`;
        3) установить у параметра `network.negotiate-auth.trusted-uris` значение `http://`.

3. Браузер выдаёт сообщение «**Запрещено. У вас недостаточно привилегий для доступа...**» при попытке зайти на инструментальную панель.
Проверьте значение параметра `network.negotiate-auth.delegation-uris` в настройках веб-браузера. Должно быть `http://`.

4. После создания статус тома **creating**.
   1) Проверьте на узле, где установлен cinder-volume его состояние (из-под root'а команда lvs).
   2) Если висит, перегрузите узел.
   3) Перезапустите сервис `cinder-volume`.

5. Что делать, если том находится в подвешенном состоянии (Creating, Deleting...) и его нужно удалить?
Используйте команду `force-delete` *<имя>*.

6. ВМ создается со статусом `ERROR`, команда `nova show` *<имя>* показывает в поле `fault` причину `"No valid host was found."`
   1) Проверьте `nova service-list` (сервис `nova-compute` должен быть доступен).
   2) Проверьте наличие доступных ресурсов на узлах `nova-compute`.