Санкт-Петербургское государственное бюджетное
профессиональное образовательное учреждение
«Академия управления городской средой, градостроительства и печати»

УТВЕРЖДАЮ
Заместитель директора
по учебно-производственной работе
_____ О.В. Фомичева
«12» 12 2023 г.

МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ
по выполнению практических работ
по МДК.05.02 Безопасность облачных сервисов
ПМ.05 ЭКСПЛУАТАЦИЯ ОБЛАЧНЫХ СЕРВИСОВ

для специальности

09.02.06 Сетевое и системное администрирование

Санкт-Петербург
2023 г.

Методические рекомендации рассмотрены на заседании методического совета СПб ГБПОУ «АУГСГиП»
Протокол № 2 от «19» 11 2023 г.

Методические рекомендации одобрены на заседании цикловой комиссии информационных технологий
Протокол № 4 от «21» 11 2023 г.

Председатель цикловой комиссии: Караченцева М.С. _____

Разработчики: преподаватели СПб ГБПОУ «АУГСГиП»

С О Д Е Р Ж А Н И Е

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Рабочая тетрадь по выполнению практических работ предназначена для организации работы на практических занятиях по МДК.05.02 Безопасность облачных сервисов, которая является важной составной частью в системе подготовки специалистов среднего профессионального образования по специальности 09.02.06 «Сетевое и системное администрирование».

Практические занятия являются неотъемлемым этапом изучения учебной дисциплины и проводятся с целью:

- формирования практических умений в соответствии с требованиями к уровню подготовки обучающихся, установленными рабочей программой учебной дисциплины;
- обобщения, систематизации, углубления, закрепления полученных теоретических знаний;
- готовности использовать теоретические знания на практике.

Практические занятия по МДК.05.02 Безопасность облачных сервисов способствуют формированию в дальнейшем при изучении профессиональных модулей, следующих общих и профессиональных компетенций:

ОК 1. Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам;

ОК 2. Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности;

ОК 3. Планировать и реализовывать собственное профессиональное и личностное развитие, предпринимательскую деятельность в профессиональной сфере, использовать знания по правовой и финансовой грамотности в различных жизненных ситуациях;

ОК 4. Эффективно взаимодействовать и работать в коллективе и команде;

ОК 5. Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста;

ОК 6. Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных российских духовно-нравственных ценностей, в том числе с учетом гармонизации межнациональных и межрелигиозных отношений, применять стандарты антикоррупционного поведения;

ОК 7. Содействовать сохранению окружающей среды, ресурсосбережению, применять знания об изменении климата, принципы бережливого производства, эффективно действовать в чрезвычайных ситуациях;

ОК 8. Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержания необходимого уровня физической подготовленности;

ОК 9. Пользоваться профессиональной документацией на государственном и иностранном языках.

ПК 5.5. Обеспечивать информационную безопасность в облачной инфраструктуре с помощью различных инструментов.

ПК 5.6. Проводить мониторинг системы в облачных сервисах

В рабочей тетради предлагаются к выполнению практические работы, предусмотренные учебной рабочей программой МДК.05.02 Безопасность облачных сервисов.

При разработке содержания практических работ учитывался уровень сложности освоения студентами соответствующей темы, общих и профессиональных компетенций, на формирование которых направлена дисциплина.

Выполнение практических работ в рамках МДК.05.02 Безопасность облачных сервисов позволяет освоить комплекс работ по выполнению практических заданий по всем темам МДК.05.02 Безопасность облачных сервисов.

Рабочая тетрадь по МДК.05.02 Безопасность облачных сервисов имеют практическую направленность и значимость. Формируемые в процессе практических занятий умения могут быть использованы студентами в будущей профессиональной деятельности.

Рабочая тетрадь предназначена для студентов колледжа, изучающих МДК.05.02 Безопасность облачных сервисов.

Оценки за выполнение практических работ выставляются по пятибалльной системе. Оценки за практические работы являются обязательными текущими оценками и выставляются в журнале теоретического обучения.

1. Перечень практических работ по МДК.05.02 Безопасность облачных сервисов

№ раздела, темы	Освоение умений в процессе занятия	Формируемые ОК и ПК	Тема практического занятия	Кол-во часов
Тема 2.1. Безопасность облачных сервисов	– Разрабатывать и внедрять процессы проверки подлинности на уровне подразделения и компании в целом, контролировать доступ к системе управления общедоступным облаком; – Проводить постоянные проверки отказоустойчивости и восстановления системы; – Внедрение решений для мониторинга с целью формирования предупреждений и автоматизации реагирования на различные инциденты;	ПК 5.5.-5.6 ОК 1-9	Практическое занятие 1. Развёртывание WAF (Web Application Firewall)	4
			Практическое занятие 2. Настройка WAF (Web Application Firewall)	4
			Практическое занятие 3. Настройка сервисов сертификации на сервисах	4
			Практическое занятие 4. Настройка сервисов аутентификации на сервисах	4
			Практическое занятие 5. Настройка системы мониторинга состояния сети и сервисов	4
			Практическое занятие 6. Настройка механизмов управления правами доступа пользователей	4
			Практическое занятие 7. Настройка отказоустойчивости	4
			Практическое занятие 8. Развёртывание защиты от DoS атак	4
			Практическое занятие 9. Развёртывание защиты от DDoS атак	4
			Практическое занятие 10. Настройка микросегментации сети виртуального дата-центра	4
			Практическое занятие 11. Настройка макросегментации сети виртуального дата-центра	4
			Практическое занятие 12. Установка облачного хранилища типа: файловое	4
			Практическое занятие 13. Установка облачного хранилища типа: блочное	4

2. Описание порядка выполнения практических работ

Практическая работа №1 Настройка WAF (Web Application Firewall)

Задание:

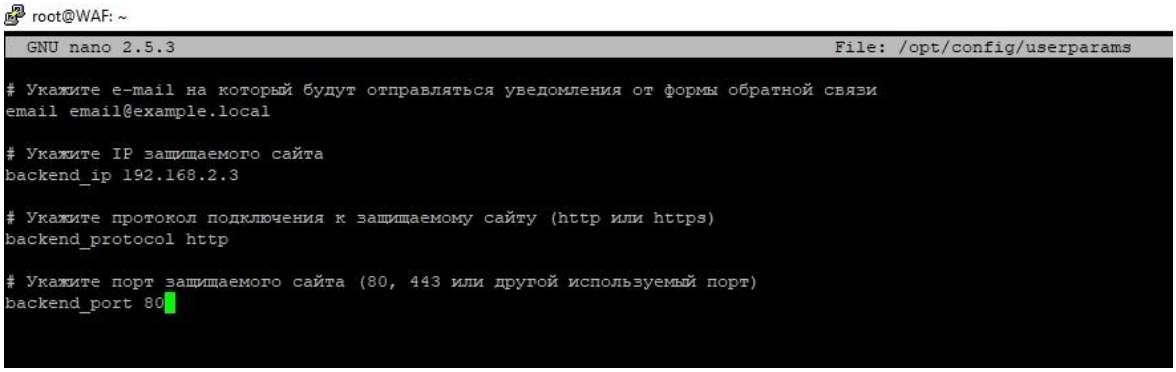
1. Для быстрого внедрения WAF клиентам в Public Catalogs доступен шаблон WAF-modsecurity.

   Шаблон на Ubuntu 16.04 LTS включает:
   - Nginx/1.13.12
   - ModSecurity for Nginx/3.0.0
   - OWASP ModSecurity Core Rule Set Version 3.0.0
   - Модуль Nginx Length Hiding Filter Module
   - Модуль Nginx Headers More Module
   - OpenSSL 1.0.2g
   - Fail2Ban
   - Форму обратной связи для оповещения о ложных срабатываниях WAF

Для первоначальной настройки необходимо:

1. Скопировать имеющиеся у Вас SSL сертификаты в папку **/opt/ssl,** переименовав их в ssl_certificate.crt и ssl_certificate.key (шаблон WAF подразумевает, что защищаемый сайт использует HTTPS);

2. Внести в файл **/opt/config/userparams** следующие данные:



3. Выполнить скрипт **/opt/config/set_userparams.sh,** который перенесёт указанные параметры в конфигурацию Nginx и форму обратной связи.

После этого WAF готов к работе.

Для проверки WAF можно открыть сайт, добавив в адресную строку ?testparam=test
Например, https://Ваш IP/?testparam=test
Если все настроено правильно, откроется форма обратной связи с предложением отправить сообщение администратору о ложном срабатывании WAF.
При нажатии на кнопку "Unblock \\ Разблокировать" на указанный ранее E-mail будет от-

правлено письмо с техническими данными, данными из логов и текстом сообщения администратору.

Практическая работа № 2 Создание RAID на Windows Server

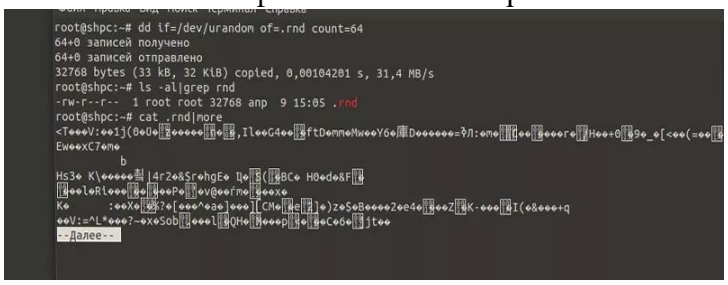
Задание:

1. Базовый WAF. Включить базовую веб-защиту (edgecenter.ru)

Практическая работа № 3 Настройка сервисов сертификации на сервисах

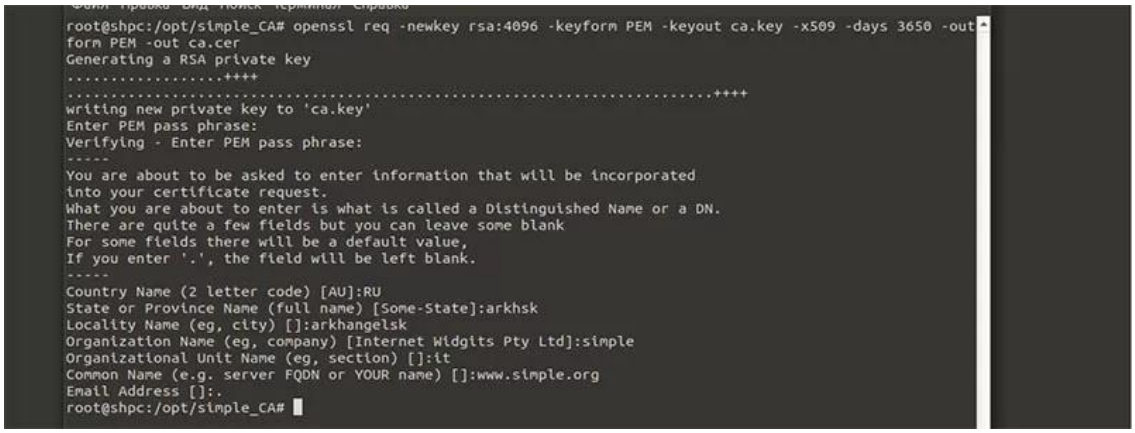
Задание:

1. перва заполним файл /root/.rnd, который используется как источник начальных случайных значений в генераторе псевдослучайных чисел OpenSSL. Суть использования этого файла описана на сайте OpenSSL. Нам надо заполнить свой файл случайными данными, как вариант — скопировав из /dev/urandom 32768 байт в файл .rnd таким образом:



Далее создадим сертификат для CA:

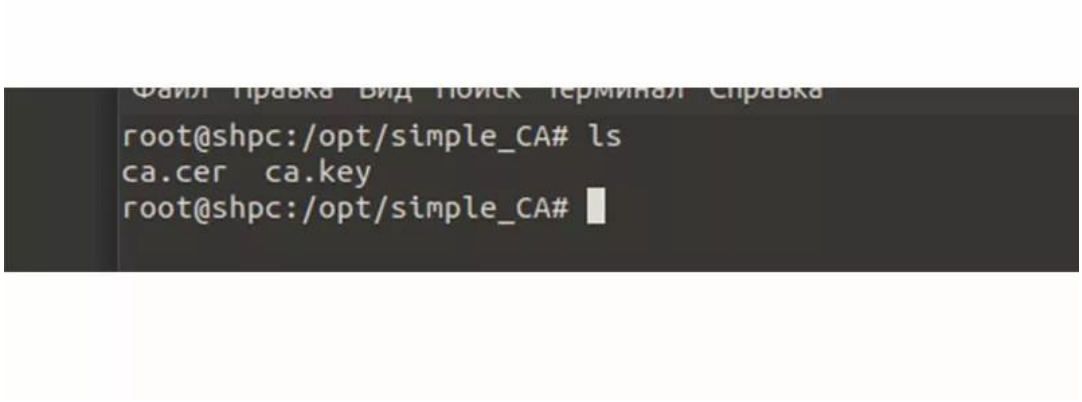
root@shpc:/opt/simple_CA# openssl req -newkey rsa:4096 -keyform PEM -keyout ca.key -x509 -days 3650 -outform PEM -out ca.cer



Опции для сертификата берутся из файла /etc/ssl/openssl.cnf. Сам файл довольно большой, мы не будем приводить здесь его содержимое. В реальности стоит создать свой конфигура-

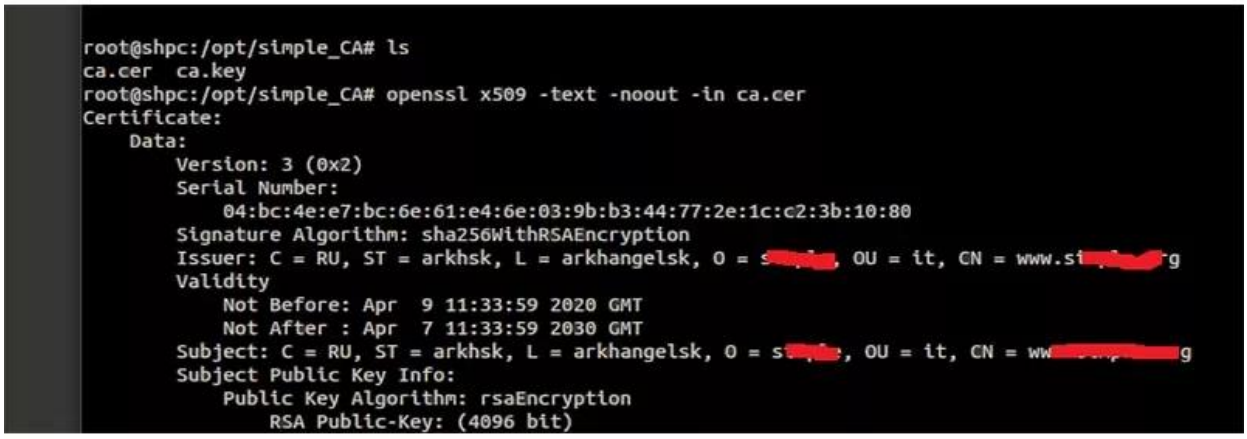
ционный файл с необходимыми опциями и блоками — и использовать его для генерации сертификатов.

На выходе получим два файла: ключ и сертификат.



Оба файла надо беречь. Если они попадут к злоумышленнику, он сможет использовать их для генерации сертификатов. Посмотреть сертификат можно при помощи этой команды (вывод обрезан для краткости):

root@shpc:/opt/simple_CA# openssl x509 -text -noout -in ca.cer



Далее на основе полученного сертификата (напомним, что это сертификат корневого CA) можно сгенерировать сертификат для сервера. Вначале генерируем закрытый ключ для сервера:

root@shpc:/opt/simple_CA# openssl genrsa -out server.key 4096

```
root@shpc:/opt/simple_CA# openssl genrsa -out server.key 4096
Generating RSA private key, 4096 bit long modulus (2 primes)
...........................++++
.....................................................++++
e is 65537 (0x010001)
root@shpc:/opt/simple_CA# ls
ca.cer  ca.key  server.key
root@shpc:/opt/simple_CA# cat server.key
-----BEGIN RSA PRIVATE KEY-----
MIIJKQIBAAKCAgEAudJDVv/Y4RhFl30woL8cm4rugYcz1erGoEw9tq0J4jdsNJwz
JDFBJJuM6MCn430Umwr9FzXeaVWV8+FBc+dZp/CotcI9OTzRy6avfyi7rEmqWvJ+
```

Теперь используем этот ключ для генерации запроса на выдачу сертификата (CSR):

root@shpc:/opt/simple_CA# openssl req -new -key server.key -out server.req -sha256

```
root@shpc:/opt/simple_CA# openssl req -new -key server.key -out server.req -sha256
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:RU
State or Province Name (full name) [Some-State]:arkhsk
Locality Name (eg, city) []:arkhangelsk
Organization Name (eg, company) [Internet Widgits Pty Ltd]:s[redacted]
Organizational Unit Name (eg, section) []:it
Common Name (e.g. server FQDN or YOUR name) []:w[redacted]rg
Email Address []:.

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
root@shpc:/opt/simple_CA#
```

Отметим, что данные, которые вы вводите в поля, должны совпадать со значениями в тех полях, что указывались при создании сертификата корневого сервера. Теперь возьмём корневой сертификат CA и запрос — и сгенерируем на их основе сертификат для сервера:

root@shpc:/opt/simple_CA# openssl x509 -req -in server.req -CA ca.cer -CAkey ca.key -set_serial 100 -extensions server -days 1460 -outform PEM -out server.cer -sha256

```
root@shpc:/opt/simple_CA# openssl x509 -req -in server.req -CA ca.cer -CAkey ca.key -set_serial 100 -extensions
rver -days 1460 -outform PEM -out server.cer -sha256
Signature ok
subject=C = RU, ST = arkhsk, L = arkhangelsk, O = s[redacted], OU = it, CN = w[redacted]rg
Getting CA Private Key
Enter pass phrase for ca.key:
root@shpc:/opt/simple_CA#
```

Должно получиться 5 файлов.

Файл server.req можно удалить — а при необходимости создать заново. Теперь надо перенастроить веб-сервер для работы с сертификатами.

Практическая работа № 4 Настройка сервисов аутентификации на сервисах

Задание:
Создайте файл docker-compose.yaml и добавьте в него демонстрационное приложение. Пробросьте порт 8081 для проверки работоспособности.

```
services:
  spring-service:
    container_name: spring-service
    image: openidentityplatform/spring-security-openam-example
    restart: always
    ports:
      - "8081:8081"
    environment:
      JAVA_OPTS: -Dspring.profiles.active=jwt
    networks:
      openam_network:
        aliases:
          - spring-service
networks:
openam_network:
driver: bridge
```

После того, как приложение запущено, проверим к доступ к API, для которого требуется добавить аутентификацию.

```
curl http://localhost:8081/api/protected-jwt | json_pp
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   105    0   105    0     0  12684      0 --:--:-- --:--:-- --:--:-- 13125
{
   "error" : "Unauthorized",
   "path" : "/protected-jwt",
   "status" : 401,
   "timestamp" : "2024-04-16T06:01:25.331+00:00"
}
```

Для успешной аутентификации, в API нужно передать валидный JWT в HTTP заголовке Authorization. За это и будет отвечать связка OpenAM + OpenIG. Потушите пока тестовый сервис командной docker compose down

Настройка OpenAM

Сначала настроим сервис аутентификации OpenAM. Он будет отвечать за аутентификацию и конвертацию токена аутентификации в JWT (об этом ниже).

Добавьте имена хостов OpenAM и OpenIG в файл hosts, например 127.0.0.1 openam.example.org openig.example.org .

В Windows системах файл hosts находится по адресу C:\\Windows\\System32\\drivers\\etc\\hosts , в Linux и Mac находится по адресу /etc/hosts .

Добавьте в файл docker-compose.yaml сервис OpenAM:

```
...
  openam:
    image: openidentityplatform/openam:latest
    container_name: openam
    ports:
      - "8080:8080"
    networks:
      openam_network:
        aliases:
          - openam.example.org
...
```

Запустите OpenAM командой docker compose up openam. После того, как OpenAM запущен, сконфигурируйте его командой:

```
docker exec -w '/usr/openam/ssoconfiguratortools' openam bash -c \
'echo "ACCEPT_LICENSES=true
SERVER_URL=http://openam.example.org:8080
DEPLOYMENT_URI=/$OPENAM_PATH
BASE_DIR=$OPENAM_DATA_DIR
locale=en_US
PLATFORM_LOCALE=en_US
AM_ENC_KEY=
ADMIN_PWD=passw0rd
AMLDAPUSERPASSWD=p@passw0rd
COOKIE_DOMAIN=example.org
ACCEPT_LICENSES=true
DATA_STORE=embedded
DIRECTORY_SSL=SIMPLE
DIRECTORY_SERVER=openam.example.org
DIRECTORY_PORT=50389
DIRECTORY_ADMIN_PORT=4444
DIRECTORY_JMX_PORT=1689
ROOT_SUFFIX=dc=openam,dc=example,dc=org
DS_DIRMGRDN=cn=Directory Manager
DS_DIRMGRPASSWD=passw0rd" > conf.file && java -jar openam-configurator-tool*.jar --file
conf.file'
```

И дождитесь завершения выполнения.

Настройка STS

STS (Security Token Service) - сервис конвертации токена OpenAM в JWT. После аутентификации OpenAM возвращает токен аутентификации, который является случайно сгенерированной последовательностью символов. Сервис STS отвечает за конвертацию токена

аутентификации в JWT, который и содержит информацию об аутентифицированном пользователе. Этот сервис и будет использовать OpenIG.
Для настройки STS зайдите в консоль администратора по ссылке
http://openam.example.org:8080/openam/XUI/#login/
В поле логин введите значение amadmin, в поле пароль введите значение из параметра ADMIN_PWD команды установки, в данном случае passw0rd

Откройте корневой realm, в левом меню нажмите пункт STS и создайте новый инстанс со следующими параметрами:

Setting	Value
Supported Token Transforms	OPENAM->OPENIDCONNECT;don't invalidate interim OpenAM session
Deployment Url Element	jwt
The id of the OpenID Connect Token Provider	https://openam.example.org/openam
Client secret	changeme
Confirm client secret	changeme
The audience for issued tokens	https://openam.example.org/openam

Настройка тестового пользователя
В консоли администратора OpenAM перейдите в корневой realm, в меню слева выберите пункт Subjects. Задайте пароль для пользователя demo. Для этого выберите его в списке пользователей, и нажмите ссылку Edit в пункте Password. Введите и сохраните новый пароль. После настройки выйдите из консоли администратора.

Практическая работа № 5 Установка Zabbix-server на Linux

Задание:
Рассмотрим теперь практические аспекты использования Prometheus. Начнём с описания процедуры установки.
Совсем недавно Prometheus был включён в официальные репозитории Debian 8 и Ubuntu 15.10.
В Ubuntu 14.04 его тоже можно установить при помощи стандартного менеджера пакетов. Естественно, для этого понадобится подключить соответствующий репозиторий:

```
$ echo 'deb http://deb.robustperception.io/ precise nightly' > /etc/apt/sources.list
$ wget https://s3-eu-west-1.amazonaws.com/deb.robustperception.io/41EFC99D.gpg
$ sudo apt-key add 41EFC99D.gpg
$ sudo apt-get update
$ sudo apt-get install prometheus node-exporter alertmanager
```

С помощью приведённых команд мы установили сервер Prometheus, а также дополнительные компоненты — node_exporter и alertmanager. Node_exporter собирает данные о состоянии сервера, а alertmanager (о нём мы более подробно поговорим ниже) — рассыла-

ет уведомления в случае выполнения или невыполнения заданных условий.

Установка завершена, но остался ещё один маленький штрих: нужно сделать так, чтобы node_exporter постоянно собирал метрики в фоновом режиме. Для этого сначала создадим символическую ссылку в /usr/bin:
$ sudo ln -s ~/Prometheus/node_exporter/node_exporter /usr/bin

Затем создадим файл /etc/init/node_exporter.conf и добавим в него следующие строки:
Run node_exporter

start on startup

script
  /usr/bin/node_exporter
end script

Сохраним внесённые изменения и выполним команду:
$ sudo service node_exporter start

В дистрибутивах, перешедших на systemd (например, в Ubuntu 15.10), для запуска node_exporter в фоновом режиме нужно создать файл /etc/systemd/system/node_exporter.service и добавить в него следующие строки:
[Unit]
Description=Node Exporter

[Service]
ExecStart=/usr/sbin/node_exporter
Restart=Always

[Install]
WantedBy=default.target

Сохранив внесённые изменения, нужно выполнить команды:
$ sudo systemctl enable node_exporter.service
$ sudo systemctl start node_exporter

Конфигурирование

Настроек Prometheus по умолчанию вполне достаточно, чтобы следить за всем происходящим на локальной машине. Дополнительные настройки в случае необходимости всегда можно прописать в конфигурационном файле /etc/prometheus/prometheus.yml. Рассмотрим его структуру более подробно. Начинается он с секции globals:
global:
  scrape_interval: 15s
  evaluation_interval: 15s
  rule_files:

Она включает следующие параметры:
scrape_interval — интервал сбора метрик (по умолчанию — 15 секунд);
evaluation_interval — интервал сверки с правилами (по умолчанию — 15 секунд);
rule_files — файлы правил (речь о них пойдёт ниже).

Далее следует секция scrape_configs с базовыми настройками сбора метрик на сервере:

```
scrape_configs:
  - job_name: "prometheus"
  - scrape_interval: "15s"
target_groups:
    - targets:
       - "localhost:9090"
```

Она включает следующие обязательные параметры:
job_name — имя задачи;
scrape_interval — интервал сбора метрик (в приведённом примере — каждые 15 секунд);
target_groups — сервисы и группы сервисов, для которых нужно собирать метрики.

В этой же секции можно прописать дополнительные настройки:
scrape_timeout — время ожидания данных;
metrics_path — HTTP-ресурс, на который будут передаваться метрики;
scheme — протокол, который будет использоваться для передачи метрик;
basic_auth — реквизиты для авторизации на сервере, с которого будут собираться метрики (username:, password:).

Выше мы уже упомянули о том, что в конфигурационном файле можно ссылать на файлы правил. Правила помогают предварительно вычислять наиболее часто используемые или требующие значительных затрат ресурсов параметры и сохранять их в виде новых временных рядов. Осуществлять поиск по предварительно рассчитанным параметрам значительно проще, чем при каждом запросе заново вычислять их значения. Это может оказаться полезным, например, при работе с дашбордами, которые запрашивают значения параметров при каждом обновлении.

В общем виде синтаксис правил можно представить так:
<имя временного ряда>{метки} = <параметр для записи>

Приведём более конкретные и понятные примеры:

```
job:http_inprogress_requests:sum = sum(http_inprogress_requests) by (job)

new_time_series{label_to_change="new_value",label_to_drop=""} = old_time_series
```

Prometheus сверяется с правилами с определённой периодичностью, указанной в конфигурационном файле в параметре evaluation_interval). После каждой сверки Prometheus пересчитывает значение параметра и сохраняет его под новым именем

с текущей временной меткой.
Итак, структуру и синтаксис конфигурационного файла мы в общих чертах рассмотрели. Чтобы прописанные настройки вступили в силу, нужно выполнить следующую команду (вместо path/to/prometheus.yml указываем путь к конфигурационному файлу):
$ prometheus -config.file “path/to/prometheus.yml”

Веб-интерфейс

Веб-интерфейс Prometheus будет доступен в браузере по адресу: http://[IP-адрес сервера]:9090:

В поле Expression можно выбрать метрику, для которой будет отображаться график. Попробуем отследить, например, объём активной памяти на сервере. Выбираем метрику node_memory_active и нажимаем на кнопку Execute:

Над графиком расположены кнопки, с помощью которых можно выбирать период для отображения статистики.
Шаблоны консолей

Основную консоль Prometheus мы только что рассмотрели. Для просмотра более специализированных графиков используются кастомные консоли.
На сервере они хранятся в директории /etc/prometheus/consoles. Кастомные консоли отображают общую статистику сервера (node.html), статистику CPU (node-cpu.html), статистику операций ввода-вывода на сервере (cpu-disk.html) и другие. В браузере они доступны по адресу: http://[IP адрес сервера]:9090/consoles/<имя консоли>.html.
Вот так, например, выглядит консоль node.html:

Если вам не подходит ни одна из имеющихся консолей, вы можете создать собственную консоль, которая будет отображать нужную вам статистику. Для написания консолей в Prometheus используется HTML-шаблонизатор Go. Подробные инструкции по созданию кастомных консолей приведены в официальной документации.
А если вас по тем или иным причинам не устраивают имеющиеся консоли, вы можете интегрировать Prometheus с популярным инструментом Grafana.

Разработчики Prometheus создали и собственный инструмент для создания дашбордов под названием Promdash (см. также репозиторий на GitHub), по интерфейсу напоминающий Grafana. На наш взгляд, он ещё находится в несколько «сыром» состоянии, и рекомендовать его к использованию пока что рано.
Alertmanager: настройка уведомлений

Ни один инструмент мониторинга немыслим без компонента для рассылки уведомлений.

В Prometheus для этой цели используется alertmanager. Настройки уведомлений хранятся в конфигурационном файле alertmanager.conf.
Рассмотрим следующий фрагмент:

```
notification_config {
  name: "alertmanager_test"
  email_config {
    email: "test@example.org"
  }

aggregation_rule {
  notification_config_name: "alertmanager_test"
}
```

Его синтаксис вполне понятен: мы указали, что уведомления при наступлении определённого условия нужно отправлять по электронной почте на адрес test@example.org.

В конфигурационный файл можно добавлять ссылки на файлы правил (по сути они ничем не отличаются от файлов правил для сбора метрик, описанных выше). В правилах прописываются условия, при которых нужно отправлять уведомления.

В общем виде синтаксис правила выглядит так:

```
ALERT <имя проверки>
  IF <параметр и его значение>
  FOR <период времени>
  WITH <набор меток>>
  SUMMARY "<краткое описание>"
  DESCRIPTION "<образец уведомления>"
```

Рассмотрим функции правил на более конкретных примерах.
Пример1:

```
ALERT InstanceDown
  IF up == 0
  FOR 5m
  WITH {
    severity="page"
  }
  SUMMARY "Instance {{$labels.instance}} down"
  DESCRIPTION "{{$labels.instance}} of job {{$labels.job}} has been down for more than 5 minutes."
```

Это правило указывает, что уведомление нужно отправлять в случае, если некоторый инстанс недоступен в течение 5 минут и более.

Пример2:

```
ALERT ApiHighRequestLatency
  IF api_http_request_latencies_ms{quantile="0.5"} > 1000
  FOR 1m
  SUMMARY "High request latency on {{$labels.instance}}"
```

```
  DESCRIPTION "{{$labels.instance}} has a median request latency above 1s (current value:
{{$value}})"
```

Согласно этому правилу, уведомления нужно посылать, как только среднее время ответа на запросы к API превысит 1 мс.

Чтобы прописанные в конфигурационном файле настройки вступили в силу, нужно сохранить его и выполнить команду:

```
$ alertmanager -config.file alertmanager.conf
```

Можно создать несколько конфигурационных файлов и прописать в них настройки уведомлений для различных случаев.

Уведомления Prometheus отправляет в формате JSON. Выглядят они примерно так:

```
{
  "version": "1",
  "status": "firing",
  "alert": [
    {
      "summary": "summary",
      "description": "description",
      "labels": {
        "alertname": "TestAlert"
      },
      "payload": {
        "activeSince": "2015-06-01T12:55:47.356+01:00",
        "alertingRule": "ALERT TestAlert IF absent(metric_name) FOR 0y WITH ",
        "generatorURL":
"http://localhost:9090/graph#%5B%7B%22expr%22%3A%22absent%28metric_name%29%22%
2C%22tab%22%3A0%7D%5D",
        "value": "1"
      }
    }
  ]
}
```

Отправка уведомлений осуществляется по электронной почте, через веб-хук, а также с помощью специализированных сервисов: PagerDuty, HipChat и других.

Практическая работа № 6 Настройка механизмов управления правами доступа пользователей

Задание:

1. Настройте права доступа в системе в соответствии с матрицей доступа компании
2. Зафиксируйте скриншотами

Практическая работа № 7 Настройка отказоустойчивости

Задание:
Создать виртуальную сеть для кластера:
azure network vnet create vnet1 -e 10.0.0.0 -l 'West Europe'
Создать виртуальные машины:
azure vm create -l 'West Europe' --ssh 22 -A availability-set -w vnet1 my-cluster-vm-1 $IMG azureuser 'R00tp@$$'
azure vm create -l 'West Europe' --ssh 22 -A availability-set -w vnet1 my-cluster-vm-2 $IMG azureuser 'R00tp@$$'
В данном случае машины доступные снаружи под разными именами. Мы можем дать единое имя всему кластеру, тогда машины должны быть доступны на разных портах.
azure vm create -l 'West Europe' --ssh 22 -w vnet1 -n my-cluster-vm-1 my-first-cluster $IMG azureuser 'R00tp@$$'
azure vm create -l 'West Europe' --ssh 23 -w vnet1 -n my-cluster-vm-2 -c my-first-cluster $IMG azureuser 'R00tp@$$'
После того как машины запустятся, можно приступать к настройке. Адреса, полученные машинами:
$ azure vm list
info: Executing command vm list
+ Getting virtual machines
data: Name Status Location DNS Name IP Address
data: ---------- ---------- ---------- ---------- ----------
data: my-cluster-vm-1 ReadyRole West Europe my-first-cluster.cloudapp.net 10.32.0.4
data: my-cluster-vm-2 RoleStateUnknown West Europe my-first-cluster.cloudapp.net 10.32.0.5
[править]Инсталляция ключей
Определяем адреса машин (azure vm list), после этого инсталлируем на них SSH-ключи.
На одной из машин (в данном случае на первой машине, 10.32.0.4):
ssh-keygen -t dsa
ssh-copy-id -i ~/.ssh/id_dsa.pub 10.32.0.5
ssh-copy-id -i ~/.ssh/id_dsa.pub 10.32.0.4
scp ~/.ssh/id_dsa 10.32.0.5:~/.ssh/
[править]Инсталляция необходимых программ на узлах кластера
sudo apt-get install corosync pacemaker drbd8-utils
[править]Подготовка отказоустойчивого хранилища
Вне узла:
azure vm disk attach-new my-cluster-vm-1 10
На узле:
$ sudo fdisk /dev/sdc
Welcome to fdisk (util-linux 2.25.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x2810cb0c.

Command (m for help): o
Created a new DOS disklabel with disk identifier 0xaeafd4f0.

Command (m for help): n
Partition type

```
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-20971519, default 2048):
Last sector, +sectors or +size{K,M,G,T,P} (2048-20971519, default 20971519):

Created a new partition 1 of type 'Linux' and of size 10 GiB.

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.
```

Аналогично для второй машины.
Вне узла:
azure vm disk attach-new my-cluster-vm-2 10
На узле:
sudo fdisk /dev/sdc
На обоих узлах создаём конфигурационные файлы ресурса DRBD.
Файл /etc/drbd.d/r0.res:

```
resource r0 {
  on my-cluster-vm-1 {
    device  /dev/drbd1;
    disk   /dev/sdc1;
    address  10.32.0.4:7789;
    meta-disk internal;
  }
  on my-cluster-vm-2 {
    device  /dev/drbd1;
    disk   /dev/sdc1;
    address  10.32.0.5:7789;
    meta-disk internal;
  }
}
```

Файл должен быть на обеих машинах.
После этого можно приступать к запуску DRBD.
На обоих узлах одновременно:
sudo /etc/init.d/drbd start
На первом узле:
sudo drbdadm create-md r0
sudo drbdadm attach r0

<pre>
sudo drbdadm create-md r0
sudo drbdadm attach r0
На первой машине опять:
sudo drbdadm primary --force r0
Синхронизация началась:
$ cat /proc/drbd
version: 8.4.3 (api:1/proto:86-101)
srcversion: 69A5E1D3708F09A9D055736

```
 1: cs:SyncSource ro:Primary/Secondary ds:UpToDate/Inconsistent C r-----
    ns:10520 nr:0 dw:0 dr:11432 al:0 bm:0 lo:0 pe:0 ua:0 ap:0 ep:1 wo:f oos:10473860
        [>.....................] sync'ed:  0.2% (10228/10236)Mfinish: 1:20:49 speed: 2,104 (2,104) K/sec
```

[править]Инсталляция и настройка MySQL-сервера

Готовим файловую систему, на которой будут находиться данные MySQL-сервера. Файловая система будет размешена на нашем отказоустойчивом хранилище.

На первом узле это можно сделать так: отформатировать DRBD-устройство и примонтировать его в /var/lib/mysql. После этого установить MySQL-сервер. Его данные будут записаны в каталог /var/lib/mysql.

```
sudo mkfs.ext3 /dev/drbd1
sudo mkdir /var/lib/mysql
sudo mount /dev/drbd1 /var/lib/mysql
```

Инсталлируем сервер:

```
sudo apt-get install mysql-server
sudo update-rc.d mysql disable
```

На втором узле нужно или проинсталлировать mysql-сервер, а потом стереть его файлы данных, или подмонтировать временно другой каталог в /var/lib/mysql, а потом отмонитировать его. Настоящие данные уже находятся на DRBD-устройстве.

```
sudo mkdir /tmp/mysql
sudo mount --bind /tmp/mysql /var/lib/mysql
sudo apt-get install mysql-server
sudo /etc/init.d/mysql stop
sudo update-rc.d mysql disable
```

[править]Настройка кластерного программного обеспечения

Конфигурационный файл corosync /etc/corosync/corosync.conf

```
totem {
  version: 2
  crypto_cipher: none
  crypto_hash: none
  interface {
    ringnumber: 0
    bindnetaddr: 10.32.0.0
    mcastport: 5405
    ttl: 1
  }
  transport: udpu
}

logging {
  fileline: off
  to_logfile: yes
  to_syslog: yes
  logfile: /var/log/corosync/corosync.log
  debug: off
  timestamp: on
  logger_subsys {
    subsys: QUORUM
    debug: off
    }
  }
```

```
nodelist {
  node {
    ring0_addr: 10.32.0.4
    nodeid: 1
  }

  node {
    ring0_addr: 10.32.0.5
    nodeid: 2
  }
}

quorum {
  provider: corosync_votequorum
}
```

Файл должен находиться на обоих узлах.

После того как конфигурационный файл создан, можно запускать **corosync**. Чтобы после перезагрузки corosync запускался, нужно разрешить запуск в файле /etc/default/corosync:

```
sudo vim /etc/default/corosync
```

Запустить **corosync**:

```
sudo /etc/init.d/corosync start
```

Проверить, что узлы видят друг друга:

```
my-cluster-vm-2:~$ sudo corosync-quorumtool -l

Membership information
----------------------
    Nodeid      Votes Name
         1          1 10.32.0.4
         2          1 10.32.0.5 (local)
```

[править]Настройка Pacemaker

Запускаем Pacemaker (на обоих узлах):

```
$ sudo /etc/init.d/pacemaker start
```

Просматриваем текущую конфигурацию Pacemaker'a:

```
$ sudo crm configure show
node $id="1" my-cluster-vm-1
node $id="2" my-cluster-vm-2
property $id="cib-bootstrap-options" \
        dc-version="1.1.10-42f2063" \
        cluster-infrastructure="corosync"
```

Войти в режиме настройки crm:

```
sudo crm configure
```

В режиме настройки:

```
primitive res_drbd_r0 ocf:linbit:drbd params drbd_resource="r0"
primitive res_fs ocf:heartbeat:Filesystem params device="/dev/drbd1" directory="/var/lib/mysql" fstype="ext3"
ms ms_drbd_r0 res_drbd_r0 meta notify="true" master-max="1" master-node-max="1" clone-max="2" clone-node-max="1"
colocation c_r0_on_drbd inf: res_fs ms_drbd_r0:Master
order o_drbd_before_nfs inf: ms_drbd_r0:promote res_fs:start
property stonith-enabled=false
property no-quorum-policy=ignore
```

Применить изменения в конфигурации:

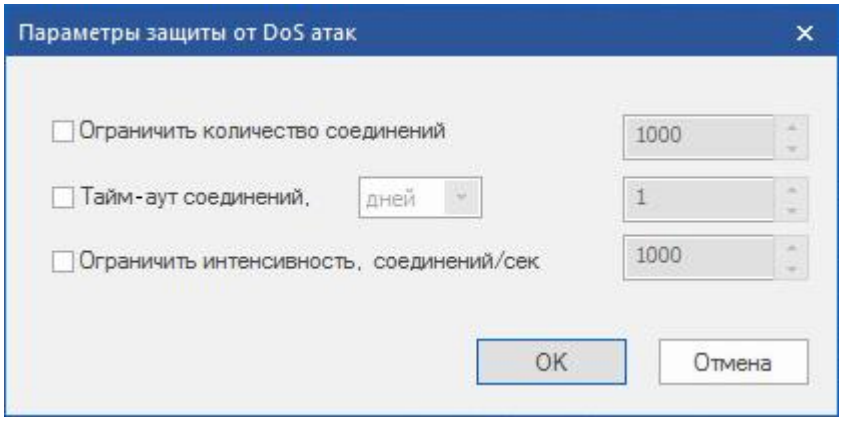
commit
В данный момент crm управляет DRBD-устройством и файловой системой, работающий поверх него. Для управление MySQL-сервером необходимо создать соответствующий примитив и объединить его в группу с файловой системой:
(в режиме crm configure)
primitive mysqld ocf:heartbeat:mysql
group mysql res_fs mysqld
Сделанные изменения нужно применить командой commit.
Полезные команды:
crm resource cleanup res_fs — очистить состояние ресурса res_fs;
crm resource list — просмотреть список доступных ресурсов;
crm node list — просмотреть список доступных узлов;
crm node standby — перевести узел в состояние standby;
crm node online — перевести узел в активное состояние.
[править]Настройка endpoint
Привязать MySQL к 0.0.0.0 в файле /etc/mysql/my.cnf:
[mysqld]
bind-address = 0.0.0.0
Разрешение подключаться удалённо (лучше не под root'ом, здесь это только для демонстрации):
GRANT ALL PRIVILEGES ON *.* TO 'root'@'%' IDENTIFIED BY 'r00tp@$$' WITH GRANT OPTION;
Создать endpoint'ы:
$ azure vm endpoint create-multiple my-cluster-vm-1 3306:3306:tcp:false:MySQL:tcp:3306
$ azure vm endpoint create-multiple my-cluster-vm-2 3306:3306:tcp:false:MySQL:tcp:3306
Список endpoint'ов:
$ azure vm endpoint list my-cluster-vm-1
info: Executing command vm endpoint list
+ Getting virtual machines
data: Name Protocol Public Port Private Port Virtual IP EnableDirectServerReturn Load Balanced
data: ----- -------- ----------- ------------ -------------- ----------------------- -------------
data: MySQL tcp 3306 3306 191.233.84.226 false Yes
data: ssh tcp 22 22 191.233.84.226 false No
info: vm endpoint list command OK

Практическая работа № 8 Развёртывание защиты от DoS атак

Задание:
Настройка режима защиты от DoS-атак
Включение и настройку режима защиты от DoS-атак выполняют в окне редактирования правила фильтрации. При этом осуществляется контроль параметров состояния соединений. Этот режим действует для данного правила при наличии следующих условий:
поле "Действие" содержит значение "Пропустить";
установлена отметка в поле "Контролировать состояние соединения".
Для настройки параметров правила фильтрации:

1. В окне редактирования правила фильтрации установите отметку в поле "Защита от DoS-атак" и нажмите кнопку "Параметры...".

На экране появится диалог "Параметры защиты от DoS-атак".



2. Заполните поля диалога и нажмите кнопку "OK".

Ограничить количество соединений	Максимальное количество соединений, которое может быть установлено по указанному правилу фильтрации
Тайм-аут соединений	Время, по истечении которого неактивное соединение будет автоматически разорвано
Ограничить интенсивность соединений/сек.	Количество новых соединений, регистрируемых для данного правила, в секунду

Практическая работа № 9 Развёртывание защиты от DDoS атак

Задание:

Стенд состоит из трёх основных частей:

1. Виртуальная среда на базе VMWare, имитирующая локальную сеть + ЦОД (слева на схеме)
2. Межсетевой экран HP NGFW
3. Виртуальная среда, имитирующая сегмент сети Интернет (справа на схеме)

Виртуальная среда, имитирующая локальную сеть, представляет собой развернутые на рабочей станции виртуальные машины на базе операционной системы Windows. Для имитации работы вредоносного ПО, на часть виртуальных машин установлен тестирующий агент, способный сгенерировать DDoS трафик.

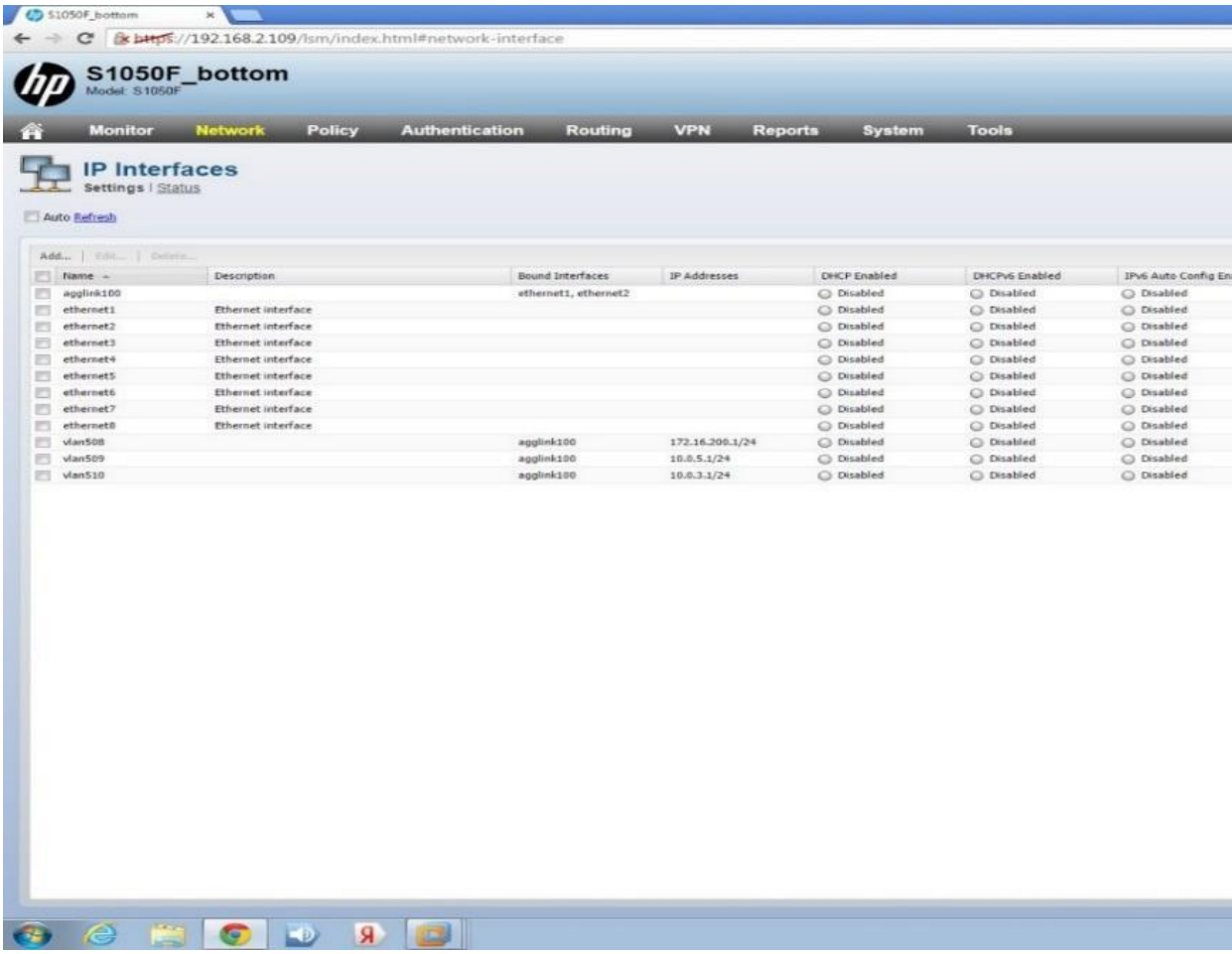
Межсетевой экран установлен в разрыв между сегментом Интернет и локальной сетью и в данном стенде представляет собой устройство третьего уровня, маршрути-

зирующее трафик между сегментами. Базовые правила и настройки межсетевого экрана показаны на схеме выше.

Виртуальная среда, имитирующая сегмент Интернет, представляет собой набор виртуальных машин на базе операционной системы Linux с установленном на них специализированным ПО – в числе прочего, веб-сервер Apache, сервер баз данных MySQL, интерпретатор языка PHP версии 5, сканер безопасности Nessus, утилита сканирования сети nmap.

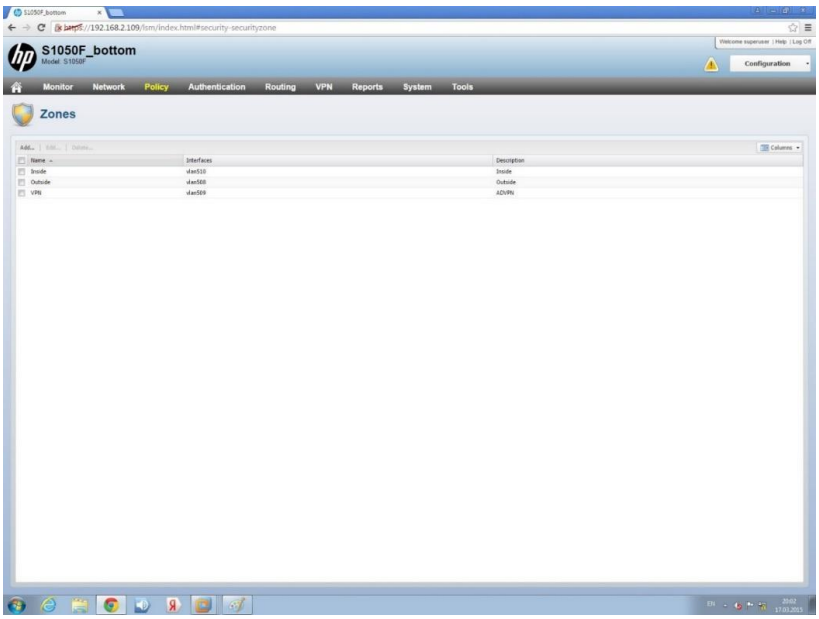
Базовая конфигурация межсетевого экрана

Ниже приведены базовые настройки NGFW в соответствии с разработанной схемой стенда, приведенной на рисунке 3. Все настройки показаны в собственном веб-интерфейсе NGFW. На рисунке 4 приведены настройки IP интерфейсов FW в соответствии со схемой приведенной выше.



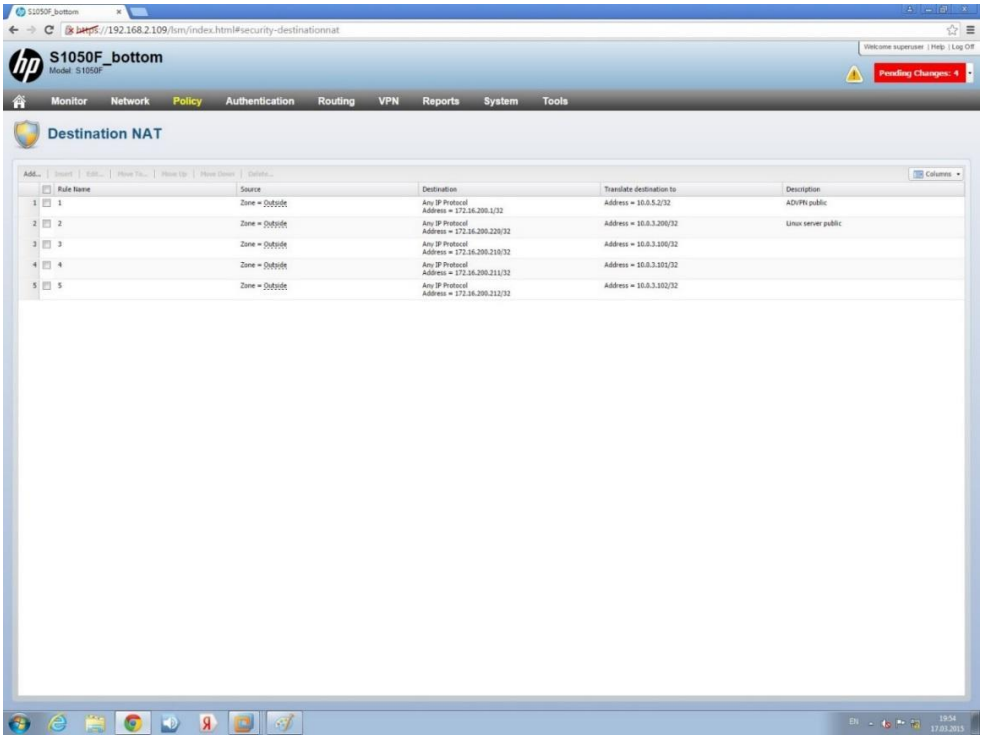
Настройки IP интерфейсов

Ниже показаны настройки зон безопасности. В стенде для простоты настроено три зоны безопасности – Inside, Outside и VPN.



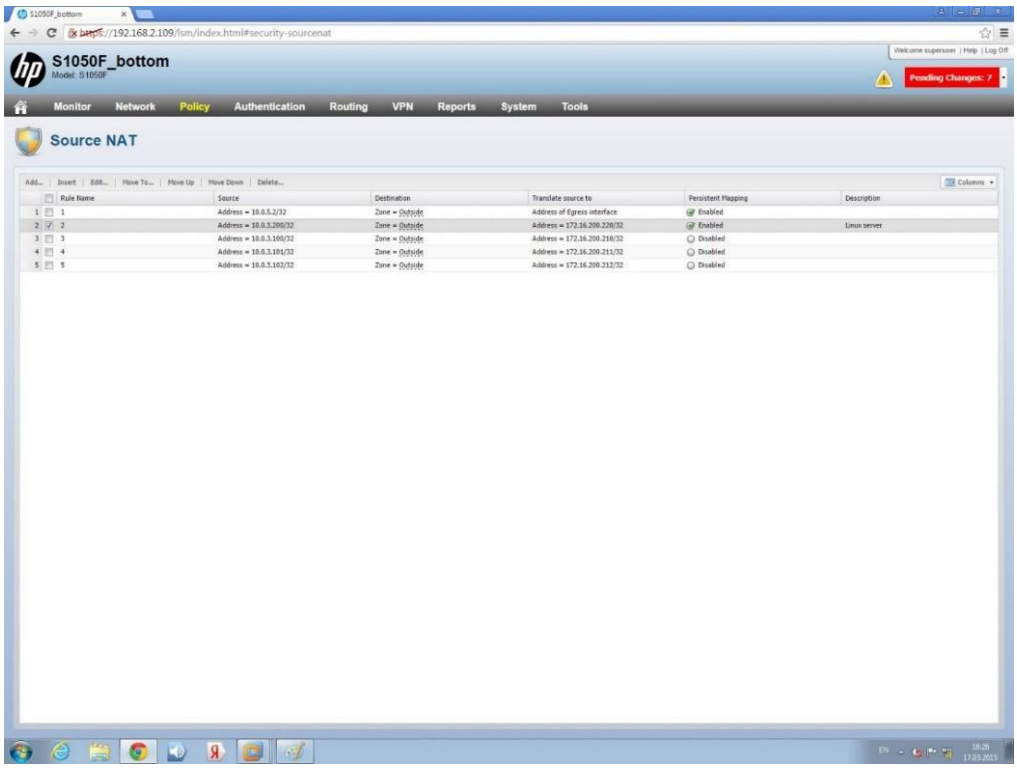
Настройки зон безопасности

На рисунке 6 показаны базовые настройки, сделанные для Destination NAT. Опубликовано несколько внутренних адресов виртуальных машин, на которых запущены различные сетевые сервисы (HTTP/HTTPS, FTP и т.д.).



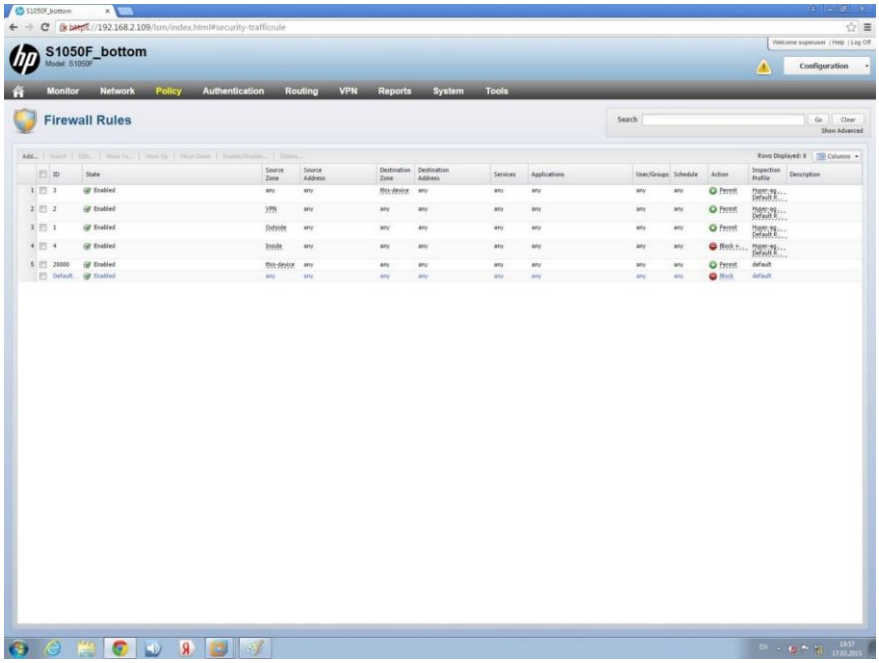
Настройки Destination NAT

Ниже показаны базовые настройки Source NAT. Несколько адресов виртуальных машин транслированы во внешние адреса, чтобы обеспечить их доступ к эмулированным сервисам с вредоносным ПО.



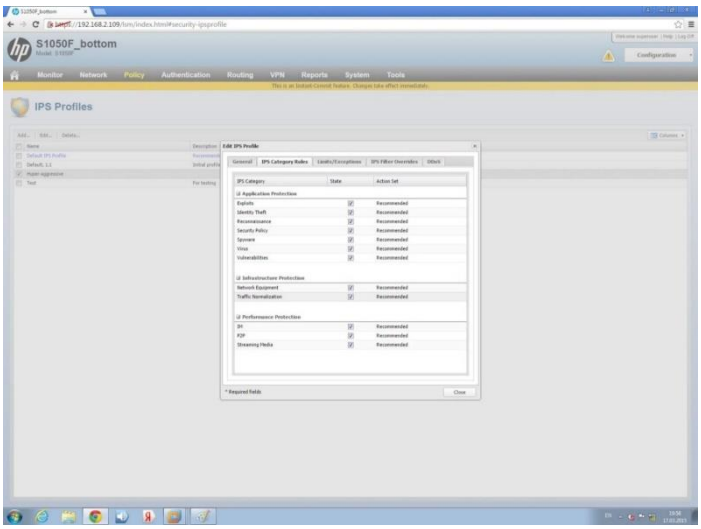
Source NAT

Базовые настройки FW Policy, в соответствии с которыми обрабатывается трафик, показаны на рисунке 8. Для того, чтобы проверить как отрабатывает политика IPS, настроенная по умолчанию, мы пропустим весь трафик через базовые фильтры Stateful FW прозрачно.



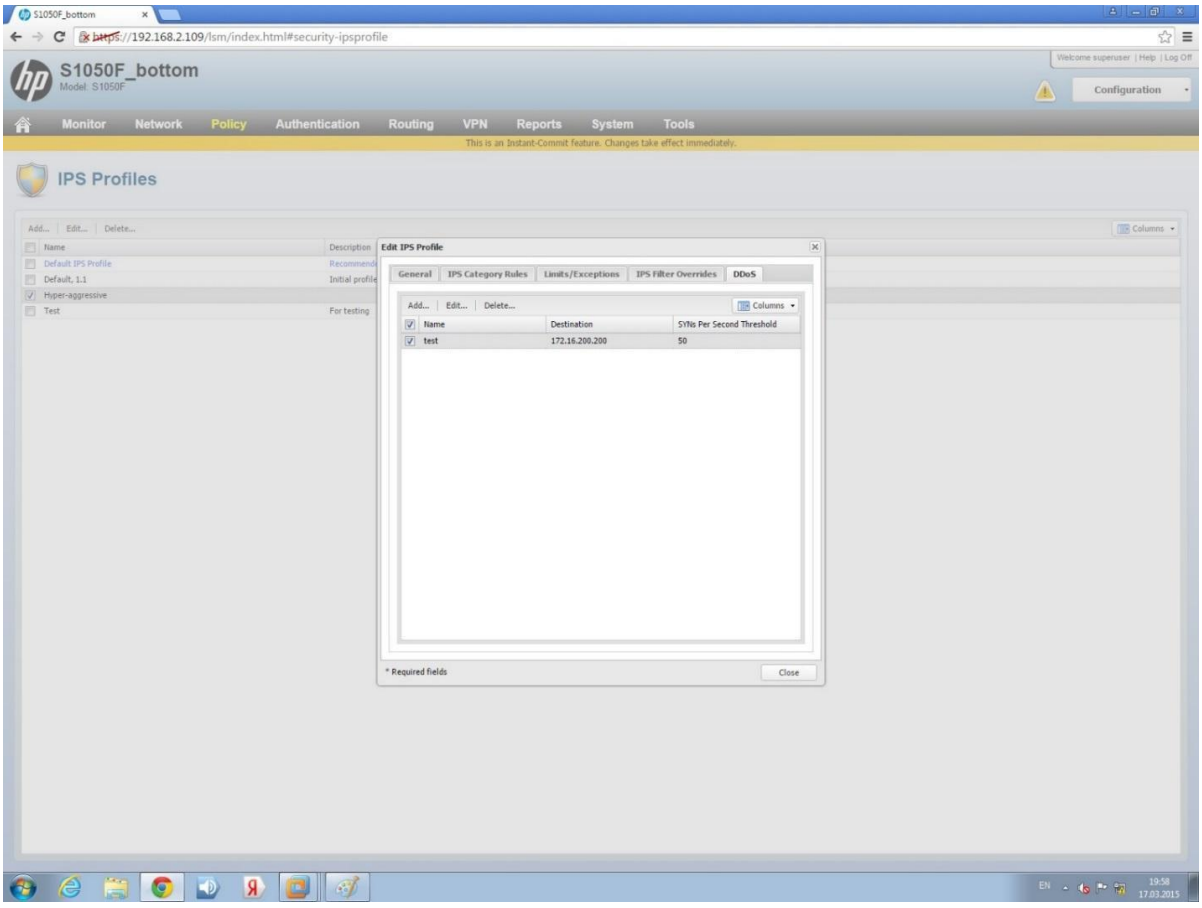
FW Policy

Базовые настройки системы противодействия вторжениям (IPS). Все рекомендованные категории фильтрации IPS включены в данном тесте.



Базовые настройки IPS

Настройка порогов для атак типа DDoS. Порог срабатывания IPS на DDoS атаку типа SYN FLOOD – 100 пакетов в секунду.

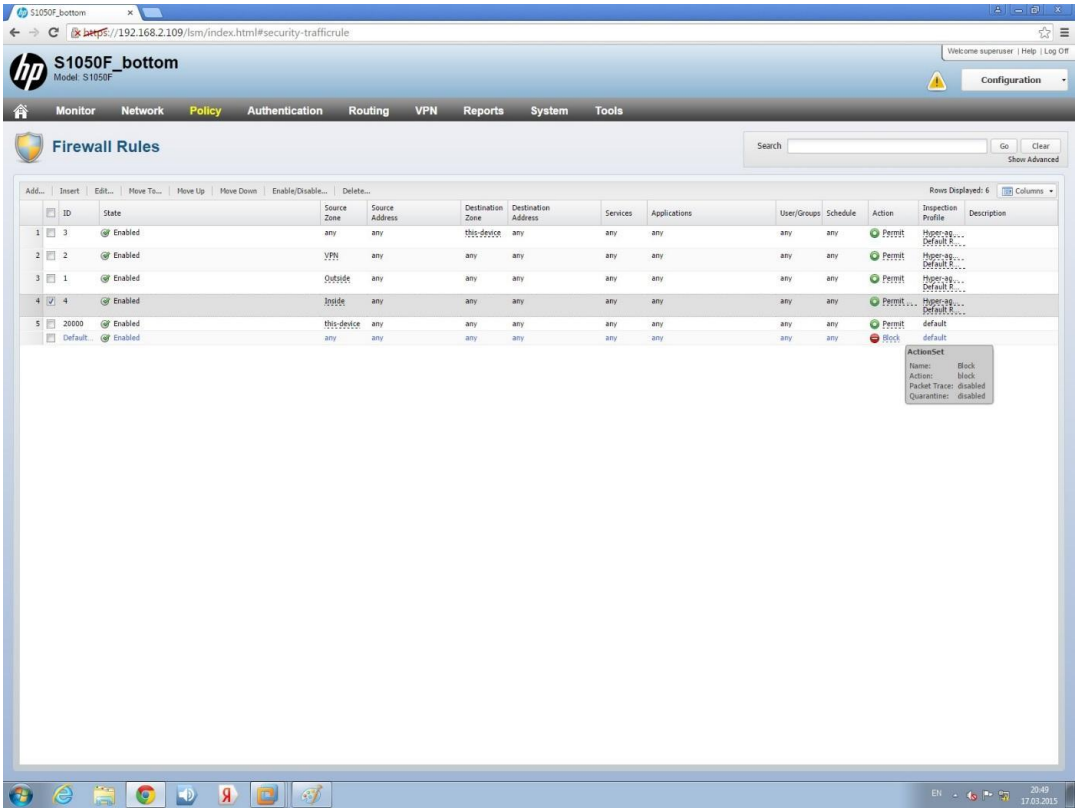


IPS DDoS

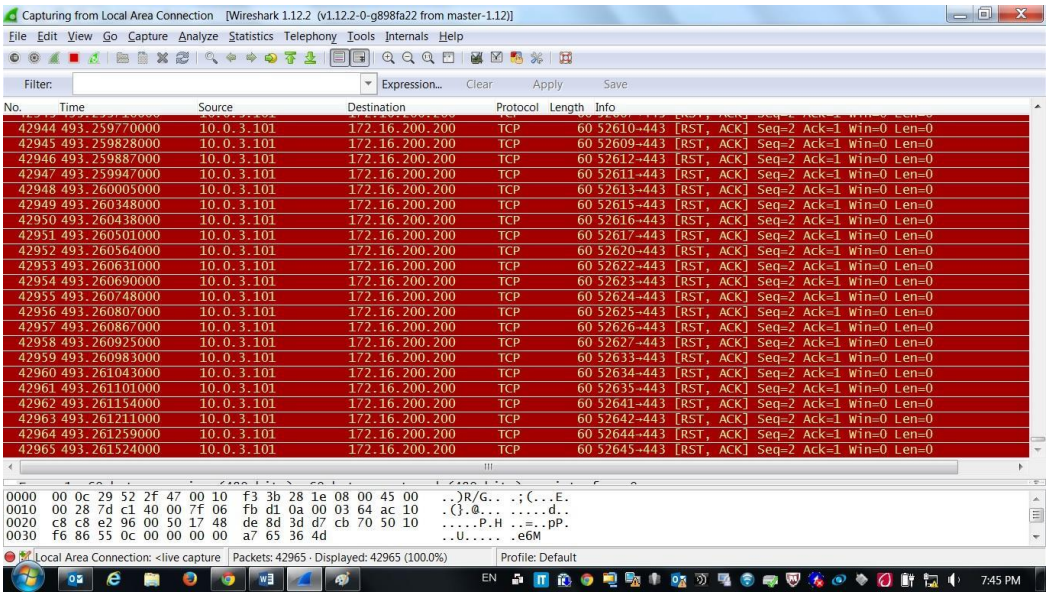
Эмулируем атаку согласно классической схеме, приведенной в части 2 серии и попытаемся защититься от атаки при помощи NGFW. При этом, в реальной DDoS атаке фильтровать по портам/адресам – подход неэффективный, так как заражённые машины разбросаны, как правило, по сети и имеют хаотически разбросанные адреса. Поэтому пропускаем весь трафик на IPS и, как показано выше, настраиваем в

IPS правило защиты от DDoS:

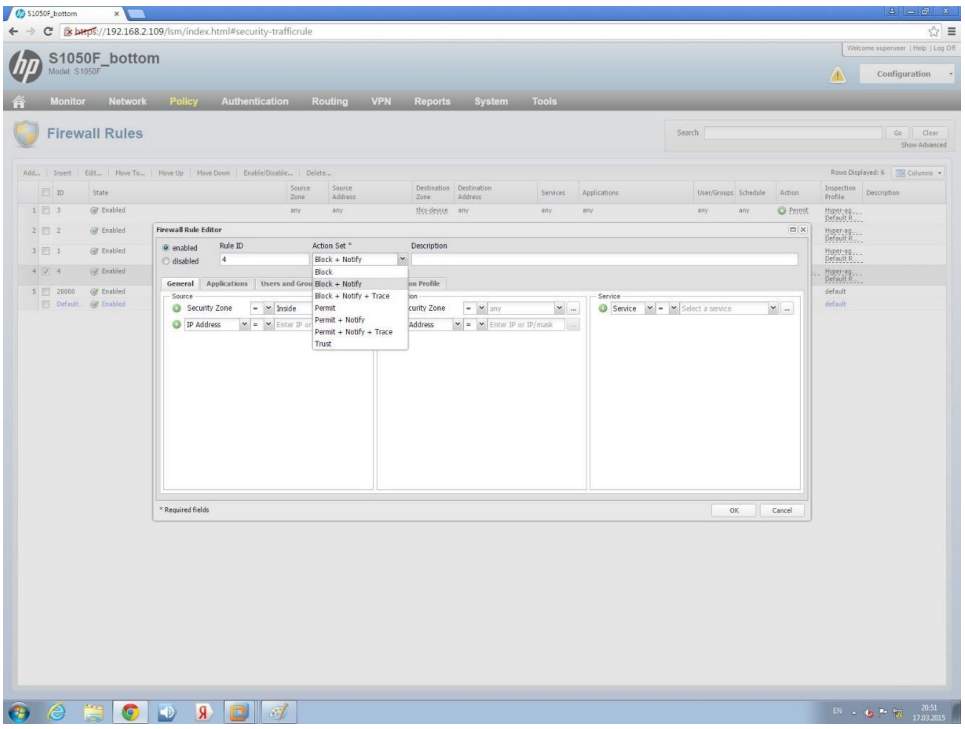
1. Чтобы убедиться, что работает именно IPS, выключаем на NGFW правило, по которому трафик хоста-источника сетевой атаки будет блокирован:



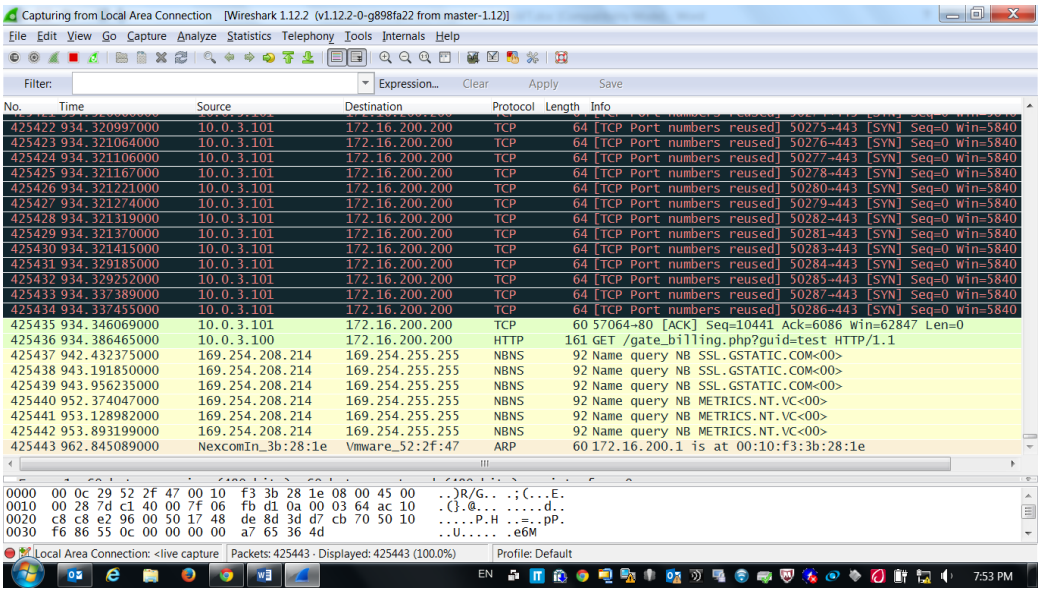
2. Запускаем сетевую атаку DDoS и смотрим в логи Wireshark и наблюдаем классическую DoS атаку с зараженной агентом виртуальной машины:



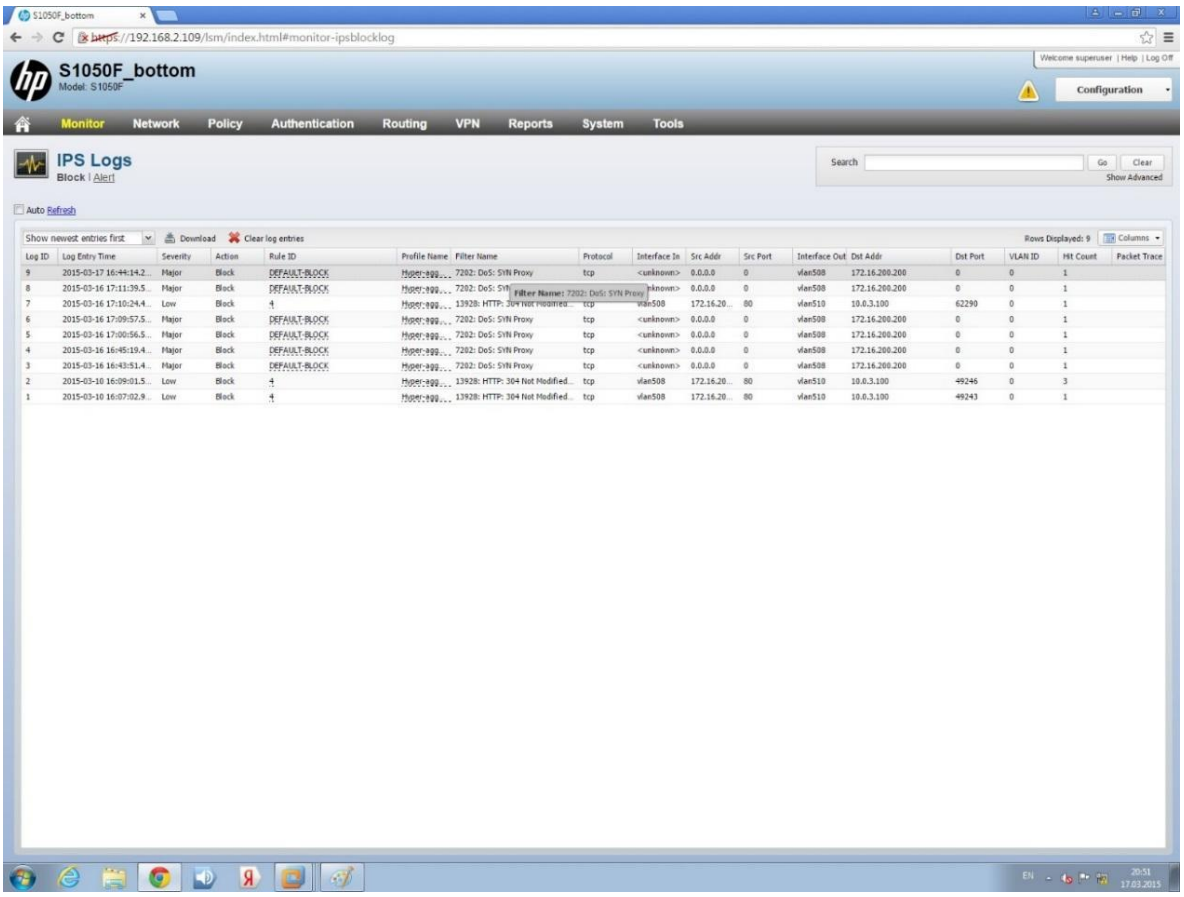
3. Включаем на NGFW IPS правило, блокирующее трафик хоста, с которого идет сетевая атака:



4. Убеждаемся в логах Wireshark, что правило сработало и DoS атака прекращена:



5. Смотрим в логи NGFW и отмечаем, что атака обнаружена и трафик с атакующего хоста закрыт:



Данный пример показывает, как при помощи правил, преднастроенных в системе обнаружения вторжений NGFW, обнаруживаются и предотвращаются классические атаки типа DDoS.

Практическая работа № 10 Настройка системы мониторинга состояния сети и сервисов

Задание:

Как с помощью микросегментации сети обеспечить безопасность мультиоблачной инфраструктуры / Хабр (habr.com)

Первоначальная настройка

При входе в систему отражается первоначальный экран панели управления, на котором представлены данные рабочих потоков, полученные от уже существующих сенсоров.

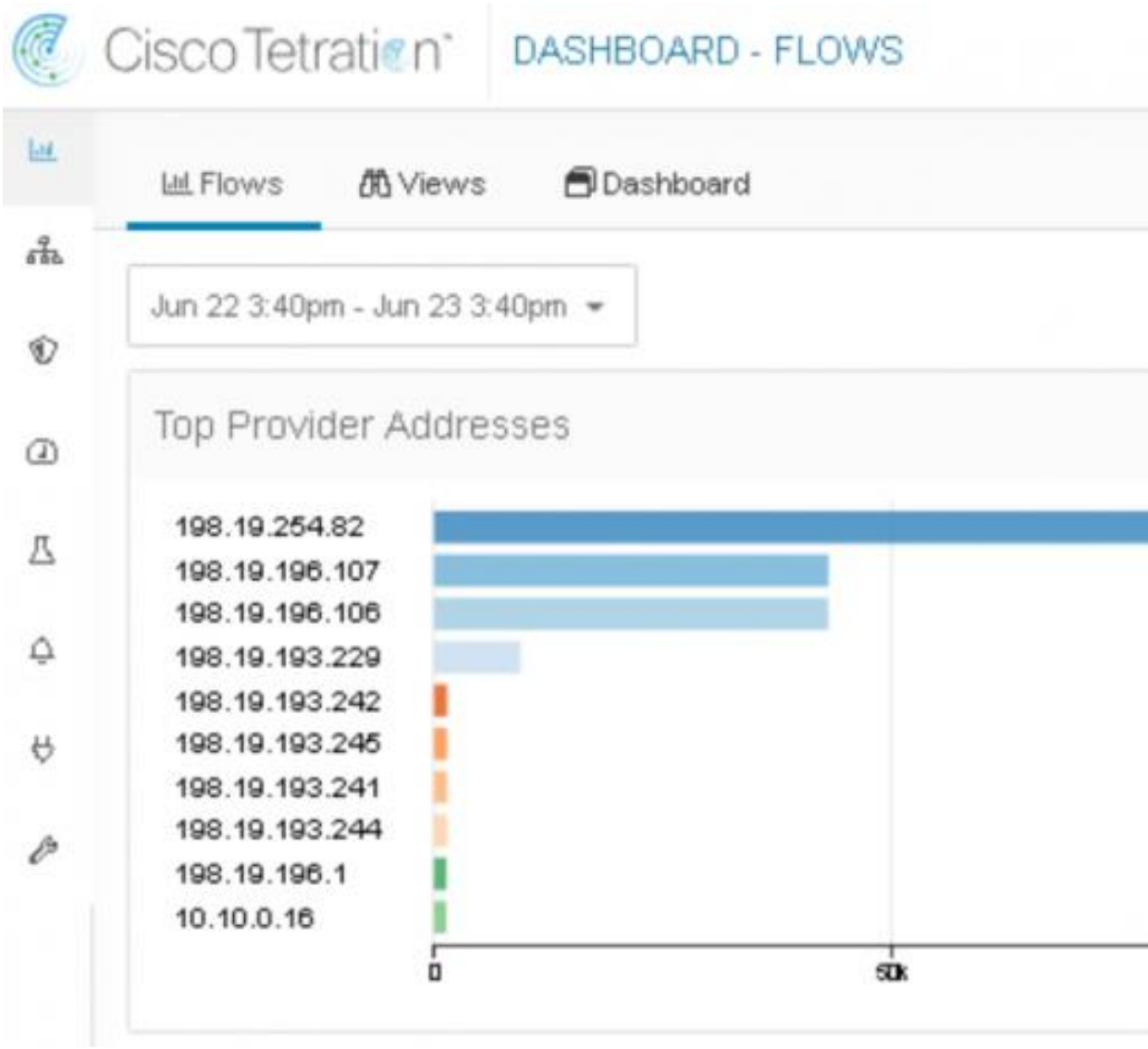


Рисунок 3. Существующие сетевые потоки

Добавление нового агента

Список существующих подов на вкладке «мониторинг», отражающей подключенные программные агенты.

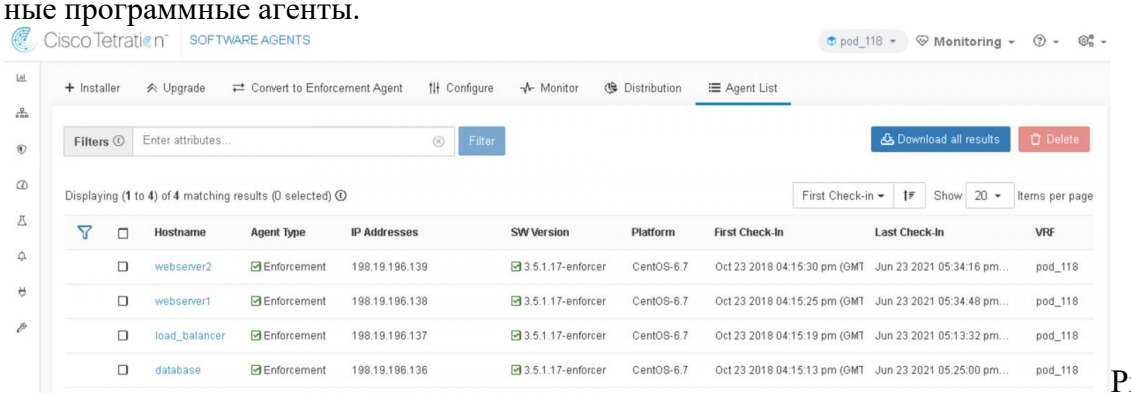


Рисунок 4. Список существующих агентов

Решением представляется два варианта установки программного агента: с автоматической привязкой и без. Для экономии времени рекомендуется к использованию вариант с автоматической привязкой. Для наглядности будем использовать его в дальнейшей работе компонентов решения.

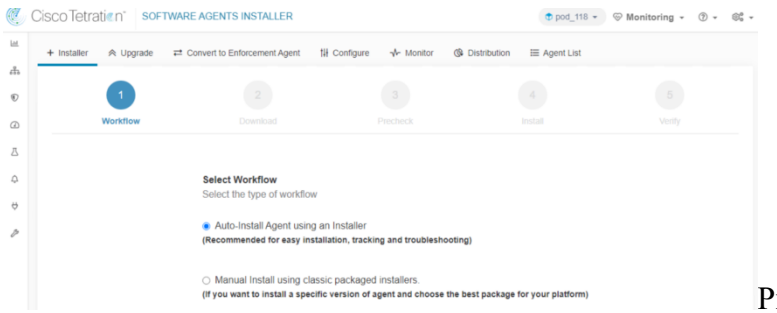


Рисунок 5. Выбор способа установки

Пример установки ПО программного сенсора на ОС MS Windows 2019.

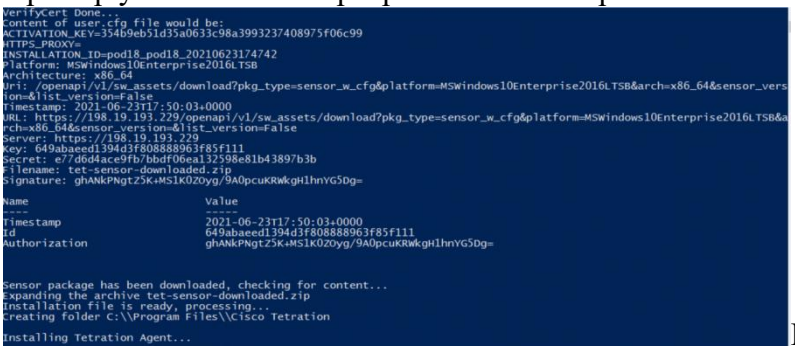


Рисунок 6. Установка средствами PowerShell

Определение пула

Cisco Secure Workload может запускать алгоритмы машинного обучения для любой рабочей нагрузки, однако необходимо иметь механизм, задающий определенные границы. В связи с этим в платформе существует концепция, называемая «scoping». Она позволяет без затруднений осуществлять запросы по изменению настроек, связанных с определенными областями инфраструктуры, обеспечивая гибкость по мере развития структуры организации.

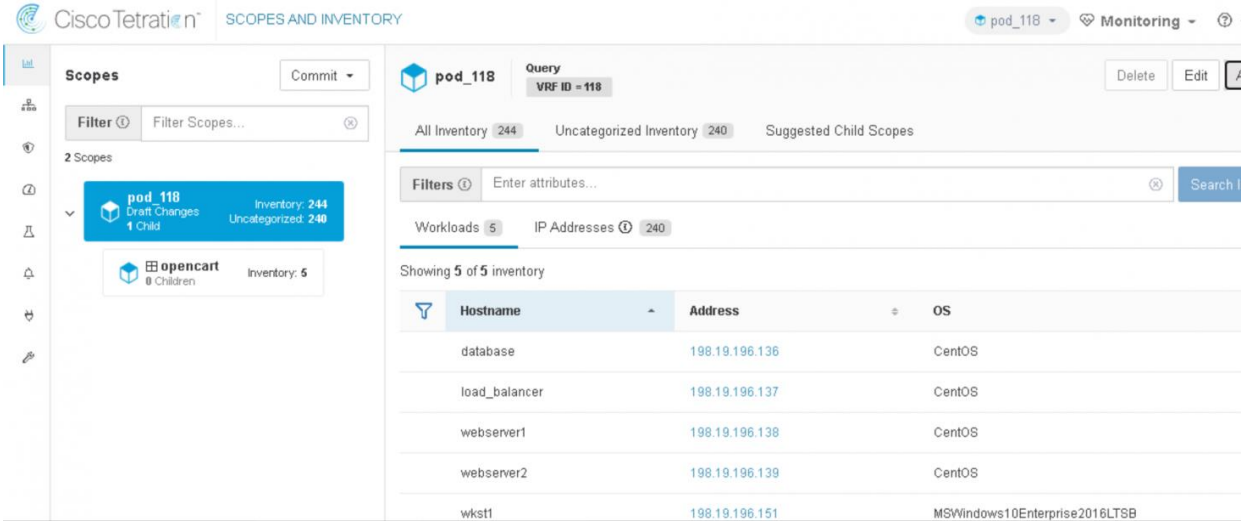
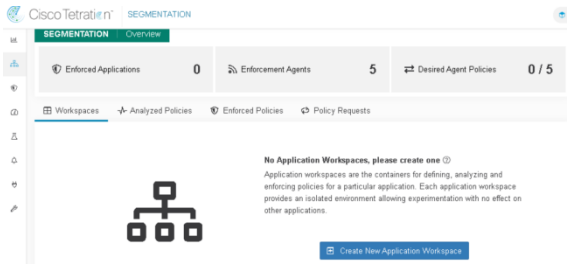


Рисунок 7. Определение нового «scope»

Картографирование (mapping)

Cisco Secure Workload может автоматически обнаруживать и сопоставлять соответствующие политики и приложения. В решении этот процесс называется сопоставлением зависимостей приложения (также известный как ADM).

Настроим процесс запуска ADM для одного из приложений. Автоматически обнаружим список разрешенных политик для приложения: для этого во вкладке "Segmentation" необходимо создать новое рабочее пространство.



Определяем политики сетевой безопасности. Каждая политика представляет собой взаимосвязь между набором рабочих нагрузок источника и пункта назначения.

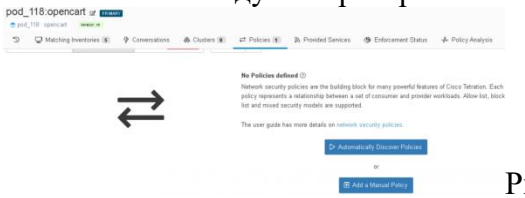


Рисунок 9. Выявление существующих политик

Во время запуска функциональности ADM Cisco Secure Workload использует неконтролируемое пользователем машинное обучение для анализа всех метаданных заданной области и пытается сгруппировать конечные точки в кластеры.

Обнаруженные существующие политики можно просмотреть на вкладке "Policy".

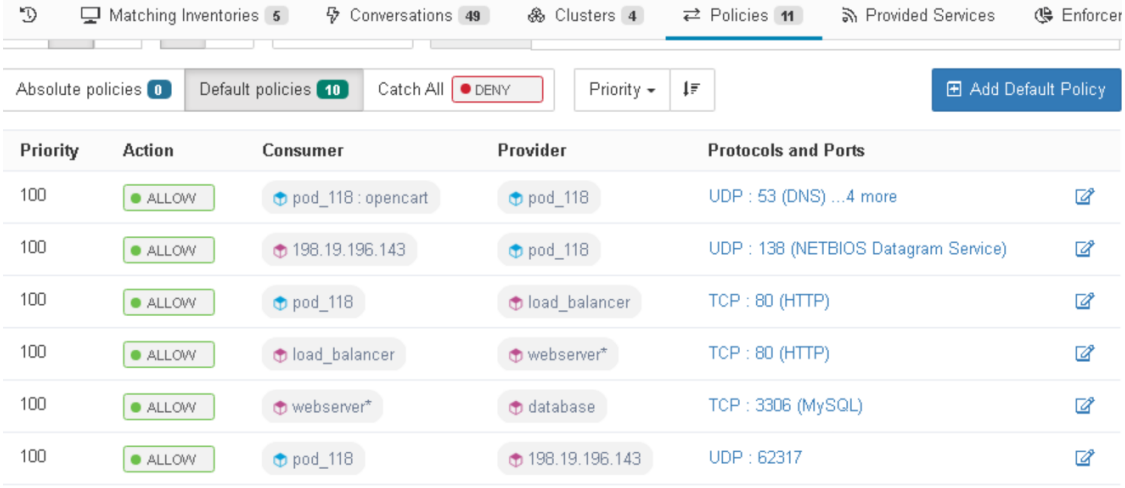


Рисунок 10. Список существующих политик

Используем встроенные средства для понимания необходимости существования политики.

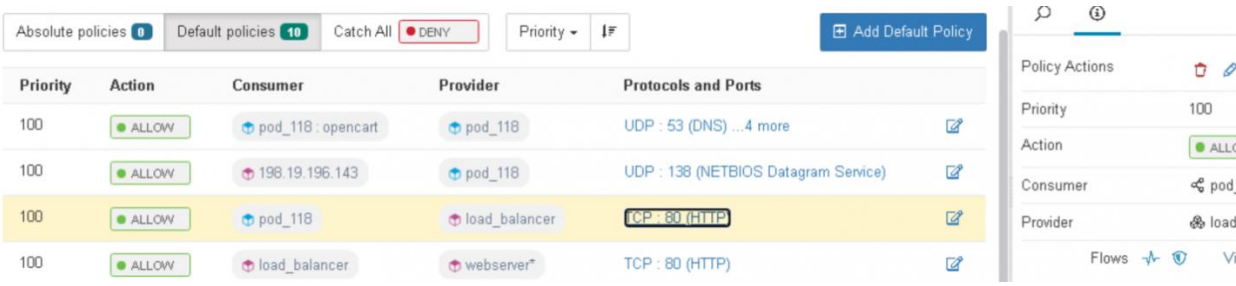


Рисунок 11. Список агрегированных потоков

Данный способ позволяет просматривать агрегированные потоки на протяжении всего выполнения ADM, даже когда порт получателя удален и за все время существует только одна запись.

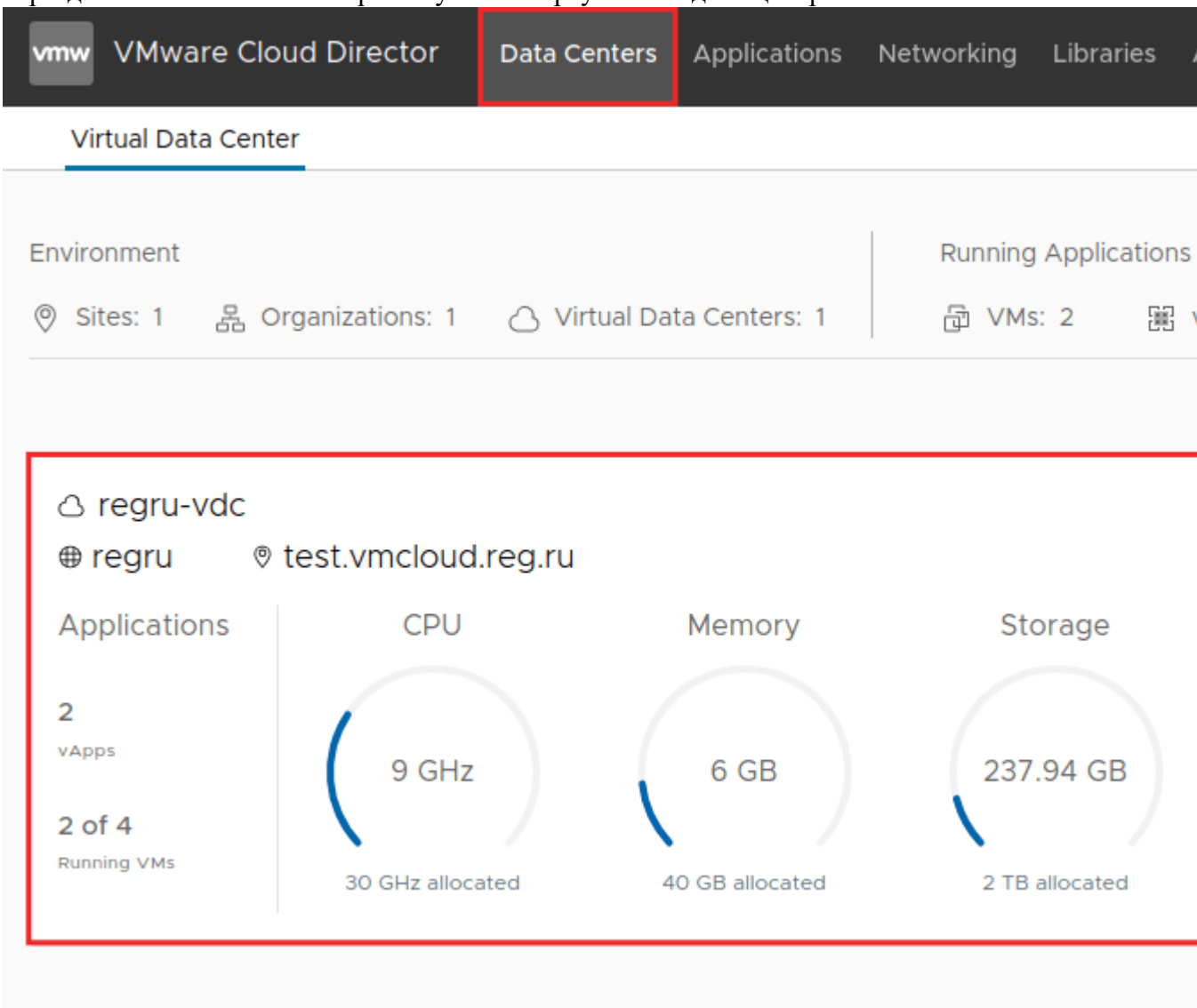
Практическая работа № 11 Настройка макросегментации сети виртуального дата-центра

Задание:

Этап 1. Создание маршрутизируемой сети

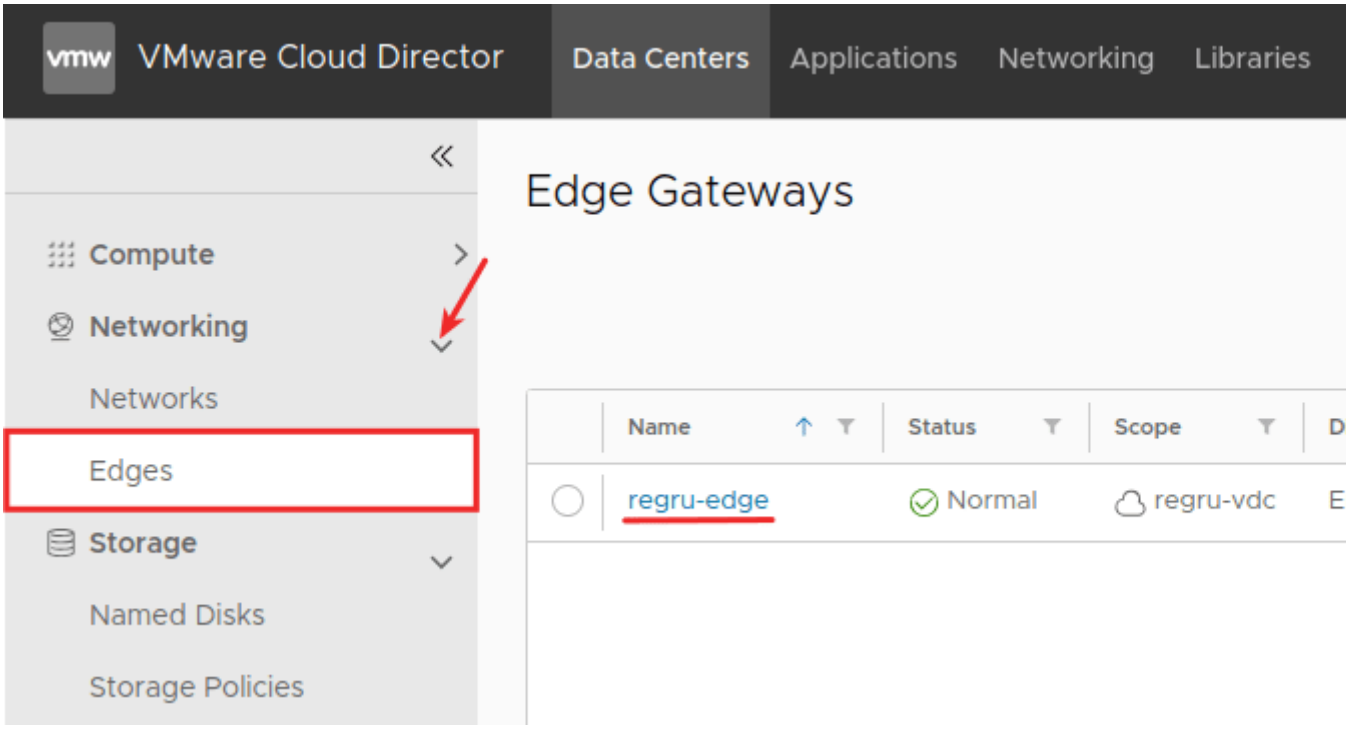
- 1Войдите в панель управления виртуальным дата-центром vCloud Director по инструкции.
- 2

В разделе «Datacenters» выберите нужный виртуальный дата-центр и кликните на него:



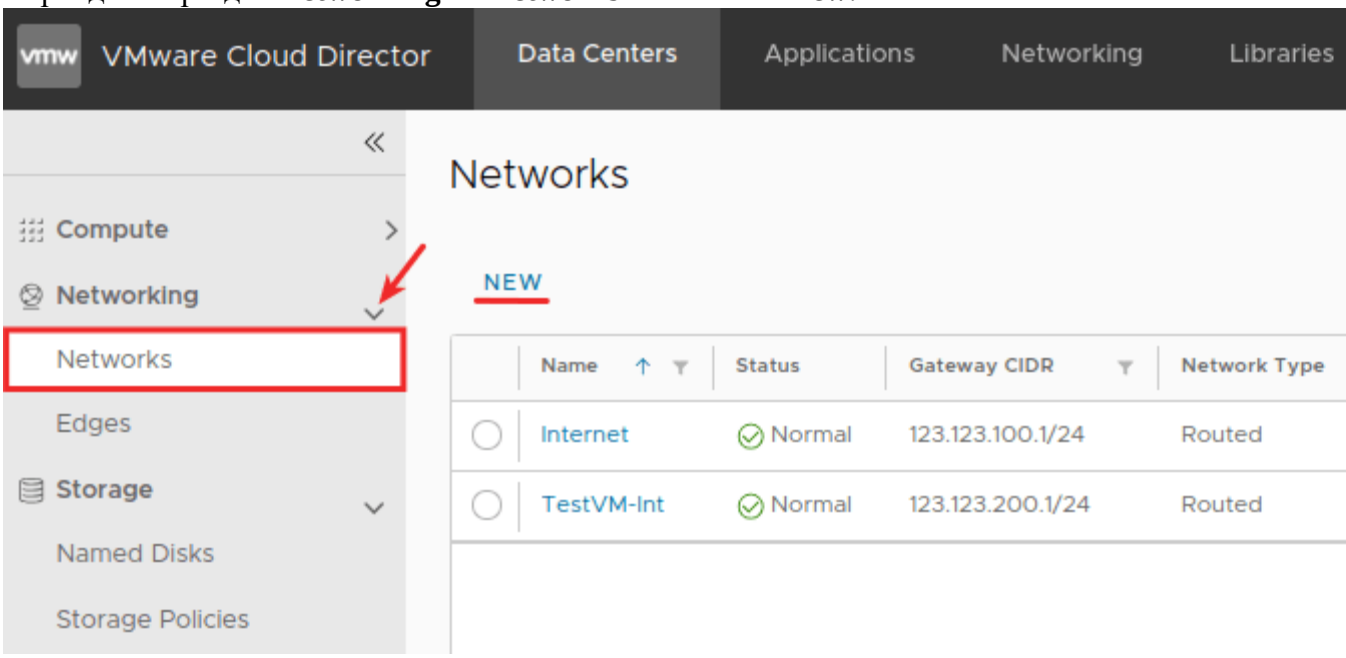
- 3

Разверните раздел **Networking** перейдите в блок **Edges**. Затем кликните на нужный Edge:



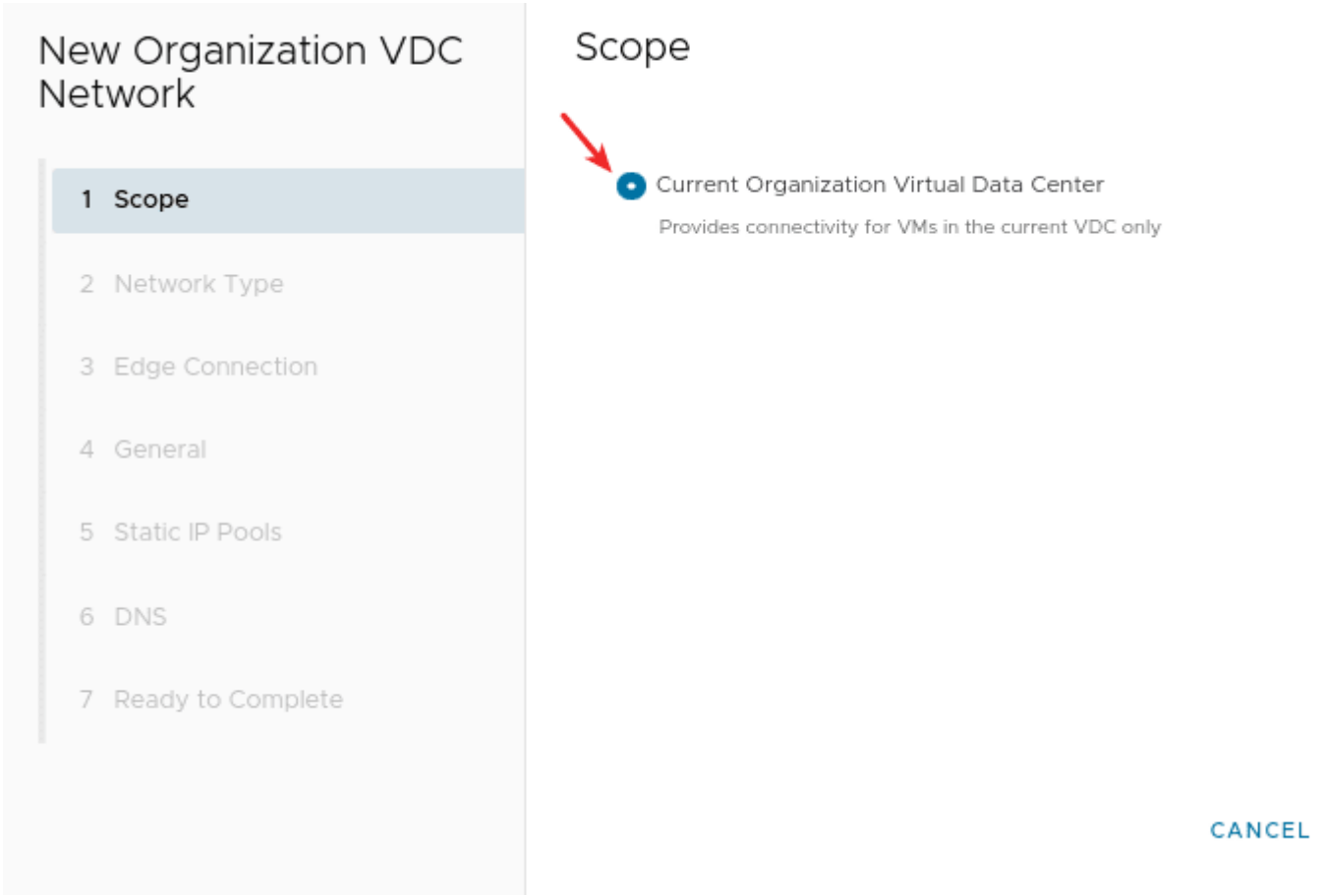
- 4

Перейдите в раздел **Networking — Networks** и кликните **New**:



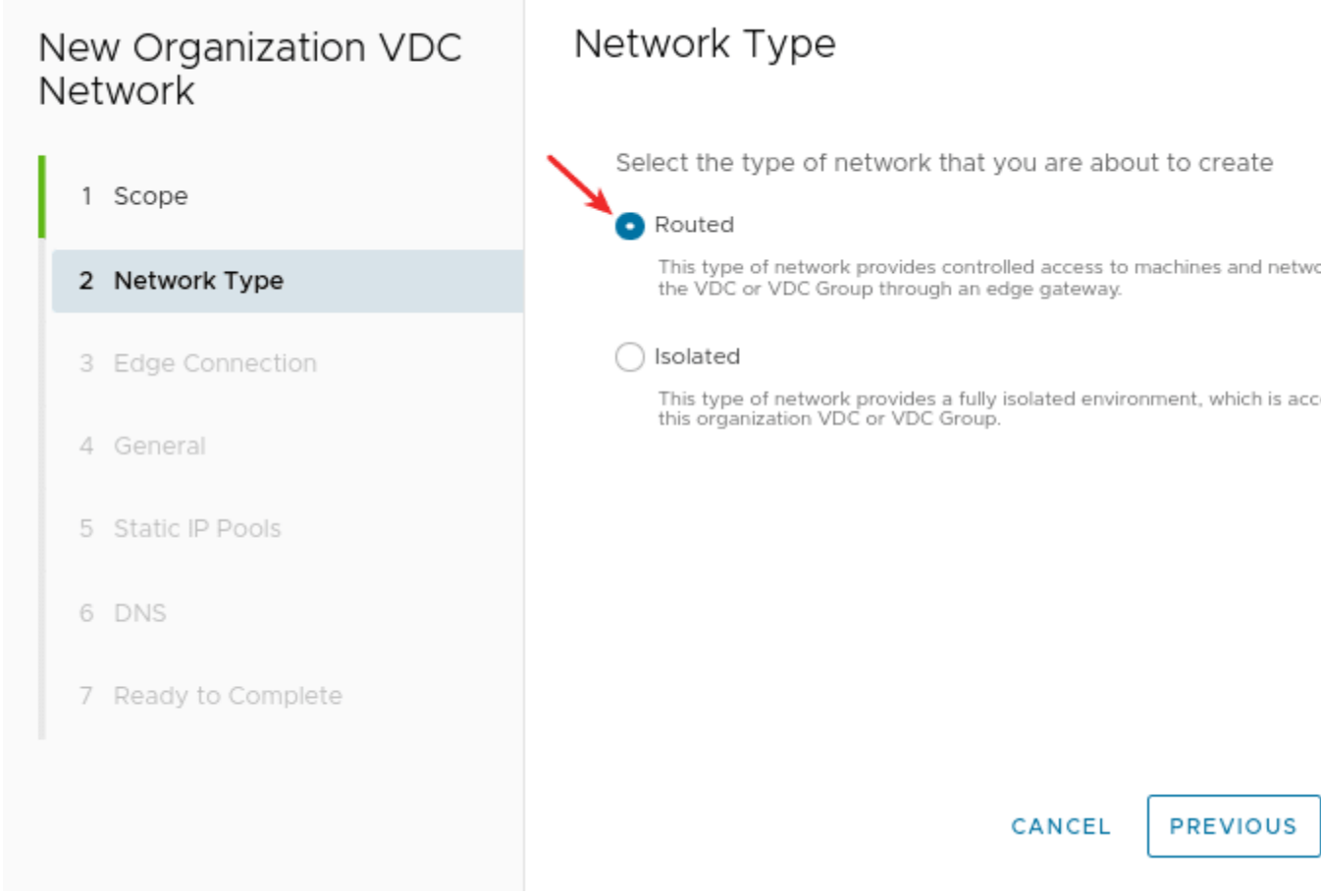
- 5

Отметьте пункт **Current Organization Virtual Data Center** и нажмите **Next**:



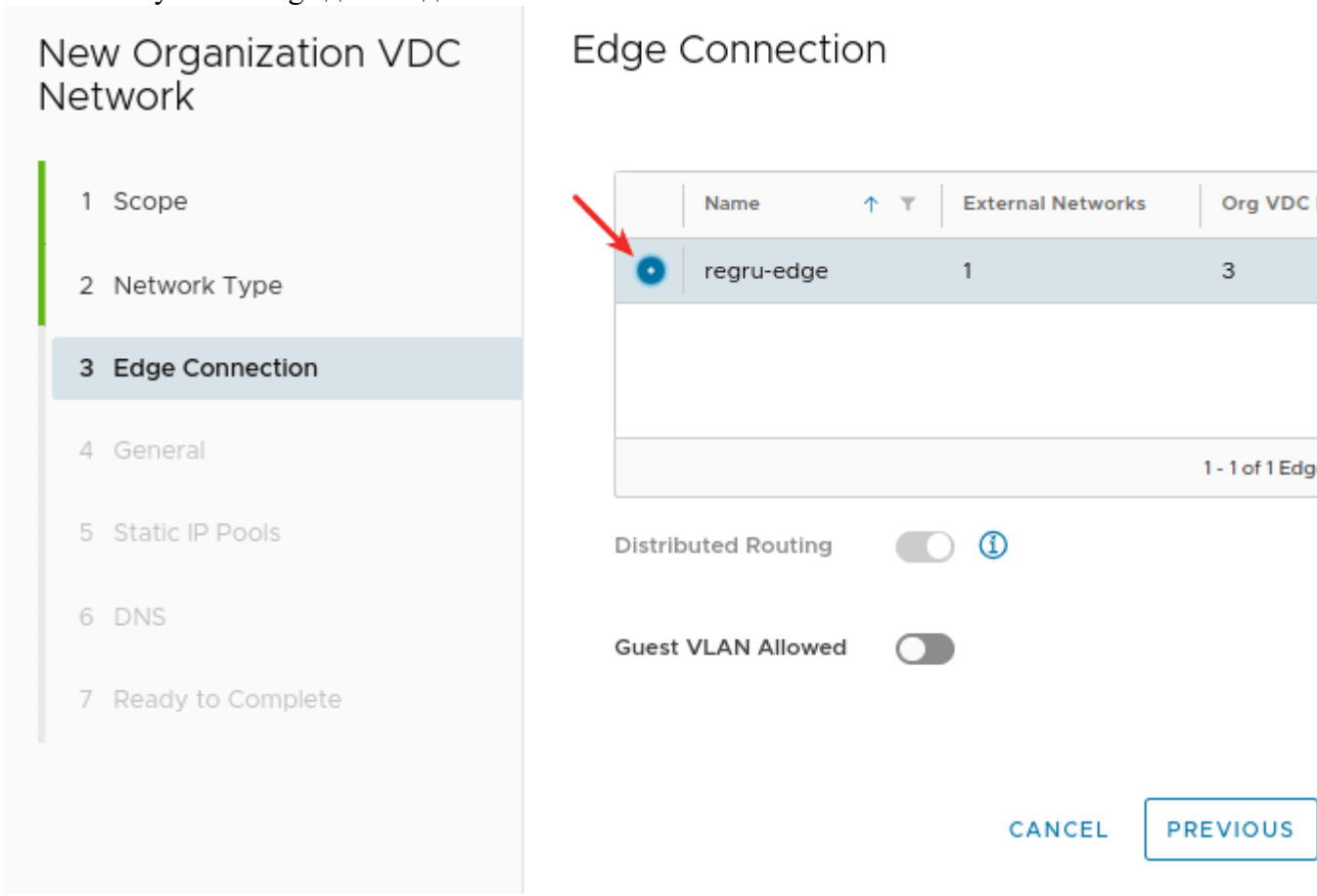
- 6

Выберите тип сети **Routed** и кликните **Next**:



- 7

Отметьте нужный Edge для соединения и кликните **Next**:

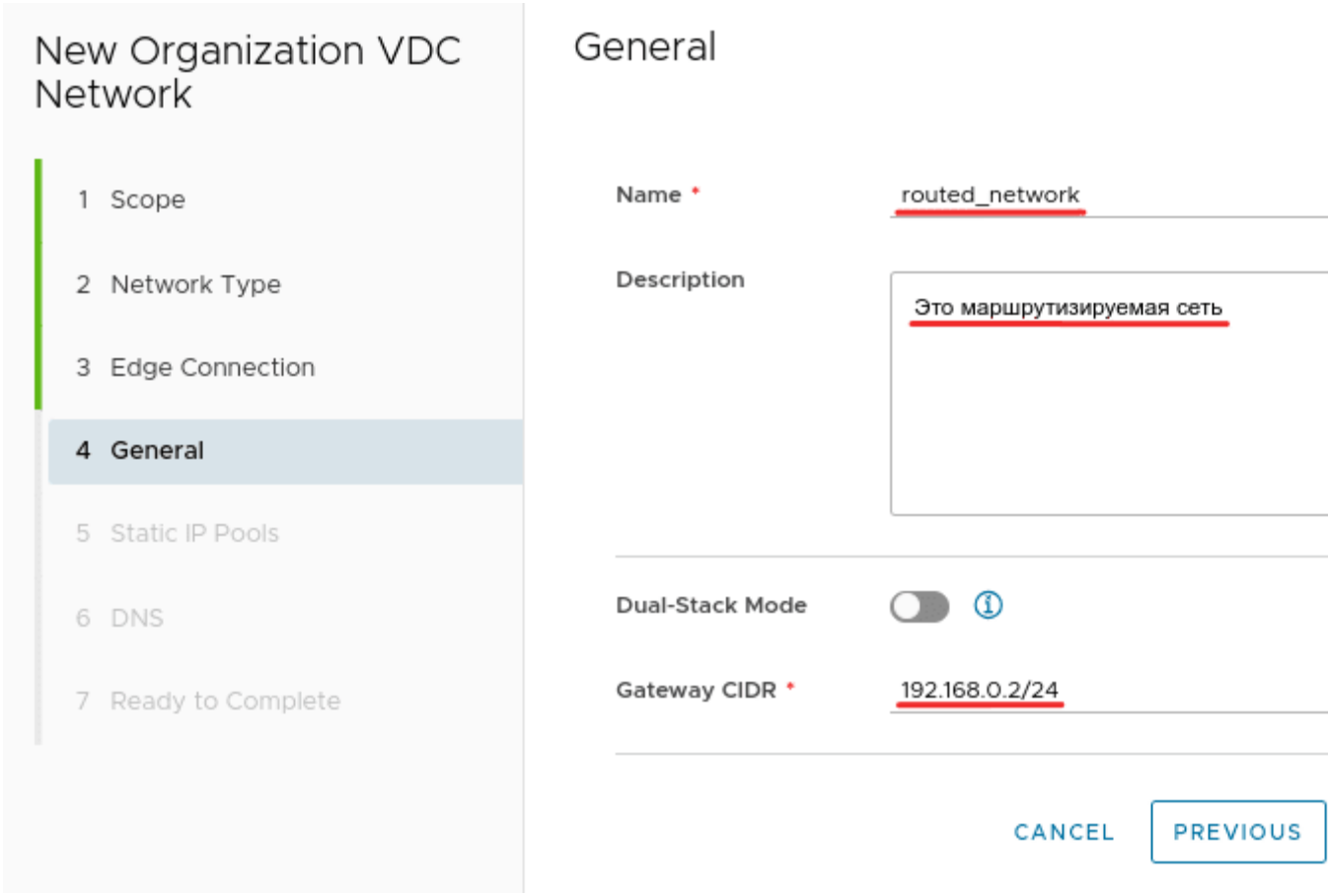


- 8

Заполните поля:

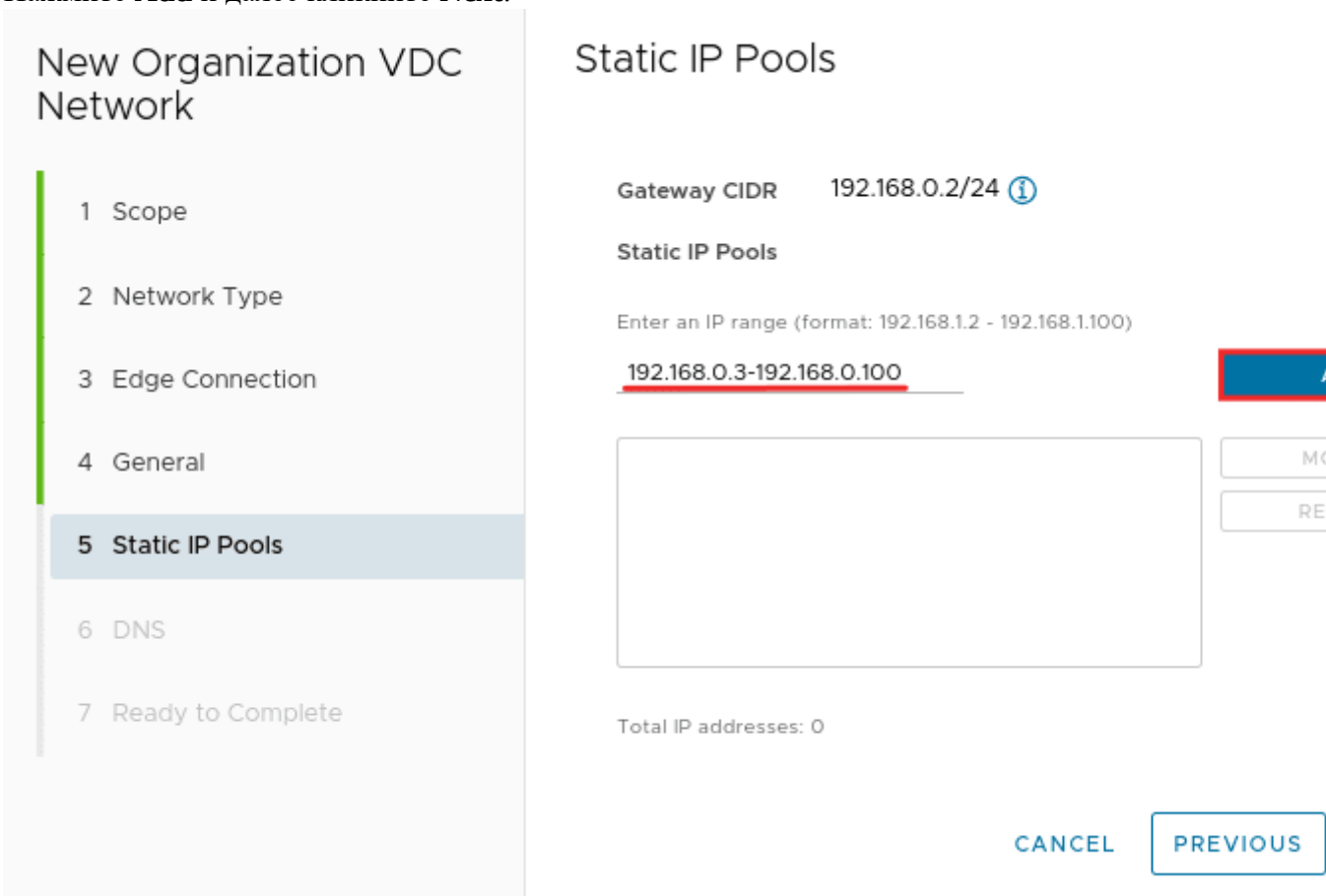
- **Name** — укажите название сети,
- **Description** — по необходимости добавьте описание сети,
- **Gateway CIDR** — укажите шлюз с маской подсети.

После этого нажмите **Next**:



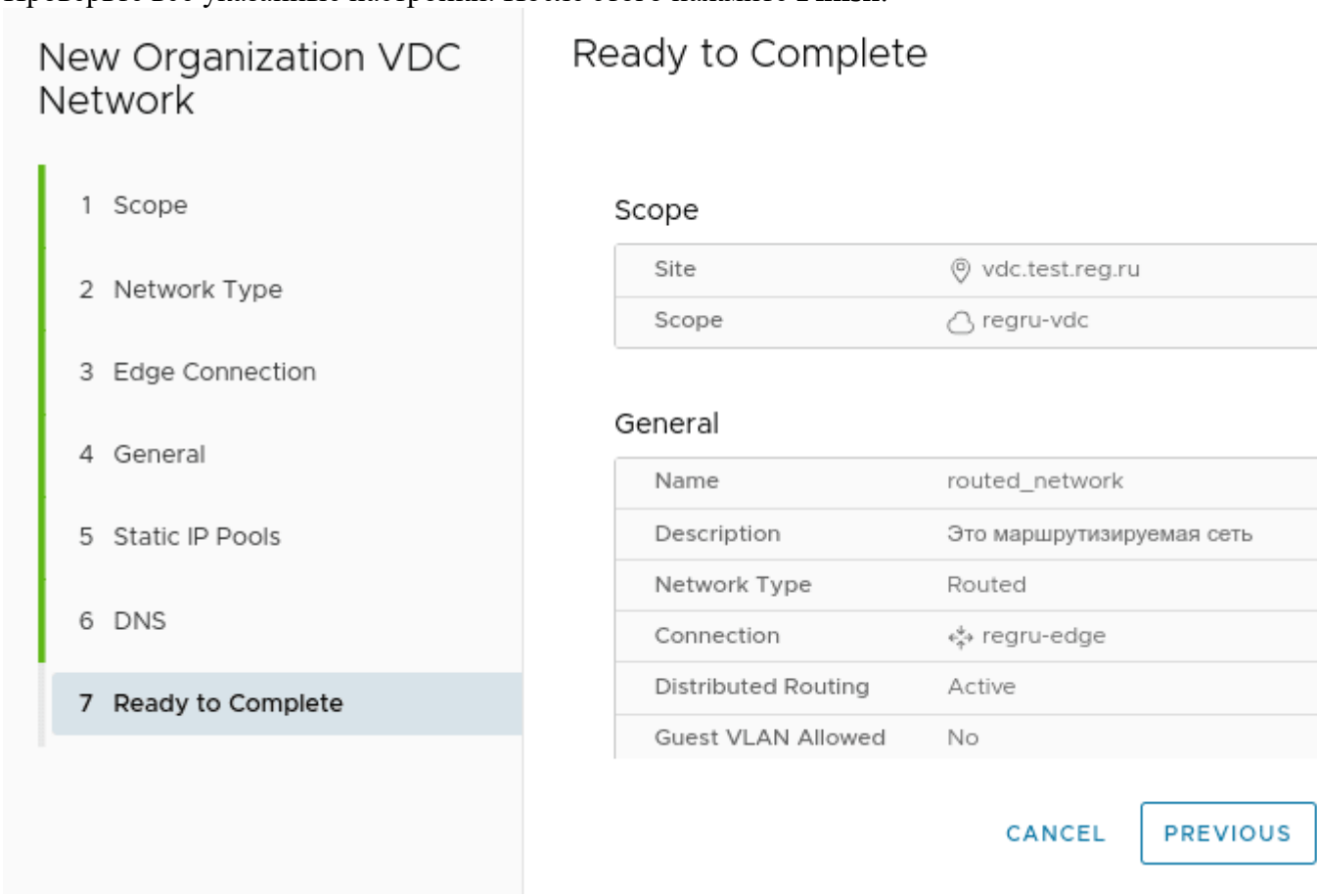
- 9

Укажите диапазон IP-адресов из подсети, которую вы вводили на предыдущем этапе. Нажмите **Add** и далее кликните **Next**:



- 10

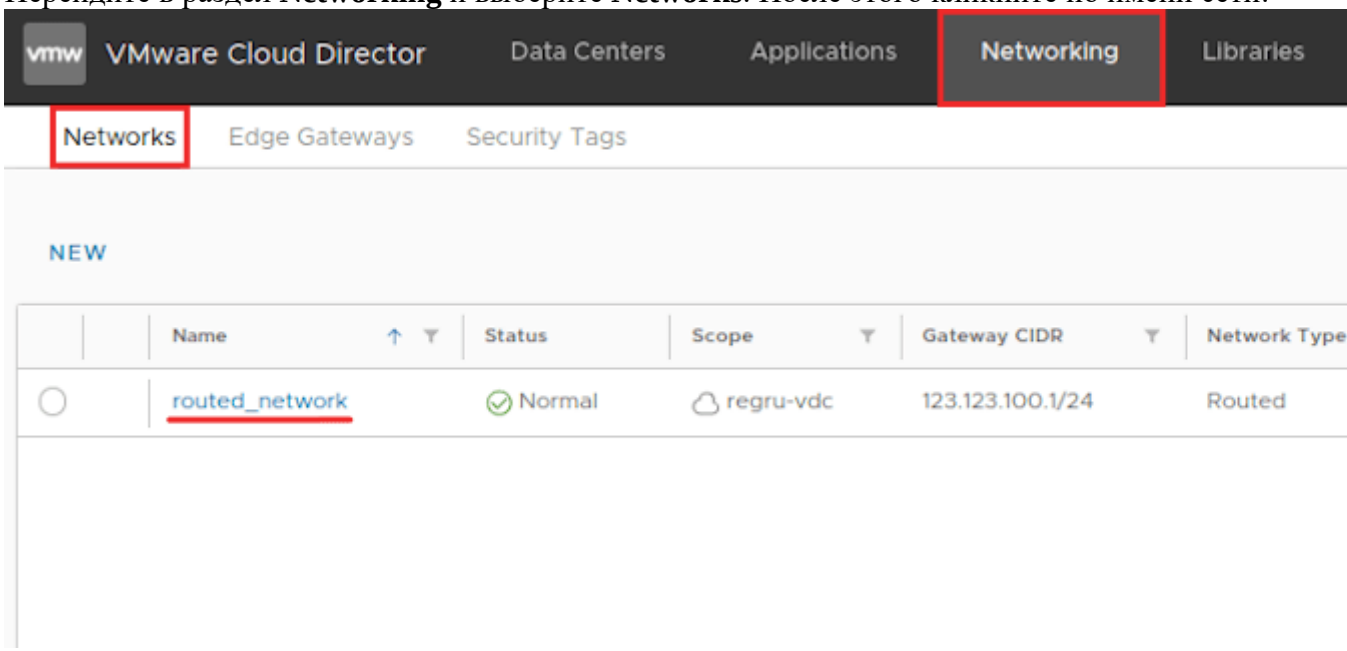
Проверьте все указанные настройки. После этого нажмите **Finish**:



По необходимости вы можете добавить автонастройку IP на виртуальных машинах при помощи DHCP. Для этого:

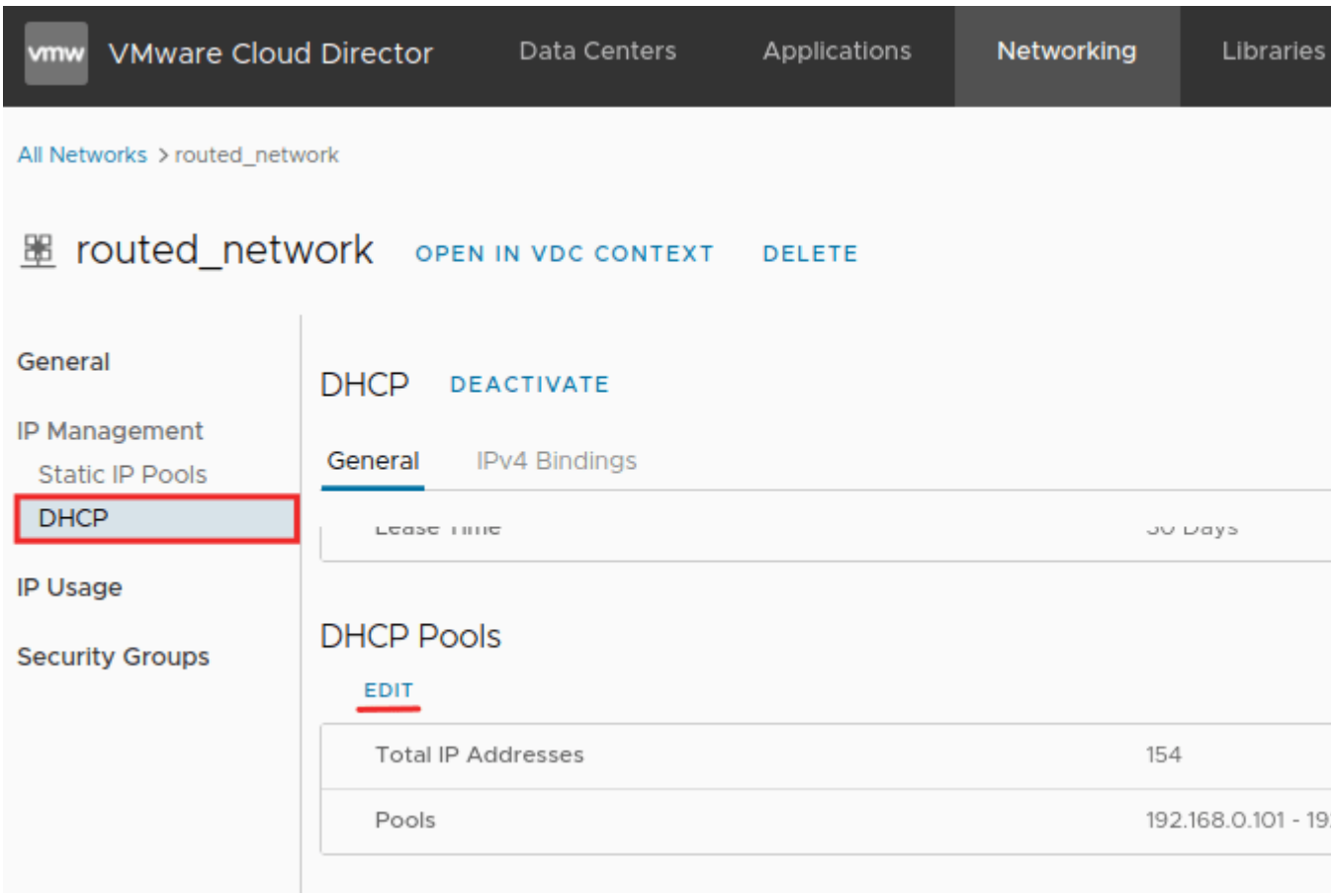
- 1

Перейдите в раздел **Networking** и выберите **Networks**. После этого кликните по имени сети:



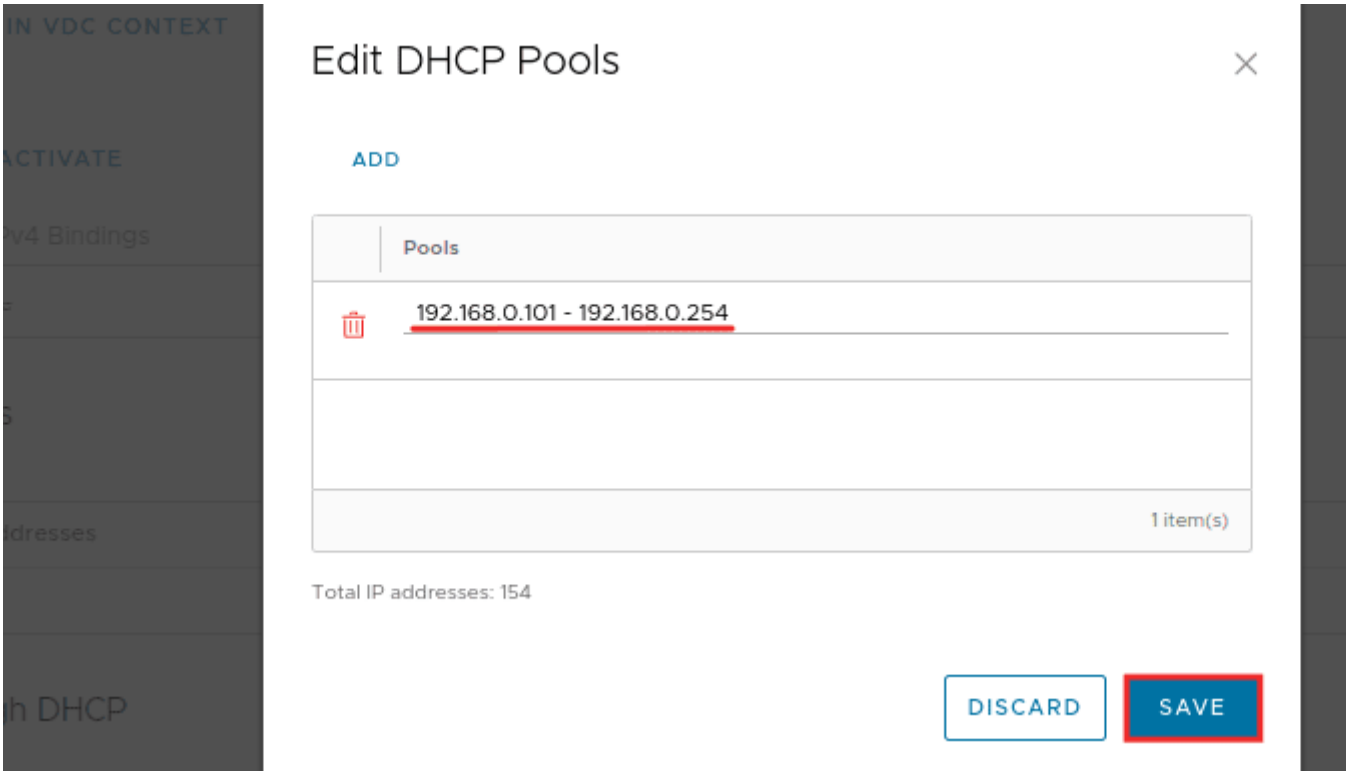
- 2

Перейдите в раздел **IP Management — DHCP**. В блоке «DHCP Tools» кликните **Edit**:



- 3

Укажите диапазон IP, которые будут выдаваться по DHCP.
Важно: пул адресов из блока DHCP должны отличаться от диапазона Static Pool. Например, если в Static Pool вы указали диапазон 192.168.0.3 — 192.168.0.100, то в DHCP можно добавить 192.168.0.101 — 192.168.0.254.
Затем нажмите **Save**:

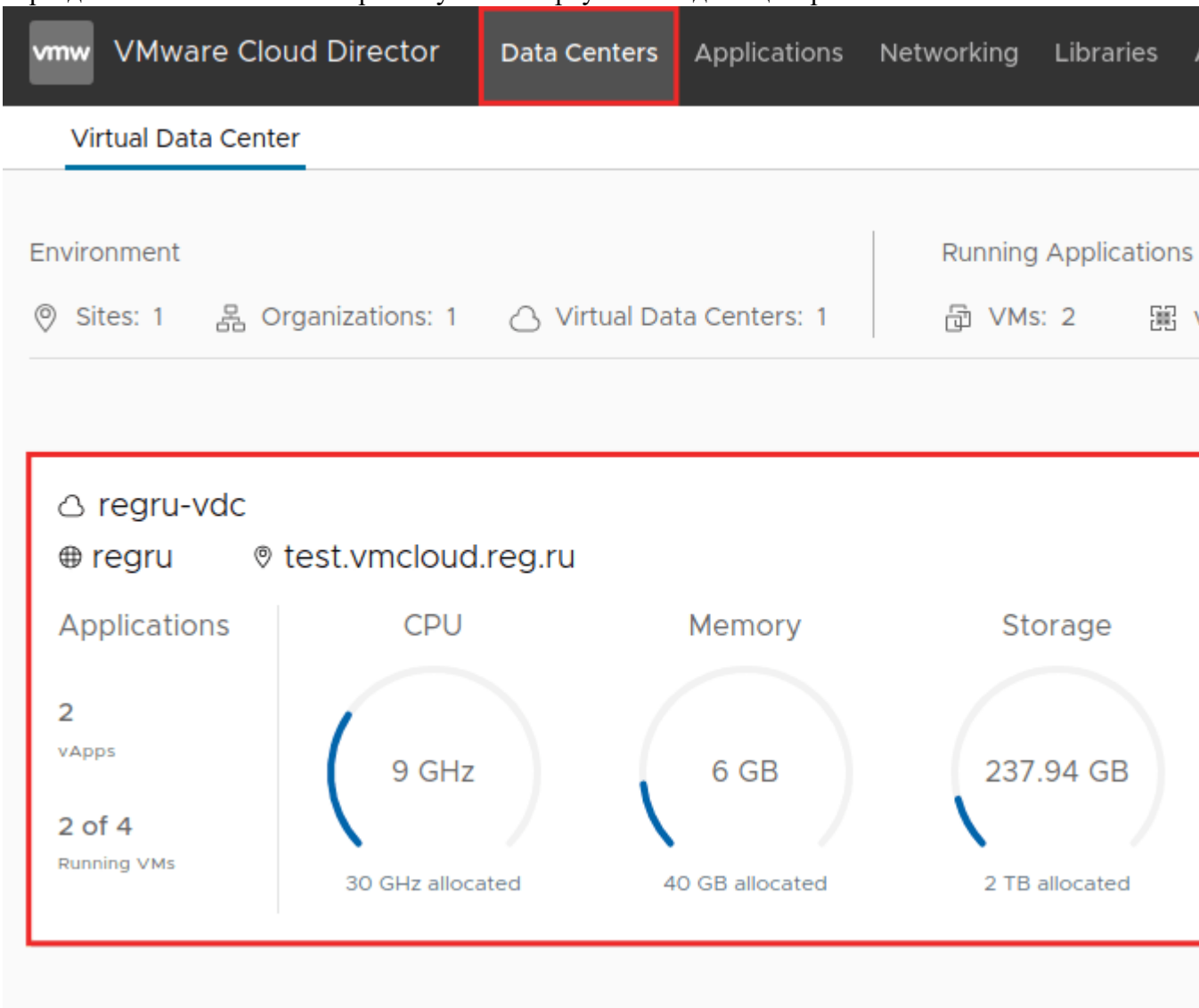


Готово, вы создали маршрутизируемую сеть.

Этап 2. Настройка подключения к интернету по NAT
Для подключения к интернету нужно настроить NAT, создав два правила: DNAT и SNAT. Настройка каждого из них описана ниже.
Настройка правила DNAT
DNAT — это правило, которое описывает маршрутизацию с внешнего IP на внутренние адреса. Для создания правила:

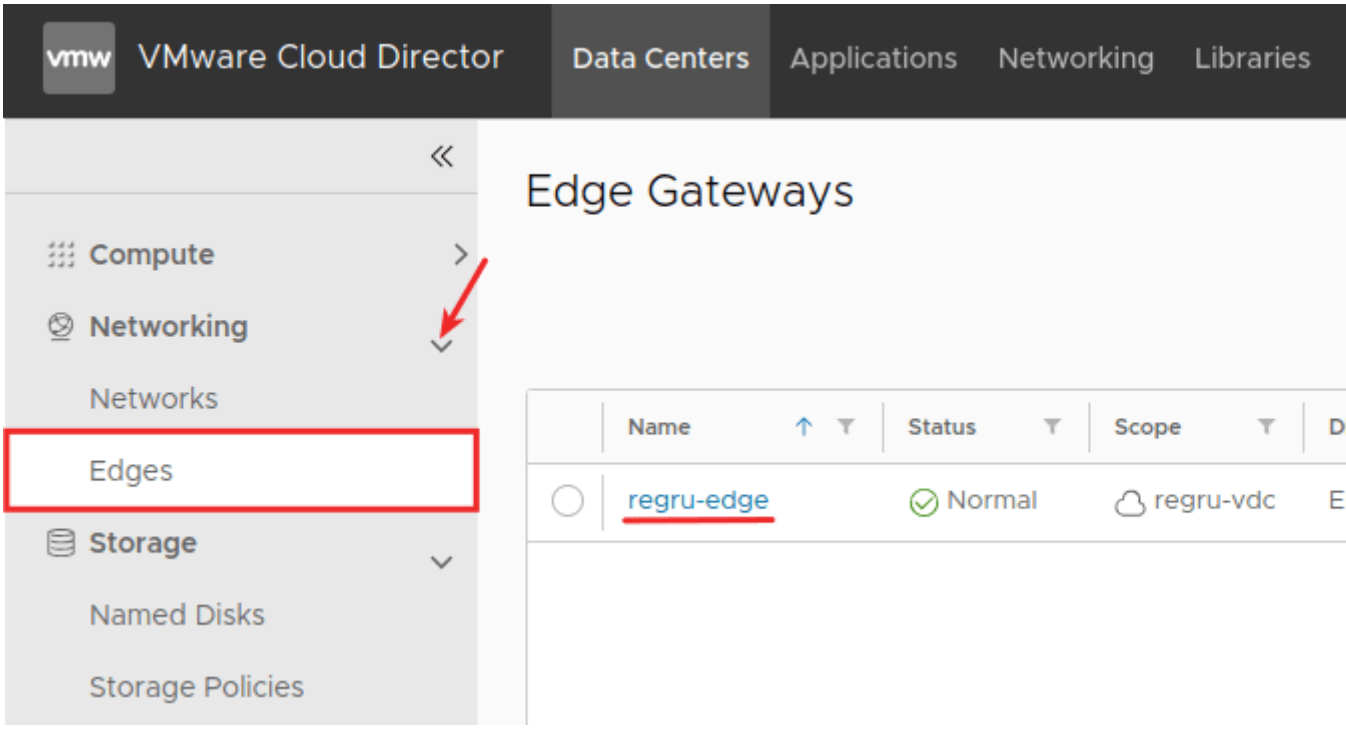
- 1Войдите в панель управления виртуальным дата-центром vCloud Director по инструкции.
- 2

В разделе «Datacenters» выберите нужный виртуальный дата-центр и кликните на него:



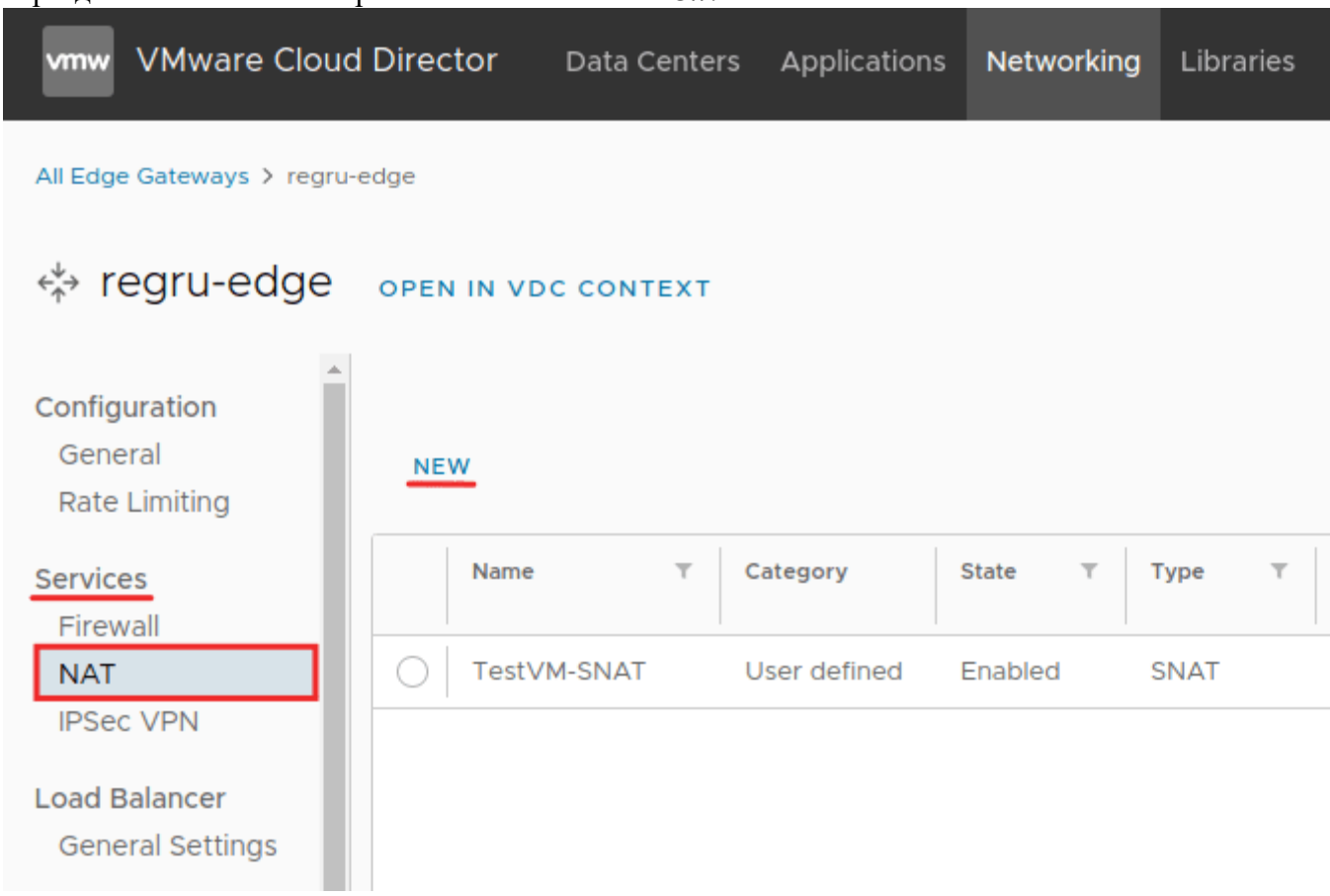
- 3

Разверните раздел **Networking** перейдите в блок **Edges**. Затем кликните на нужный Edge:



- 4

В разделе «Services» выберите **NAT** и нажмите **New**:



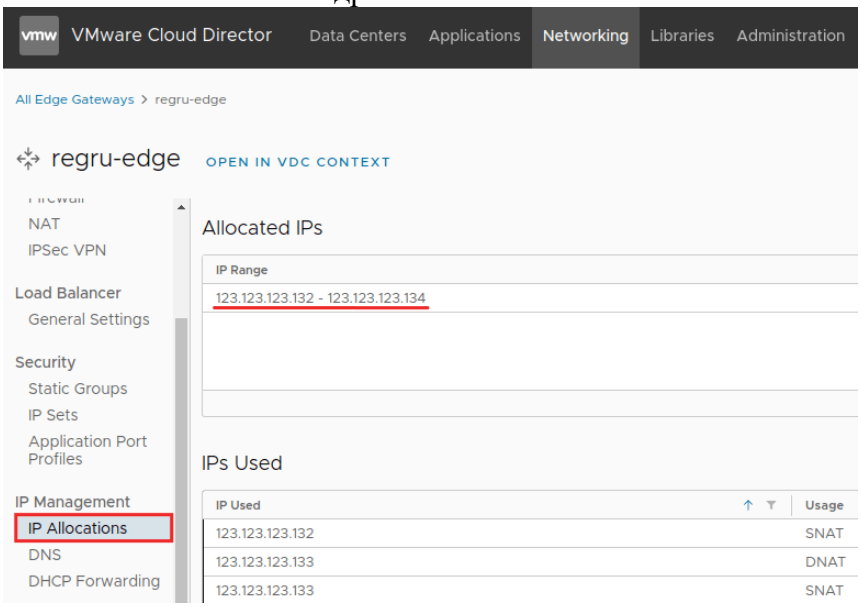
- 5

В появившемся окне введите данные для настройки доступа:

- **Name** — укажите название правила. Оно может быть любым;
- **Description** — добавьте описание правила по своему усмотрению;
- **Interface Type** — выберите тип DNAT;

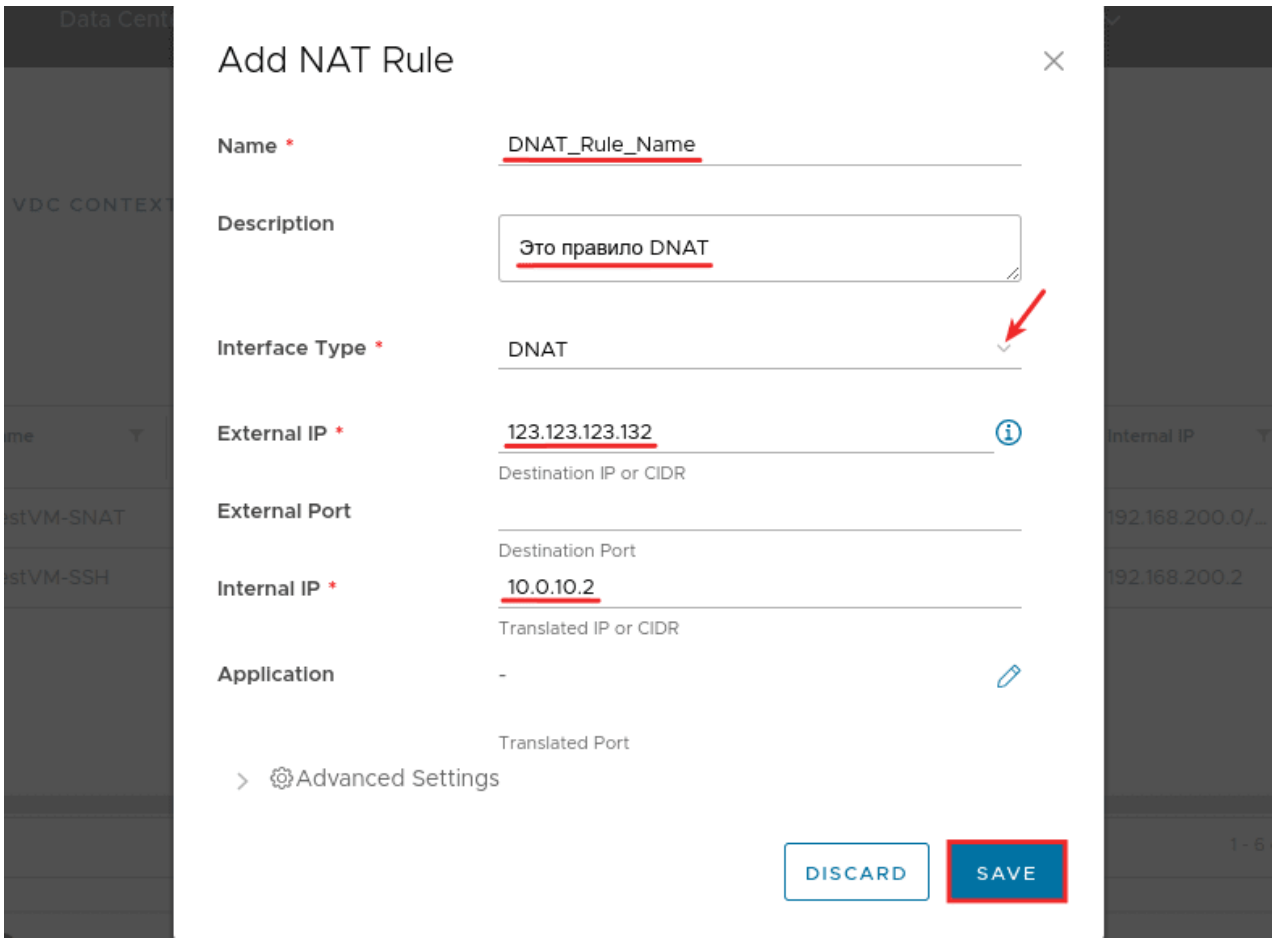
- **External IP** — введите внешний IP, к которому будут поступать запросы извне;

Как найти внешний IP-адрес



- **External Port** — укажите внешний порт. Если вам нужно перенаправить все порты с внешнего IP, оставьте поле пустым;
- **Internal IP** — внутренний IP-адрес или диапазон адресов виртуальных машин;
- **Application** — если вы заполняли поле **External Port**, укажите порт для приходящего трафика. Номер порта необходимо выбирать из списка приложений — в данном случае используется порт приложения по умолчанию. Для удобства вы можете использовать фильтрацию по номеру порта: для этого достаточно ввести номер порта и отметить нужное приложение. Если в списке нет приложения с нужным портом, добавьте его в разделе **Application Port Profile**.

После этого нажмите **Save**:

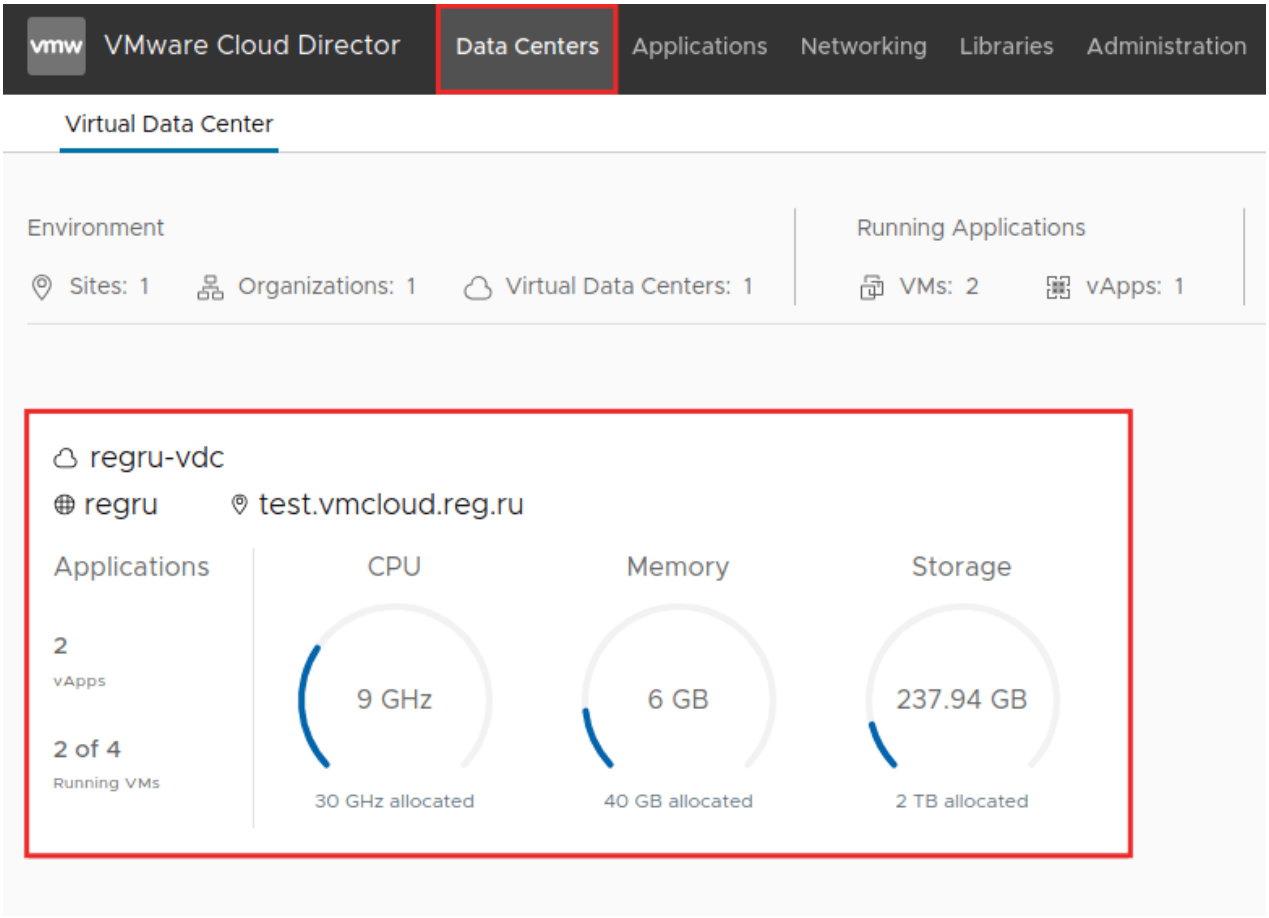


Настройка правила SNAT

SNAT — это правило, которое описывает маршрутизацию с внутренних адресов на внешний IP. Для создания правила:

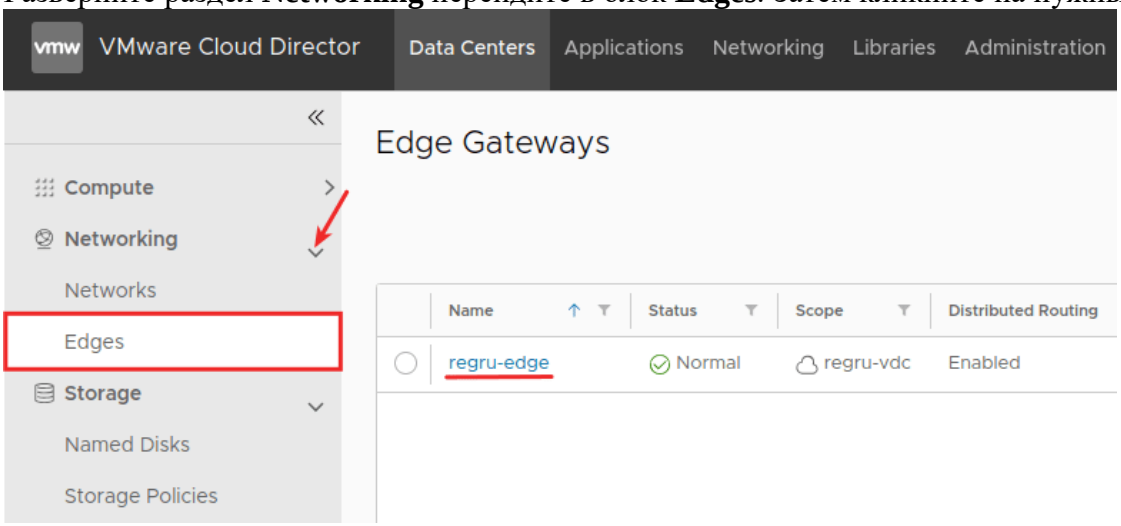
- 1Войдите в панель управления виртуальным дата-центром vCloud Director по инструкции.
- 2

В разделе «Datacenters» выберите нужный виртуальный дата-центр и кликните на него:



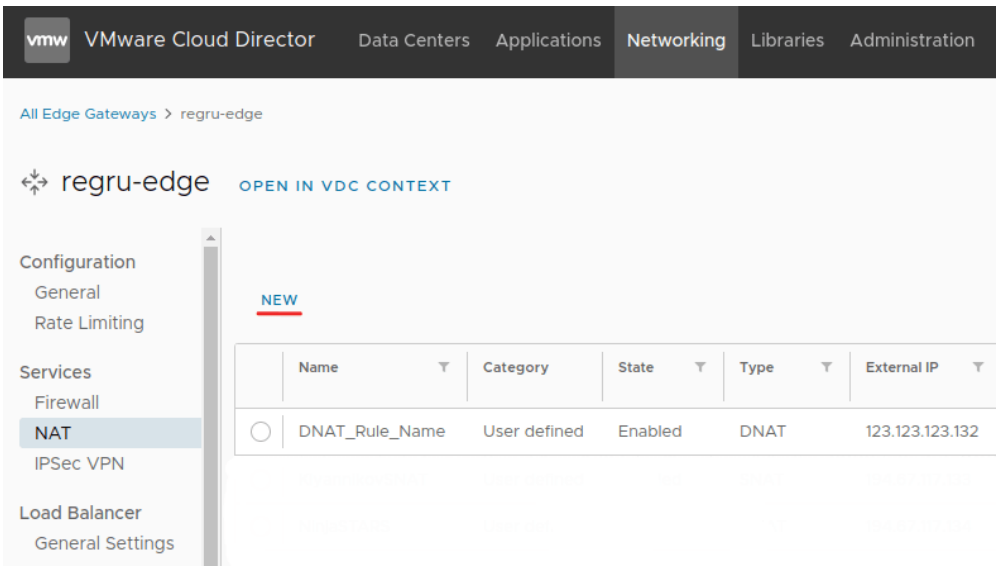
- 3

Разверните раздел **Networking** перейдите в блок **Edges**. Затем кликните на нужный Edge:



- 4

В разделе «Services» выберите **NAT** и нажмите **New**:



Практическая работа № 12 Установка облачного хранилища типа: файловое

Задание:

1. Сначала в систему нужно установить следующие пакеты:

```
sudo apt-get install apache2 php5 php-pear php-xml-parser php5-sqlite
 php5-json sqlite php5-mysql mp3info curl libcurl3 libcurl3-dev
 php5-curl zip
```

2. Потом качаем архив с сайта owncloud.org. Архив "весит" около мегабайта и представляет собой набор PHP-скриптов.

3. Распаковываем и перемещаем папку со скриптами в корневую директорию веб-сервера. В случае локального использования это будет, например, /var/www/owncloud

4. В каталоге с ownCloud нужно создать каталог data и установить на него права доступа 750, а также установить права доступа 777 на каталог config.

5. Ставим права доступа

```
sudo chown -R www-data:www-data /var/www/owncloud
```

6. Также необходимо отредактировать файл /etc/apache2/sites-enabled/000-default, где изменить "AllowOverride None" на "AllowOverride All". После чего перезапустить сервер

```
sudo /etc/init.d/apache2 restart
```

7. Системными средствами создаём отдельную базу данных mysql для нашего хранилища.

На этом установка закончена, остаётся только зайти через браузер по адресу localhost/owncloud и здесь ввести логин и пароль для нового администратора хранилища файлов и настройки базы данных mysql.

Практическая работа № 13 Установка облачного хранилища типа: блочное

Задание:

1. Обновление пакетов

Обновляем список пакетов:

apt update

На свежей системе также рекомендуется выполнить обновление установленных в системе пакетов:

apt upgrade

2. Время

Для нормального отображения времени создания и редактирования файлов, убедимся, что на нашем сервере стоит правильное время.

Установим сервис синхронизации времени и разрешим его автозапуск:

apt install chrony

systemctl enable chrony

Настраиваем временную зону:

timedatectl set-timezone Europe/Moscow

** В данном примере мы задаем зону по московскому времени.*

Список всех доступных зон можно посмотреть командой

timedatectl list-timezones

3. Настройка брандмауэра

По умолчанию, в Ubuntu брандмауэр пропускает все сетевые пакеты и не требует настройки. В этом случае, можно пропустить выполнение данного пункта.

Если же в вашем случае брандмауэр настроен на блокировку портов, выполняем нижеописанные действия.

Для нормальной работы системы нам нужно открыть 2 порта:

iptables -I INPUT -p tcp --dport 8000 -j ACCEPT

iptables -I INPUT -p tcp --dport 8082 -j ACCEPT

** на порту **8000** работает веб-сервер seahub; на **8002** будет слушать сервер seafile.*

Для сохранения правила используем утилиту iptables-persistent:

apt install iptables-persistent

netfilter-persistent save

Установка программных компонентов

Установим программные продукты, которые необходимы для работы Seafile.

Memcache

Начать необходимо с Memcache, так как его библиотеки нужны для установки пакетов python. Для установки вводим:

apt install memcached libmemcached-dev

Внесем небольшую корректировку в работу сервиса:

vi /etc/memcached.conf

Добавим памяти, которую может использовать memcached для своей работы:

-m 512

** до **512** Мб. Но можно и больше...*

Перезапускаем сервис и разрешаем его автозапуск:

systemctl restart memcached

systemctl enable memcached

Python

Для запуска и работы нашей облачной системы необходим python версии 3. Выполним его установку с дополнительными компонентами:

apt install python3 python3-setuptools python3-pip libmysqlclient-dev

** где:*

- ***python3** — интерпретатор для языка программирования python.*
- ***python3-setuptools** — дополнение для python-distutils, который в свою очередь, нужен для установки дополнительных модулей.*
- ***python3-pip** — менеджер установки пакетов python.*
- ***libmysqlclient-dev** — набор файлов для разработки под СУБД MySQL/MariaDB.*

С помощью менеджера пакетов python также установим:

pip3 install --timeout=3600 django==3.2.* Pillow pylibmc captcha jinja2 sqlalchemy==1.4.3 django-pylibmc django-simple-captcha python3-ldap mysqlclient pycryptodome==3.12.0 cffi==1.14.0

MariaDB

Как говорилось выше, мы будем использовать в качестве сервера баз данных MariaDB. Она может быть установлена из репозитория командой:

apt install mariadb-server

** подробнее об установке MariaDB читайте инструкцию Установка MariaDB на CentOS или Ubuntu.*

Установим пароль для основной учетной записи СУБД (root):

mysqladmin -u root password

** система запросит новый пароль. Его нужно ввести дважды.*

Чтобы пароль применился, нужно сбросить привилегии в СУБД. Для этого заходим в оболочку sql:

mysql -uroot -p

И вводим:

> flush privileges;

После выходим обратно:

> quit

Разрешаем автозапуск демона:

systemctl enable mariadb

Мы готовы переходить к установке Seafile.

Установка Seafile

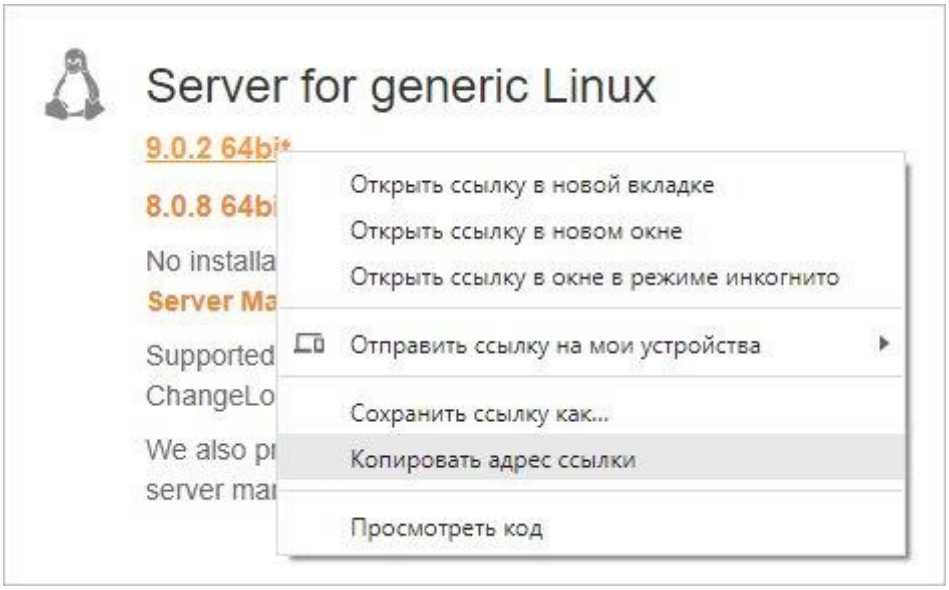
Создаем учетную запись seafile, от которой будет работать наш сервис:

useradd seafile -r

Создаем каталог, в который загрузим установочные файлы Seafile:

mkdir -p /opt/seafile/bin

На странице загрузки официального сайта копируем ссылку на архив с приложением:



С помощью скопированной ссылки загружаем архив:

wget https://s3.eu-central-1.amazonaws.com/download.seadrive.org/seafile-server_9.0.2_x86-64.tar.gz

И распаковываем его в каталог **/opt/seafile/bin**:

tar zxf seafile-server_*.tar.gz -C /opt/seafile/bin --strip-components 1

Назначим в качестве владельца каталога seafile и распакованных файлов ранее созданного пользователя:

chown -R seafile:seafile /opt/seafile

Заходим под пользователем seafile:

su seafile

Переходим в каталог с установочными файлами и запускаем скрипт:

$ cd /opt/seafile/bin

$ ./setup-seafile-mysql.sh

Мы увидим приветствие мастера установки. Просто нажимаем **ENTER**:

```
-----------------------------------------------------------------
This script will guide you to setup your seafile server using MySQL.
Make sure you have read seafile server manual at

                    https://download.seafile.com/published/seafile-manual/home.md

Press ENTER to continue
-----------------------------------------------------------------
```

Вводим имя сервера, который будут видеть клиенты:

What is the name of the server? It will be displayed on the client.
3 - 15 letters or digits
[server name] dmosk_seafile

Вводим IP-адрес или имя сервера, по которому можно подключиться к серверу:

What is the ip or domain of the server?
For example: www.mycompany.com, 192.168.1.101
[This server's ip or domain] seafile.dmosk.ru

Вводим порт, на котором должен слушать сервис или оставляем 8082:

Which port do you want to use for the seafile fileserver?
[default "8082"]

Выбираем, использовать имеющуюся базу или создать новую. Мы создадим новую:

Please choose a way to initialize seafile databases:
---

[1] Create new ccnet/seafile/seahub databases
[2] Use existing ccnet/seafile/seahub databases

[1 or 2] 1

Указываем настройки для подключения к СУБД — хост, порт, пароль для root:

What is the host of mysql server?
[default "localhost"]

What is the port of mysql server?
[default "3306"]

What is the password of the mysql root user?
[root password]

Задаем имя пользователя mysql, который будет создан для seafile (можно оставить предложенный вариант, нажав **ENTER**):

Enter the name for mysql user of seafile. It would be created if not exists.
[default "seafile"]

Задаем пароль для создаваемой учетной записи в mysql:

Enter the password for mysql user "seafile":
[password for seafile]

Вводим имя для базы данных для сервера ccnet:

Enter the database name for ccnet-server:
[default "ccnet-db"]

Вводим имя для базы данных для сервера seafile:

Enter the database name for seafile-server:
[default "seafile-db"]

Вводим имя для базы данных для сервера seahub:

Enter the database name for seahub:
[default "seahub-db"]

Смотрим на сводную информацию:

```
----------------------------------
This is your configuration
----------------------------------

        server name:                dmosk_seafile
      server ip/domain:             seafile.dmosk.ru

    seafile data dir:               /opt/seafile/seafile-data
        fileserver port:            8082

    database:                       create new
      ccnet database:               ccnet-db
       seafile database:            seafile-db
       seahub database:             seahub-db
       database user:               seafile

----------------------------------
Press ENTER to continue, or Ctrl-C to abort
----------------------------------
```

Если ошибок нет, вводим **ENTER** — начнется установке и конфигурирование.

Мы должны увидеть сообщение:

```
-----------------------------------------------------------------
Your seafile server configuration has been finished successfully.
-----------------------------------------------------------------

run seafile server:     ./seafile.sh { start | stop | restart }
run seahub server:      ./seahub.sh { start <port> | stop | restart <port> }

-----------------------------------------------------------------
If you are behind a firewall, remember to allow input/output of these tcp ports:
-----------------------------------------------------------------

port of seafile fileserver: 8082
port of seahub: 8000
```

When problems occur, Refer to

https://download.seafile.com/published/seafile-manual/home.md

for information.

Установка завершена.

Внесем изменения в конфигурационный файл:

$ vi /opt/seafile/conf/gunicorn.conf.py

Отредактируем строку

bind = "0.0.0.0:8000"

** в данном примере мы настроили наш сервер, чтобы он слушал на всех интерфейсах, а не только на локальном. Это нужно, чтобы он мог отвечать на сетевые запросы.*

Открываем другой конфигурационный файл:

vi /opt/seafile/conf/seahub_settings.py

Добавим строки:

```
CACHES = {
    'default': {
        'BACKEND': 'django_pylibmc.memcached.PyLibMCCache',
        'LOCATION': '127.0.0.1:11211',
    },
}
```

** в данном примере мы говорим серверу использовать memcached.*

Теперь нам нужно запустить 2 сервиса:

1. seafile fileserver — сервер, принимающий запросы от клиентов.
2. seahub — веб-сервер на python.

Стартуем первый:

$ /opt/seafile/bin/seafile.sh start

Мы должны увидеть:

```
Starting seafile server, please wait ...
** Message: 16:02:58.146: seafile-controller.c(621): No seafevents.

Seafile server started
```

Done.

Запускаем второй сервис:

$ /opt/seafile/bin/seahub.sh start

Система задаст несколько вопросов.

Адрес электронной почты администратора (это же будет логин администратора):

What is the email for the admin account?
[admin email] postmaster@dmosk.ru

Пароль, который будет использоваться для администратора (вводим дважды):

What is the password for the admin account?
[admin password]

Enter the password again:
[admin password again]

Мы должны увидеть:

Seahub is started

Done.

Сервер запущен.

Попробуем открыть браузер и перейти по адресу http://<IP-адрес сервера>:8000 — должна открыться страница авторизации. Вводим логин и пароль, который создали при старте seahub (в нашем примере для пользователя postmaster@dmosk.ru). Мы должны попасть в систему с правами администратора. Теперь у нас есть возможность управлять сервисом из графического интерфейса.