МИНОБРНАУКИ РОССИИ

Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» им. В. И. Ульянова (Ленина)

К. А. БОРИСЕНКО С. А. БЕЛЯЕВ

СЕТИ И ТЕЛЕКОММУНИКАЦИИ

Учебно-методическое пособие

Санкт-Петербург
Издательство СПбГЭТУ «ЛЭТИ»
2021

УДК 004.71(07)
ББК З 988.02-018я7
Б82

Борисенко К. А., Беляев С. А.

Б82 Сети и телекоммуникации: учеб.-метод. пособие. СПб.: Изд-во СПбГЭТУ «ЛЭТИ», 2021. 44 с.

ISBN 978-5-7629-2829-8

Содержит рекомендации по выполнению лабораторных работ по дисциплине «Сети и телекоммуникации». Рассматриваются такие вопросы настроек сетей, как создание сетевых адаптеров в виртуальной инфраструктуре, настройка IP-адресов, подключение к сети Интернет, маршрутизация трафика, трансляция сетевых адресов, создание сетей VLAN, а также регулирование трафика посредством сетевых экранов на примере iptables.

Предназначено для студентов направлений «Программная инженерия» и «Прикладная математика и информатика».

УДК 004.71(07)
ББК З 988.02-018я7

Рецензент: канд. техн. наук, доц. Е. Н. Шаповалов (Центр компетенций СПО АО «НИИ ПС»).

Утверждено
редакционно-издательским советом университета
в качестве учебно-методического пособия

ISBN 978-5-7629-2829-8

ВВЕДЕНИЕ

В пособии приводятся методические указания к лабораторным работам по дисциплине «Сети и телекоммуникации». Перечень лабораторных работ соответствует рабочей программе дисциплины и включает:

1. Создание и настройка виртуальной машины Ubuntu.
2. Изучение понятий IP-адреса и подсетей.
3. Изучение механизмов трансляции сетевых адресов: NAT, Masquerade.
4. Создание виртуальных локальных сетей VLAN.
5. Сетевые экраны. Iptables.

Информационные технологии (операционные системы, программное обеспечение общего и специализированного назначения, информационные справочные системы) и материально-техническая база, используемые при осуществлении образовательного процесса по дисциплине, соответствуют требованиям Федерального государственного образовательного стандарта высшего образования.

Для обеспечения образовательного процесса по дисциплине используются следующие информационные технологии:

1. Операционные системы: Microsoft Windows 10, Microsoft Windows 7, Ubuntu Server 16.04.
2. Программное обеспечение общего и специализированного назначения: Oracle VirtualBox.
3. Информационные справочные системы: международная ассоциация сетей «Интернет», электронные библиотечные системы и ресурсы удаленного доступа библиотеки СПбГЭТУ «ЛЭТИ».

Образовательный процесс обеспечивается на следующей материально-технической базе (по одному из трех вариантов):

1. Два персональных компьютера (системный блок RAMEC STORM, монитор LG L1953S, клавиатура, мышь, ИБП APC), сервер (RAMEC, монитор LG L1953S, клавиатура, мышь, ИПБ APC), проектор Mitsubishi XD430U, экран проекционный настенный.

2. Персональный компьютер (системный блок RAMEC STORM, монитор LG L1953S, клавиатура, мышь, ИБП APC), персональный компьютер (системный блок RAMEC STORM, монитор LG L222WS, клавиатура, мышь, ИБП

APC), принтер HP Laser Jet, принтер HP Color Laser Jet, сервер Super Micro, сканер планшетный HP Laser Jet, проектор Mitsubishi XD430U, экран проекционный настенный Screen Media Goldview 213x213 MW.

3. Персональный компьютер (системный блок Hewlett-Packard, монитор Samsung SyncMaster 913N, клавиатура, мышь), персональный компьютер (системный блок Hewlett-Packard, монитор Samsung SyncMaster E2220, клавиатура, мышь), проектор Vivitek D555, компьютер-сервер (системный блок Universal KOMPUMIR, монитор Samsung SyncMaster 913N, клавиатура, мышь), кондиционер Quattro Clima Industriale QA-RWD.

1. ЛАБОРАТОРНАЯ РАБОТА № 1. СОЗДАНИЕ И НАСТРОЙКА ВИРТУАЛЬНОЙ МАШИНЫ UBUNTU

1.1. Цель и задачи

Целью работы является подготовка компьютера и установка виртуальной машины с операционной системой (ОС) Ubuntu Server (https://ubuntu.com/) для выполнения следующих лабораторных работ. Необходимо решить следующие задачи:

1. Настроить компьютер для работы с технологией виртуализации.
2. Скачать и установить ОС Ubuntu Server 16.04 x64 (https://releases.ubuntu.com/).
3. Настроить и проверить различные типы подключения интерфейсов.
4. Клонировать созданную машину.

1.2. Основные теоретические сведения

Настройка компьютера для установки виртуальных машин.

Введем следующие понятия:

1. Хост – компьютер, на котором устанавливаются виртуальные машины.
2. Виртуальная машина – программная система, эмулирующая аппаратное обеспечение некоторой платформы (гостевая платформа) и исполняющая программы для гостевой платформы, установленная на хосте.

При выполнении лабораторных работ будет использоваться программный продукт Oracle VirtualBox. Допускается использовать другую систему для запуска виртуальных машин, но она должна отвечать следующим основным требованиям:

1) поддерживаются типы подключений интерфейсов: NAT, внутренняя сеть (internal network);

2) поддерживается возможность клонирования виртуальной машины.

По умолчанию в лабораторных работах рассматриваются ОС с разрядностью x64, однако если хостовый компьютер недостаточно мощный или количество оперативной памяти меньше 3 Гб, тогда можно использовать ОС с разрядностью x86 – на выполнении лабораторных работ это не отразится.

На хосте необходимо включить поддержку виртуализации. Выполнить это необходимо в BIOS, причем в каждой версии BIOS настройка своя. На процессорах Intel при входе в BIOS необходимо зайти в пункт настройки «CPU Configuration». Этот раздел может находиться в «Advanced» либо в «Integrated Peripherals». Внутри настроек CPU должна быть строчка, похожая на «Intel

Virtualization Technology». Ее необходимо активировать либо выбрать «Enable». На процессорах AMD в BIOS необходимо выполнить настройки в «Advanced» в «CPU Configuration». Там необходимо настроить пункт «SVM Mode», включив его («Enabled»). После этого необходимо сохранить изменения и перезагрузить компьютер.

При установке виртуальной машины разрядности x64 может появиться ошибка 0x80004005 (либо другая) с информацией о проблемах, связанных с виртуализацией, это означает, что были выполнены не все настройки в BIOS. В таком случае необходимо найти соответствующие пункты, связанные с использованием виртуальных машин. Для поиска можно, зная версию BIOS, найти в Интернете информацию о процессе настройки виртуализации. Если настроить данный функционал не получится, допускается использовать ОС Ubuntu Server x86. Важно: на компьютерах в компьютерном классе необходимые настройки BIOS выполнены в полном объеме.

В ОС Ubuntu LTS 18.04 и более новых версиях для работы с сетью используется `network-manager`, лабораторные работы будут выполняться с использованием `ifconfig` и `/etc/network/interfaces`. Данные утилиты используются, например, в ОС Ubuntu 16 и Ubuntu 14. Лабораторные работы подготовлены на базе ОС Ubuntu LTS 16.04, однако может быть использована и более старая версия ОС.

Лабораторные работы выполняются в терминальном режиме, поэтому достаточно устанавливать только серверную версию ОС. Рекомендуемый объем оперативной памяти – 512 Мб, но если объема оперативной памяти на хосте недостаточно, то после установки виртуальной машины можно установить 256 Мб.

В процессе создания виртуальной машины необходимо настроить количество используемых центральных процессоров, достаточно выбрать один на каждую виртуальную машину. Также необходимо настроить виртуальный жесткий диск. Рекомендуется выбрать VDI (Virtual Disk Image), динамический жесткий диск с объемом 10 Гб и выше.

В качестве типа подключения сетевого интерфейса для начальной настройки достаточно выбрать NAT или сетевой мост. Это позволит виртуальной машине иметь доступ в Интернет для настройки и скачивания необходимых приложений.

Дополнительно для удобства доступа к терминалу виртуальной машины при наличии технической возможности следует использовать тип подключения «сетевой мост». Это позволит виртуальной машине получить IP-адрес из

подсети, которая является общей с хостом. Установив на хост программу Putty, можно подключиться по ssh к виртуальной машине, указав ее IP-адрес в настройках подключения Putty. Узнать IP-адрес можно, выполнив на машине команду `ifconfig`.

Если у вас нет возможности получить IP-адрес из той же подсети, то можно настроить два сетевых интерфейса: первый с NAT для выхода в Интернет, второй – виртуальный адаптер хоста (VirtualBox Host-Only Ethernet Adapter). Благодаря ему будет использоваться виртуальная локальная сеть между хостом и виртуальной машиной и она будет доступна по IP-адресу из этой подсети к подключению по ssh.

Настройка ОС Ubuntu после установки. После установки ОС Ubuntu следует настроить необходимый функционал.

Нужно снизить время ожидания получения сетевых настроек системой в период загрузки ОС. При неверных настройках или их отсутствии (по умолчанию) ОС будет запускаться несколько минут. Чтобы уменьшить время запуска, необходимо выполнить в терминале следующие команды:

```
sudo mkdir -p /etc/systemd/system/networking.service.d/
sudo bash -c 'echo -e "[Service]\nTimeoutStartSec=20sec" >
/etc/systemd/system/networking.service.d/timeout.conf'
sudo systemctl daemon-reload
```

Для доступа в систему по ssh (например, используя Putty) необходимо установить на виртуальной машине ssh-сервер:

```
sudo apt-get update
sudo apt-get install openssh-server
```

Также стоит отметить, что для применения настроек сети необходимо перезагружать виртуальную машину, так как перезагрузка сетевого сервиса поможет не во всех случаях. Выполнить это можно командой `sudo reboot`, после чего через 20…30 с установить новое ssh-подключение с хоста на виртуальную машину.

После выполнения перечисленных действий виртуальная машина будет готова к использованию в следующих лабораторных работах. Для их выполнения можно осуществить клонирование настроенной виртуальной машины. Для этого необходимо выключить виртуальную машину и в меню VirtualBox найти пункт «Клонировать». В процессе клонирования необходимо задать новое имя, в политике MAC-адресов рекомендуется выбрать «Сгенерировать новые MAC-адреса для всех сетевых адаптеров». На следующем шаге нужно

выбрать «полное клонирование». Такие настройки создадут отдельную независимую копию виртуальной машины. В результате появится новая машина, которая будет полностью идентична исходной виртуальной машине и может быть использована.

1.3. Общая формулировка задач

Требуется выполнить следующие задачи:

1. Настроить компьютер для работы с технологией виртуализации.
2. Скачать и установить операционную систему Ubuntu Server 16.04 x64.
3. Настроить и проверить типы подключения интерфейсов: NAT, виртуальный адаптер хоста, сетевой мост.
4. Клонировать созданную машину.

1.4. Последовательность выполнения работы

Данную работу следует выполнять строго в последовательности, указанной в разделе «Общая формулировка задач».

Информацию для выполнения каждой задачи можно найти в разделе «Основные теоретические сведения».

В задаче по настройке типов подключения интерфейсов в отчете необходимо продемонстрировать результаты настроек сетевых адаптеров при каждом из типов. Для отслеживания полученных IP-адресов используется команда `ifconfig` и анализируется ее результат. Выводы необходимо отразить в отчете.

1.5. Контрольные вопросы

1. Как настраивается поддержка виртуализации в процессорах Intel и AMD?
2. С помощью чего можно подключиться к виртуальной машине с хоста?
3. Какая команда выводит результаты настройки сетевых интерфейсов?
4. Какие возможности дает утилита openssh-server?
5. Что означает термин «клонирование машины»?
6. Как применить настройки изменения сети в ОС Ubuntu 16.04?

2. ЛАБОРАТОРНАЯ РАБОТА № 2. ИЗУЧЕНИЕ ПОНЯТИЙ IP-АДРЕСА И ПОДСЕТЕЙ

2.1. Цель и задачи

Целью работы является изучение IP-адресации (IPv4), логического построения локальных сетей. Необходимо решить следующие задачи:

1. Создать две виртуальные машины (лаб. работа № 1).
2. Определить адрес сети по IP и маске.
3. Определить широковещательный IP-адрес для конкретной подсети.
4. Определить принадлежность IP-адресов к одной подсети.
5. Построить схему сети с использованием различных масок и IP-адресов.
6. Проверить п. 4 на реальной инфраструктуре, построенной в VirtualBox.

2.2. Основные теоретические сведения

Понятие IP-адреса. IP-адрес (Internet Protocol Address) – уникальный адрес для обозначения устройств, например персональных ПК, мобильных телефонов, серверов, служащий для идентификации устройства в сети и общения с другими устройствами в сети. Каждое устройство, подключенное к сети Интернет, должно иметь уникальный IP-адрес внутри сети. Другими словами, IP-адрес аналогичен домашнему адресу и телефонному номеру, с помощью которых можно однозначно определить человека.

В рамках данного пособия будет изучен IPv4. Он использует 32-битное число для представления IP-адреса. 32 бита дают возможность получить около 4 млрд уникальных адресов, поэтому нет возможности выделить всем отдельные IP-адреса внутри одной сети. Для решения данной проблемы служит трансляция IP-адресов (NAT, Masquerade). Также существует версия протокола IPv6, которая может предоставить значительно большее количество уникальных адресов.

Обычно IP-адрес представляется в десятичном виде из четырех наборов чисел, каждый из которых хранит в себе восьмибитное число. Так как число восьмибитное, то в десятичной системе данное значение может быть от 0 до 255.

Маска подсети. Сетевые устройства логически объединяются в подсети. Каждому сетевому устройству назначаются IP-адрес и маска подсети (netmask). Маска подсети в IPv4 состоит из 32 бит, непрерывной последовательности единиц (1), за которой следует непрерывная последовательность нулей (0). В маске подсети не может стоять единица после нуля. Маска подсети служит

для разделения IP-адресов на подсети и определения того, какие из них будут принадлежать каждой подсети.

IP-адрес делится с помощью маски на адрес подсети (префикс сети) и адрес узла. Рассмотрим пример:

```
IP-адрес: 192.168.0.1
Маска: 255.255.255.0
```

Поэтому ее можно записывать приставкой к IP-адресу, например: 192.168.0.1/24. В данном случае 24 – это количество единиц в маске. Такая запись называется CIDR и может принимать значения от 1 до 32, по количеству единиц в 32-битном значении маски.

Для вычисления адреса сети по адресу узла необходимо перевести IP-адрес и маску в двоичную систему, например:

```
192.168.0.1   →   11000000.10101000.00000000.00000001
255.255.255.0 →   11111111.11111111.11111111.00000000
```

Сетевой префикс (адрес сети) вычисляется побитовой операцией AND между IP-адресом и маской. В приведенном примере это первые три четверти записи IP-адреса, поэтому префикс подсети – 192.168.0.0. Адресом узла является оставшаяся часть, которая соответствует нулям маски – 0.0.0.1.

Принадлежность IP-адресов одной подсети. Для того, чтобы выяснить, принадлежат ли два IP-адреса одной подсети, необходимо выполнить следующее. Во-первых, их маски должны быть одинаковы. Во-вторых, нужно перевести IP-адреса и маску в двоичный вид:

```
253.45.78.14   →   11111101.00101101.01 | 001110.00001110
253.45.126.14  →   11111101.00101101.01 | 111110.00001110
255.255.192.0  →   11111111.11111111.11 | 000000.00000000
```

Применив к каждому IP-адресу и маске побитовое AND, мы получаем адреса подсети для каждого IP-адреса. Если они совпадают, это значит, что эти два IP-адреса принадлежат одной подсети (как в варианте выше). В случае, если в результате адреса подсети разные – IP-адреса принадлежат разным подсетям:

```
156.14.32.78       →   10011100.00001110.00100000.01 | 001110
156.14.32.130      →   10011100.00001110.00100000.10 | 000010
255.255.155.192    →   11111111.11111111.11111111.11 | 000000
```

Если настроить IP-адреса на два ПК и физически соединить их (коммутатор, напрямую кабелем, объединить в одну сеть в VirtualBox логически и др.), то проверить принадлежность их к общей сети можно с помощью выполнения следующей команды в терминале:

```
ping <ip-адрес>
```

Здесь в качестве <ip-адрес> подставляется IP-адрес второго ПК. Если ответ приходит – они в одной подсети, если нет – в разных.

Пример успешного ответа:

```
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=45 time=4.55 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=45 time=4.57 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=45 time=4.57 ms
```

Пример неуспешного ответа:

```
PING 192.168.254.30 (192.168.254.30) 56(84) bytesofdata.
From 192.168.76.1 icmp_seq=1 Destination Host Unreachable
From 192.168.76.1 icmp_seq=2 Destination Host Unreachable
From 192.168.76.1 icmp_seq=3 Destination Host Unreachable
```

Также система может выдать сообщение «Network is unreachable». Это произойдет в случае, если местонахождение узла неизвестно.

Широковещательный IP-адрес. Широковещательный IP-адрес (broadcast) нужен для того, чтобы отправить данные всем узлам подсети. Например, протокол ARP, который связывает MAC-адрес устройства и IP-адрес, использует для этого широковещательный IP-адрес. Для того, чтобы получить широковещательный IP-адрес, необходимо «перевернуть» значения маски и применить побитовое OR между IP-адресом и маской:

```
156.14.32.78        →   10011100.00001110.00100000.01 | 001110
255.255.255.192     →   00000000.00000000.00000000.00 | 111111
Broadcast           →   10011100.00001110.00100000.01 | 111111
→156.14.32.127.
```

Отправляя пакет с указанным IP-адресом, система обеспечит его получение всеми узлами, находящимися в данной подсети.

Зарезервированные IP-адреса. Есть пул IP-адресов, которые назначать сетевому устройству нельзя: 0.0.0.0/8 (если первый октет «0»), 127.0.0.0/8 (интерфейс loopback), 224.0.0.0/4 (служебная подсеть для групповой рассылки), 240.0.0.0/4 (резервные IP-адреса) и 255.255.255.255 (широковещательный IP-адрес). Если настроить IP-адрес из этих подсетей, то сетевое устройство может работать неверно или вообще быть недоступным.

2.3. Общая формулировка задач

Необходимо решить следующие задачи:

1. Определение принадлежности IP-адресов к одной подсети. Развернуть две виртуальные машины (лаб. работа № 1), выбрать тип подключения сетевого адаптера «intnet» и выполнить следующие операции:

a. Получить два IP-адреса с маской у преподавателя. Пример IP-адресов:

```
221.238.65.231/10
221.247.74.240/10
```

b. Для полученных IP-адресов определить, относятся они к одной подсети или нет. Представить процесс вычислений в отчете.

c. Настроить IP-адреса из п. a для созданных виртуальных машин и проверить их доступность с использованием команды ping. Результат должен совпасть с п. b.

d. Если IP-адреса не принадлежат одной подсети для подсети, в которой находится первый IP-адрес, придумать IP-адрес, который будет принадлежать данной подсети, настроить вторую виртуальную машину с использованием придуманного IP-адреса и продемонстрировать успешное выполнение ping с одной виртуальной машины к другой.

e. Для каждого IP-адреса указать адрес подсети, широковещательный IP-адрес.

Варианты для создания четырех подсетей

№	CIDR 1	CIDR 2	CIDR 3	CIDR 4	№	CIDR 1	CIDR 2	CIDR 3	CIDR 4
1	6	18	22	1	26	1	13	24	26
2	12	24	1	2	27	7	19	3	27
3	18	30	11	3	28	13	25	13	28
4	24	5	21	4	29	19	0	23	29
5	30	11	0	5	30	25	6	2	0
6	5	17	10	6	31	0	12	12	1
7	11	23	20	7	32	6	18	22	2
8	17	29	30	8	33	12	24	1	3
9	23	4	9	9	34	18	30	11	4
10	29	10	19	10	35	24	5	21	5
11	4	16	29	11	36	30	11	0	6
12	10	22	8	12	37	5	17	10	7
13	16	28	18	13	38	11	23	20	8
14	22	3	28	14	39	17	29	30	9
15	28	9	7	15	40	23	4	9	10
16	3	15	17	16	41	29	10	19	11
17	9	21	27	17	42	4	16	29	12
18	15	27	6	18	43	10	22	8	13
19	21	2	16	19	44	16	28	18	14
20	27	8	26	20	45	22	3	28	15
21	2	14	5	21	46	28	9	7	16
22	8	20	15	22	47	3	15	17	17
23	14	26	25	23	48	9	21	27	18
24	20	1	4	24	49	15	27	6	19
25	26	7	14	25	50	21	2	16	20

2. Логическое проектирование сети. Используя варианты из таблицы, спроектируйте схему сети, состоящей из четырех подсетей (CIDR надо брать из вариантов), соединенных между собой несколькими маршрутизаторами. В каждой из подсетей разместите минимум 2-3 компьютера, придумайте и назначьте им IP-адреса и маски. IP-адреса не должны быть последовательными.

2.4. Последовательность выполнения работы

В данной работе задачи определения принадлежности IP-адресов к одной подсети и логической проектировки сети можно выполнять в любом порядке. При выполнении первой задачи последовательность вложенных действий a–e необходимо сохранить.

Для действия b необходимо воспользоваться процессом определения принадлежности двух IP-адресов к одной подсети. Для п. c необходимо настроить виртуальные машины (для этого можно использовать теоретические сведения из описания лаб. работы № 1.

Для задачи «Логическое проектирование сети» можно придумывать любые IP-адреса для компьютеров, так чтобы они соответствовали одной подсети (CIDR брать согласно вариантам). Оформить схему сети можно, используя, например, Microsoft Visio, draw.io; оформить на бумаге вручную либо воспользоваться другими технологиями.

2.5. Контрольные вопросы

1. Что такое IP-адрес? Для чего он нужен?
2. Что такое маска подсети? Для чего она нужна?
3. Что такое CIDR? Приведите пример.
4. Как определить широковещательный IP-адрес конкретной подсети?
5. Как определить, принадлежат ли два IP-адреса одной подсети?
6. Для чего нужен широковещательный IP-адрес?

3. ЛАБОРАТОРНАЯ РАБОТА № 3. ИЗУЧЕНИЕ МЕХАНИЗМОВ ТРАНСЛЯЦИИ СЕТЕВЫХ АДРЕСОВ: NAT, MASQUERADE

3.1. Цель и задачи

Целью работы является изучение механизмов преобразования сетевых адресов: NAT, Masquerade. Подробно рассмотрены некоторые сетевые возможности VirtualBox, который будет использован для создания необходимой инфраструктуры. Необходимо решить следующие задачи:

1. Создать три виртуальные машины (лаб. работа № 1).
2. Настроить имена, IP-адреса для каждой из подсетей в соответствии со схемой.
3. Настроить переадресацию пакетов между сетевыми интерфейсами для машины с NAT. Запретить прямой доступ между двумя частными подсетями (необходимо для воссоздания условий, приближенных к реальным).
4. Настроить Masquerade на NAT-машине и проверить доступ к сети Интернет с других машин и отсутствие доступа друг к другу.
5. Настроить доступ к сети Интернет для одной из машин с помощью sNAT.
6. Добавить вторичный IP-адрес на NAT-машину, по которому в дальнейшем будет отвечать на внешние запросы машина, указанная в п. 5.
7. Настроить dNAT для доступа к машине из внешней сети. Проверить настройки.

3.2. Основные теоретические сведения

Необходимость использования механизмов переадресации. На рис. 3.1 представлена общая схема сети, в которой используются механизмы переадресации.

Применительно к цели данной лабораторной работы основным элементом является частная сеть, включающая компьютер 1 и маршрутизатор 2, а вспомогательными – DNS-сервер google 8.8.8.8 (на рисунке отмечен номером 3) и частная сеть № 2 с компьютером 4.

Маршрутизатор 2 включает в себя частную подсеть № 1 и выход в сеть Интернет. Он (при правильной настройке) является своего рода «проводником» в Интернет для частной сети № 1. Данная лабораторная посвящена процессу настройки и работы данного маршрутизатора. Существует несколько вариантов подключения компьютера 1 к сети Интернет.

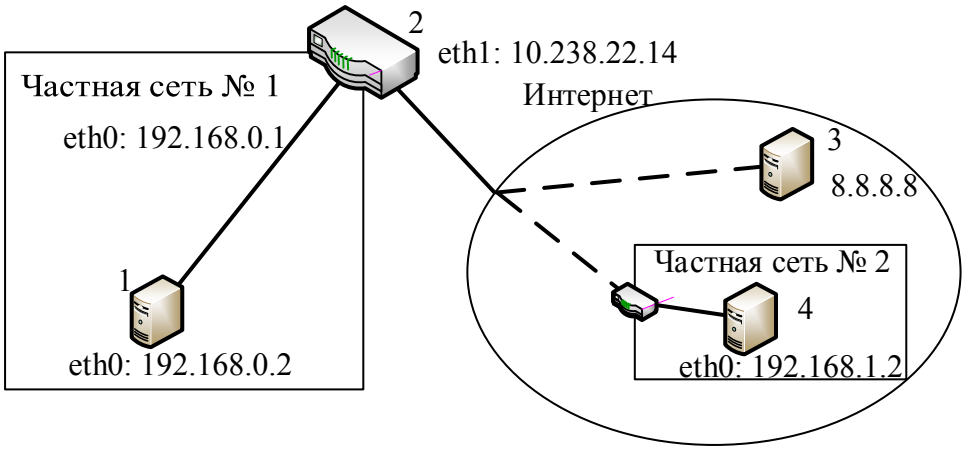

Рис. 3.1. Общая схема сети

Механизм Masquerade. Данный механизм настраивается на маршрутизаторе 2 и работает следующим образом: всем пакетам при прохождении через eth1 маршрутизатора 2 присваивается новый IP-адрес отправителя, который подключен к внешней сети (в примере это eth1: 10.238.22.14).

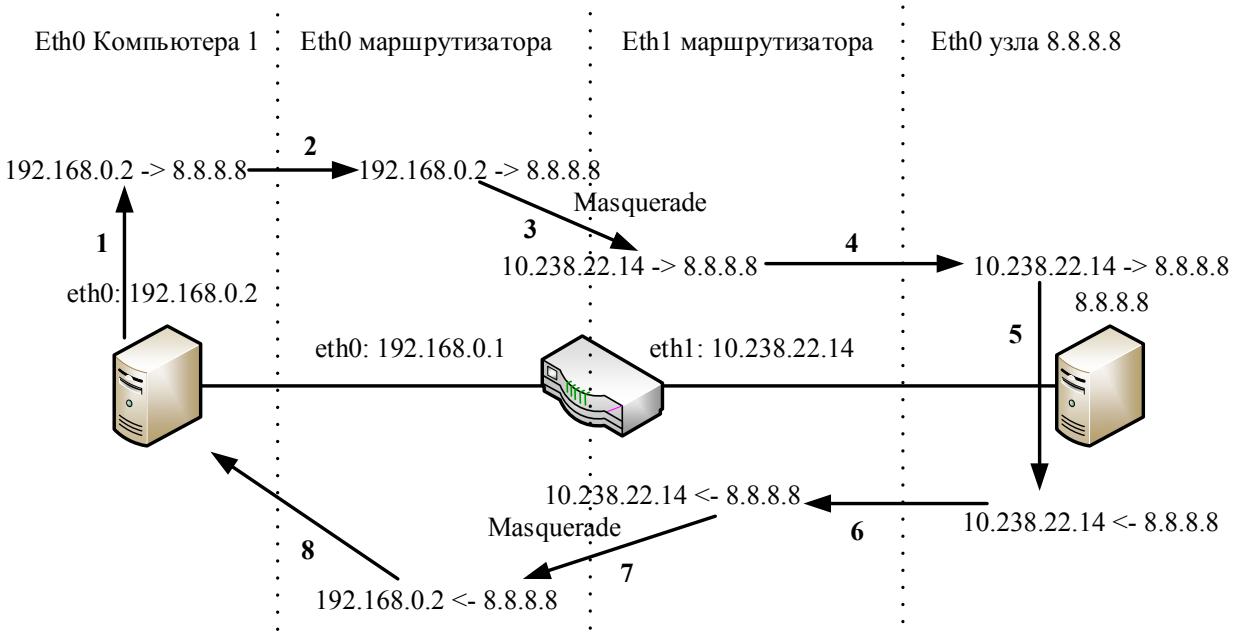

Рис. 3.2. Процесс обмена трафиком между внутренним и внешним узлами

Следует отметить, что частная сеть не видна из сети Интернет, как и сеть Интернет не видна из частной сети № 1. При использовании «маскарадинга» (masquerade) узлы частной сети могут общаться с внешней сетью (например, Интернет) благодаря подмене IP-адресов. В указанном примере IP-адрес 10.238.22.14 доступен в сети Интернет. Это позволяет запрашивать с него любую информацию, находящуюся в сети Интернет. Маскарадинг позволяет компьютерам частной сети передать запрос и получить ответ из сети Интернет. Рис. 3.2 отражает данный процесс при обращении ПК из сети № 1 к DNS-серверу 8.8.8.8.

Данный процесс состоит из восьми шагов. На шаге 1 создается пакет от ПК 192.168.0.2 к 8.8.8.8. Далее пакет присылается на интерфейс eth0 маршрутизатора (шаг 2). На маршрутизаторе настроен Masquerade, меняющий IP-адрес отправителя на свой внешний интерфейс (в примере 10.238.22.14) при прохождении пакета во внешнюю сеть (шаг 3). На шаге 4 измененный пакет доставляется на узел с IP-адресом 8.8.8.8. На шаге 5 пакет обрабатывается и узел отвечает, указывая адрес доставки 10.238.22.14. Данный адрес доступен в сети Интернет, поэтому пакет на шаге 6 доходит до маршрутизатора. На шаге 7 с помощью Masquerade пакет видоизменяется обратно, чтобы дойти до конечного узла, который выполнил запрос (192.168.0.2). На шаге 8 измененный пакет приходит к начальному узлу, который отправил запрос на шаге 1.

Далее представлен пример настройки Masquerade с помощью iptables. В качестве маршрутизатора может выступать любой узел, подключенный к частной сети № 1 и сети Интернет в ОС Ubuntu:

```
iptables -t nat -A POSTROUTING -o enp0s8 -j MASQUERADE
```

Данное правило меняет всем пакетам, проходящим через интерфейс enp0s8, IP-адрес источника на IP-адрес интерфейса enp0s8.

Механизм NAT. NAT похож на Masquerade. Разница в том, что IP-адрес отправителя подменяется не на IP-адрес устройства, через который он проходит, а на специальный IP-адрес, заданный в процессе настройки данного устройства.

В зависимости от необходимости используются различные типы NAT. С технической точки зрения они сводятся к настройке sNAT (source NAT – замена IP-адреса источника) и dNAT (destination NAT – замена IP-адреса назначения). sNAT позволяет внутренним узлам частной сети общаться с внешними сетями (например, сетью Интернет). dNAT позволяет из внешней сети успешно отправить данные на узел внутренней сети.

Следует отметить, что, подключая компьютер к сети Интернет через Wi-Fi или другой маршрутизатор, используют одну из перечисленных выше технологий для организации доступа в сеть Интернет с домашних компьютеров. При этом зачастую это уже преднастроено в маршрутизаторах, и ручная настройка не нужна, так как используется Masquerade. Если же требуется, чтобы компьютер имел конкретный IP-адрес, то необходимо дополнительно настроить маршрутизатор (sNAT, dNAT).

Рассмотрим настройку sNAT и dNAT. Для этого на узле, который имеет выход во внешнюю сеть, необходимо настроить вторичный IP-адрес. Он будет связующим звеном между частной и внешней сетями. Сначала настроим на узле маршрутизации sNAT:

```
iptables -t nat -A POSTROUTING -s 192.168.0.2/32 -o enp0s3 -j
sNAT--to-source 10.144.2.100
```

Данное правило означает, что в цепочке NAT после обработки пакета для всех пакетов, IP-адрес источника которых равен 192.168.0.2 (т. е. они были отправлены с этого узла), будет происходить его смена на 10.144.2.100 (IP-адрес, доступный во внешней сети). Благодаря этому правилу пакет, отправленный из частной сети, сможет дойти до необходимого узла во внешней сети и получить ответ.

Чтобы из внешней сети (например, сети Интернет) можно было получить доступ к узлу в частной сети, необходимо настроить dNAT:

```
iptables -t nat -A PREROUTING -d 10.144.2.100 -j DNAT --to-des-
tination 192.168.0.2
```

Данное правило означает, что если из внешней («публичной») сети пакет будет отправлен на 10.144.2.100, то при прохождении через узел, на котором это правило настроено, произойдет подмена IP-адреса назначения, и пакет дойдет до требуемого узла в частной сети с IP-адресом 192.168.0.2.

Типы трансляции сетевых адресов. Процесс сопоставления IP-адресов при трансляции обычно проходит по двум принципам: «один к одному»; «многие к одному».

При трансляции «один к одному» происходит сопоставление одного внутреннего IP к одному внешнему. Например, локальный адрес компьютера 1 на рис. 3.1 (192.168.0.2) будет сопоставляться с глобальным адресом NAT-маршрутизатора (10.238.22.14). Если в сети № 1 появится еще один компьютер (например, компьютер 5 с адресом 192.168.0.5), которому нужен доступ в Интернет, то на NAT-маршрутизаторе потребуется еще один глобальный адрес (например, 10.238.22.15).

Трансляция адресов «один к одному» может быть:

• статической, определяемой администратором;

• динамической, когда у NAT-маршрутизатора есть пул глобальных адресов, автоматически используемых для сопоставления локальным адресам по мере необходимости.

При трансляции «многие к одному» происходит сопоставление не только IP-адресов, но и портов, либо другой информации, когда протокол не использует портов для отправки сообщений.

Например, пусть компьютеры 1 (192.168.0.2) и 5 (192.168.0.5) обращаются к одному и тому же сервису в Интернете (8.8.8.8), а компьютер 1 – еще и к узлу с глобальным адресом 10.239.12.222 и портом 77. На NAT-маршрутизаторе может быть создана таблица.

Локальный адрес-порт	Глобальный адрес-порт NAT-маршрутизатора	Глобальный адрес-порт узла в Интернете
192.168.0.2:15003	10.238.22.14:1101	8.8.8.8:53
192.168.0.5:15003	10.238.22.14:1102	8.8.8.8:53
192.168.0.2:8877	10.238.22.14:1103	10.239.12.222:77

Как видно из таблицы, при таком принципе работы NAT сопоставляет и транслирует не только IP-адрес, но и порт. Это делается для того, чтобы избежать случаев, когда с двух сетевых устройств одновременно отправляются пакеты с использованием одного порта (см. первую и вторую строки таблицы).

Подобная реализация позволяет динамически обеспечить общение с нескольких IP-адресов, используя один внешний IP-адрес. Такой принцип использует Masquerade. Также с помощью принципа «многие к одному» можно настроить доступ извне локальной сети к двум ее сервисам по одному IP-адресу, настроив трансляцию по специально заданным вручную портам. Это позволит лучше использовать пространство белых IP-адресов и не покупать дополнительные.

Отслеживание смены IP-адресов в заголовках (Masquerade). Для наглядной демонстрации работы данных механизмов рассмотрим примеры настроек:

```
root@ub-nat:/home/user# tcpdump -p icmp –i eth0
17:09:10.828249 IP 192.168.0.2> 8.8.8.8: ICMP echo request, id
2407, seq 1, length 64
17:09:10.906169 IP 8.8.8.8 >192.168.0.2: ICMP echo reply, id
2407, seq 1, length 64
```

При просмотре трафика, проходящего через интерфейс eth0 маршрутизатора 2 (рис. 3.2), мы видим в заголовке пакета IP-адрес источника в частной сети. Тот же самый трафик, прошедший через интерфейс eth1 маршрутизатора 2, имеет следующий вид:

```
root@ub-nat:/home/user# tcpdump -p icmp -i enp0s3
17:09:10.828251 IP 10.238.22.14 > 8.8.8.8: ICMP echo request, id
2418, seq 1, length 64
17:13:20.906157 IP 8.8.8.8 > 10.238.22.14: ICMP echo reply, id
2418, seq 1, length 64
```

В примере в заголовке пакета изменен IP-адрес отправителя на IP-адрес внешнего интерфейса маршрутизатора 2. Узел 8.8.8.8, получив пакет, отправляет ответ на IP-адрес доступного узла (10.238.22.14), который далее транслирует его в частную сеть.

Перед началом выполнения лабораторной работы стоит отметить, что реализация технологии NAT на практике у вендоров сетевых устройств может быть различной. Нет одного конкретного алгоритма, который реализуется на

практике. В данной лабораторной работе представлены основные принципы технологии трансляции IP-адресов, однако на практике могут быть различные нюансы, например добавление ACL (контроль доступа) в процесс трансляции либо урезание некоторого функционала.

Построение инфраструктуры для выполнения работы. В условиях наличия только одного компьютера реализация данной задачи сводится, например, к построению подобной инфраструктуры в Oracle VirtualBox (рис. 3.3).

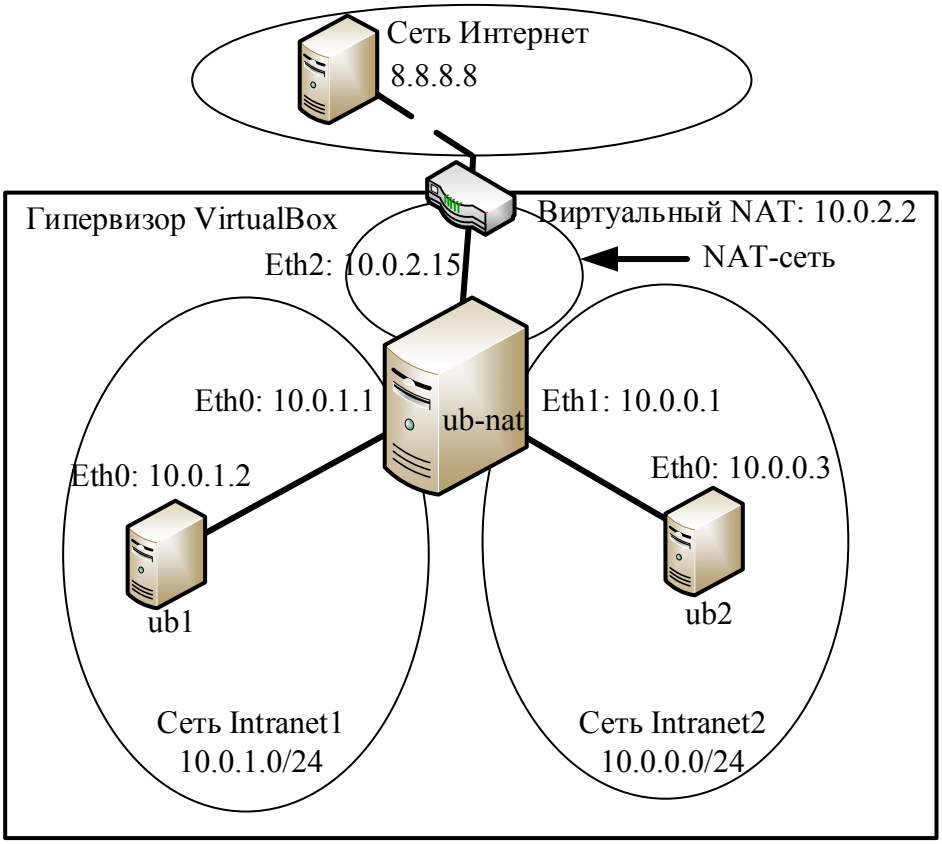

Рис. 3.3. Топология построения инфраструктуры в VirtualBox

Перед настройкой сетевых интерфейсов для ускорения загрузки виртуальных машин следует выполнить следующее:

```
sudo mkdir -p /etc/systemd/system/networking.service.d/
sudo bash -c 'echo -e "[Service]\nTimeoutStartSec=20sec" >
/etc/systemd/system/networking.service.d/timeout.conf'
sudo systemctl daemon-reload
```

Для создания данной топологии необходимо использовать следующие типы подключения интерфейсов в VirtualBox:

• Внутренняя сеть (Intranet1 и Intranet2). Внутренняя сеть, согласно руководству VirtualBox, является «программной сетью, которая может быть видима для выборочно установленных виртуальных машин, но не для приложений, работающих на хосте или на удаленных машинах, расположенных извне».

Такая сеть представляет собой набор из хоста и нескольких виртуальных машин. Но ни одно из вышеперечисленных устройств не имеет выхода через физический сетевой адаптер – он полностью программный, используемый VirtualBox в качестве сетевого маршрутизатора. В целом получается частная локальная сеть только для гостевых операционных систем без доступа в Интернет.

• Трансляция сетевых адресов (NAT). Протокол NAT позволяет гостевой операционной системе выходить в Интернет, используя при этом частный IP, который недоступен со стороны внешней сети или же для всех машин локальной физической сети. Такая сетевая настройка позволяет посещать web-страницы, скачивать файлы, просматривать электронную почту. И все это – используя гостевую операционную систему. Однако извне невозможно напрямую соединиться с такой системой, если она использует NAT. Можно провести аналогию с настройкой механизма sNAT, представленного ранее.

В качестве маршрутизатора будет выступать виртуальная машина «ub-nat», которая будет иметь выход в сеть Интернет посредством NAT-сети, а также подключена к двум внутренним сетям Intranet1 и Intranet2. Для обеспечения возможности переадресации трафика между интерфейсами внутри «ub-nat» необходимо включить данную опцию в sysctl. Для этого необходимо в файле /etc/sysctl.conf задать следующую переменную:

```
net.ipv4.ip_forward = 1
```

После этого следует перезагрузить «ub-nat», чтобы применить настройки. Однако отличие построенной инфраструктуры в VirtualBox (рис. 3.3) от общего примера (рис. 3.2) состоит в том, что внешняя частная сеть № 2 в VirtualBox является внутренней и «маршрутизатор ub-nat» напрямую подключен к данной сети – маршрут до частной сети № 2 ему «известен». В общем случае местоположение сети Intranet2 неизвестно. Для того, чтобы воссоздать подобные условия на одной из машин ub1 или ub2, необходимо закрыть прямой доступ в соседнюю внутреннюю сеть, например, для ub1:

```
root@ub1:/home/user# iptables –A OUTPUT -d 10.0.0.0/24 –j DROP
```

До применения этого правила можно проверить доступность с ub1 до ub2 с помощью команды ping:

```
root@ub1:/home/user# ping 10.0.0.3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=0.022 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.015 ms
```

После закрытия доступа к подсети 10.0.0.0/24 для ub1 команда ping успешно выполняться не будет:

```
root@ub1:/home/user# ping 10.0.0.3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
--- 10.0.0.3 ping statistics ---
6 packets transmitted, 0 received, 100% packet loss, time 5039ms
```

Далее по ходу выполнения работы понадобится выполнять команды iptables. Для того, чтобы после перезагрузки/выключения машин настройки iptables не стерлись, следует выполнить следующие действия:

1. Для сохранения правил, настроенных в текущей сессии, выполните команду:

```
# iptables-save > /root/firewall.rules
```

2. Для загрузки их после включения компьютера необходимо добавить следующую строку в файл /etc/network/interfaces:

```
pre-up iptables-restore < /root/firewall.rules
```

Это позволит применить правила после перезагрузки/выключения компьютера.

Эти действия помогут воссоздать общую картину взаимодействия узлов (ub1, ub2) с NAT-маршрутизатором (ub-nat).

Для выполнения лабораторной работы настройте виртуальные машины, как указано на рис. 3.3. Для усложнения задачи можно сменить IP-адреса узлов на любые, подходящие к условию.

Для удобства работы с виртуальными машинами рекомендуется добавить к каждой машине еще один интерфейс – виртуальный адаптер хоста (Host-only). При подключении типа «Виртуальный адаптер хоста» гостевые ОС могут взаимодействовать между собой, а также с хостом. Но все это только внутри самой виртуальной машины VirtualBox. В этом режиме адаптер хоста использует свое собственное, специально для этого предназначенное устройство, которое называется vboxnet0. Также VirtualBox создает подсеть и назначает IP-адреса сетевым картам гостевых операционных систем. Гостевые ОС не могут взаимодействовать с устройствами, находящимися во внешней сети, так как они не подключены к ней через физический интерфейс.

Данное подключение позволит поднять ssh-сессию с хостового ПК к каждому из узлов (для Windows можно использовать Putty), тем самым упростив доступ к управлению и настройке виртуальных машин.

3.3. Общая формулировка задач

Сначала необходимо изучить схему создания инфраструктуры для выполнения лабораторной работы. На рис. 3.2 отображена общая схема сети, в которой для доступа внутренних машин к сети необходима переадресация. Для настройки инфраструктуры на одном компьютере понадобится выполнить

дополнительные действия (рис. 3.3). Они описаны в подразделе «Построение инфраструктуры для выполнения работы».

Необходимо решить следующие задачи:

1. Создать и настроить инфраструктуру для выполнения лабораторной работы. Развернуть три виртуальные машины (лаб. работа № 1). Настроить их в соответствии с подразделом «Построение инфраструктуры для выполнения работы».

2. Настройка доступа с ub1, ub2 в сеть Интернет с использованием Masquerade. Настройте ub-nat, используя Masqurade, так, чтобы машины ub1 и ub2 имели доступ в сеть Интернет.

3. Настройка доступа с ub1, ub2 в сеть Интернет с использованием sNAT. Настройте ub-nat, используя sNAT, так, чтобы машины ub1 и ub2 имели доступ в сеть Интернет.

4. Настройка доступа с ub2 на ub1 с использованием dNAT. Настройте ub-nat, используя dNAT, так, чтобы с машины ub2 можно было получить доступ к ub1, используя IP-адрес из NAT-сети. Проверить успешность настроек можно, выполнив с узла ub2 команду: `ssh «SecondaryNatIPaddress»`.

В результате подключения будет отображено имя виртуальной машины ub1. Пример:

```
root@ub2:/home/user# ssh user@10.0.2.100
user@10.0.2.100'spassword:
user@ub1:~$
```

В данном примере вторичный IP-адрес на ub-nat настроен на интерфейсе, подключенном к NAT-сети, – IP-адрес: 10.0.2.100. При правильной настройке ssh доступ к этому IP-адресу будет открывать сессию с ub1.

3.4. Последовательность выполнения работы

Данную работу следует выполнять строго в последовательности, указанной в общей формулировке задач.

Для проверки доступности/недоступности узлов использовать команду ping:

```
ping 8.8.8.8 # для проверки выхода в сеть Интернет
ping 10.0.1.3 # для проверки доступности узлов, подставляя необ-
ходимый IP-адрес
```

Для отладки рекомендуется использовать утилиту tcpdump. Она выполняется от имени суперпользователя (sudo su). Данная команда отобразит все пакеты протокола ICMP, проходящие через все сетевые интерфейсы компьютера:

```
# tcpdump –p icmp –i any
```

Чтобы проверить наличие проходящих пакетов на конкретном интерфейсе, необходимо выполнить команду, например:

```
# tcpdump –p icmp –i eth0
```

Перед выполнением работ стоит отметить, что имена сетевых интерфейсов могут отличаться. Проверить это можно с помощью команды `ip a`

Вот пример результата вызова данной команды:

```
root@ub1:/home/user# ip a
1: lo: <LOOPBACK,UP,LOWER_UP>mtu 65536 qdiscnoqueue state
UNKNOWN group default qlen 1
    link/loopback 00:00:00:00:00:00brd 00:00:00:00:00:00
inet 127.0.0.1/8 scope host lo
valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP>mtu 1500 qdiscp-
fifo_fast state UP group default qlen 1000
    link/ether 08:00:27:f0:15:29 brdff:ff:ff:ff:ff:ff
inet 192.168.56.103/24 brd 192.168.56.255 scope global enp0s3
valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fef0:1529/64 scope link
valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP>mtu 1500 qdiscp-
fifo_fast state UP group default qlen 1000
    link/ether 08:00:27:bb:01:6d brdff:ff:ff:ff:ff:ff
inet 10.0.1.2/24 brd 10.0.1.255 scope global enp0s8
valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:febb:16d/64 scope link
valid_lft forever preferred_lft forever
```

В данном примере на узле ub1 настроены три интерфейса: `lo` – интерфейс-петля; `enp0s3` – интерфейс, подключенный к сети «Виртуальный адаптер хоста»; `enp0s8` – интерфейс во внутренней сети Intranet1.

При выполнении первой задачи (создать и настроить инфраструктуру для выполнения лабораторной работы) необходимо отразить в отчете результаты настройки: выполненные команды на узлах, недоступность ub2 с ub1. Наличие подключения к сети Интернет с виртуальной машины ub-nat и отсутствие – с ub1 и ub2. Также необходимо проверить доступ с ub1 и ub2 к интерфейсу ub-nat, подключенному к NAT-сети.

В результате выполнения второй задачи (настройка доступа с ub1, ub2 в сеть Интернет с использованием Masquerade) необходимо продемонстрировать доступ с ub1, ub2 в сеть Интернет. А также, что с ub1 нет доступа к ub2. Также нужно показать результаты выполнения команд для решения задачи.

Для того, чтобы сбросить настройки iptables, можно воспользоваться следующими командами:

```
# iptables -F; iptables -t nat -F; iptables -t mangle -F
```

Пользоваться этим можно каждый раз, если настройки были выполнены неверно. При выполнении задачи настройки доступа с ub1, ub2 в сеть Интернет с использованием sNAT результатом является отчет, аналогичный отчету в задаче с настройкой с использованием Masquerade.

В начале работы над задачей настройки доступа с ub2 на ub1 с использованием dNAT необходимо настроить «`Secondary Nat IP address`» на ub-nat. Пример настройки файла `/etc/network/interfaces` указан далее (подменить имя интерфейса и IP-адрес на соответствующие):

```
iface eth0 inet static
address 10.234.23.108
netmask 255.255.255.0
```

После редактирования файла необходимо перезагрузить виртуальную машину. Проверить успешность данной настройки можно таким образом:

```
root@ub2:/home/user# ssh user@10.234.23.108
user@10.234.23.108'spassword:
user@ub-nat:~$
```

Как видно, подключение по ssh открывает доступ к ub-nat. Об успешном выполнении данной задачи свидетельствует демонстрация ssh-подключения к вторичному IP-адресу, в результате которой будет открываться сессия с ub1:

```
user@10.234.23.108'spassword:
user@ub1:~$
```

3.5. Контрольные вопросы

1. Для чего используется трансляция сетевых адресов?
2. Чем Masquerade отличается от NAT?
3. Чем sNAT отличается от dNAT?
4. Что необходимо настроить, чтобы устройства из внешней сети имели доступ к узлу внутренней сети?
5. Как может выполняться проверка сетевых настроек в ОС Ubuntu?
6. Какой тип сети в VirtualBox не имеет выхода к внешним сетям?
7. Какой тип сети в VIrtualBox имеет выход только к одному узлу во внешней сети? Какой это узел?

4. ЛАБОРАТОРНАЯ РАБОТА № 4. СОЗДАНИЕ ВИРТУАЛЬНЫХ ЛОКАЛЬНЫХ СЕТЕЙ VLAN

4.1. Цель и задачи

Целью работы является изучение процессов создания и настройки виртуальных локальных сетей VLAN. Необходимо решить следующие задачи:

1. Создать три виртуальные машины (лаб. работа № 1).
2. Настроить VLAN между машинами.
3. Организовать две виртуальные сети между тремя машинами.
4. Обеспечить обмен данными между двумя разными виртуальными подсетями.

4.2. Основные теоретические сведения

Логическое разграничение компьютерной сети на виртуальные подсети. Локальные сети в наше время состоят из большого количества узлов: компьютеров, маршрутизаторов, коммутаторов, оргтехники. Чем больше количество узлов в одной подсети, тем большее количество сетевых процессов в них происходит, например, широковещательных запросов, передачи данных с узла на узел, отправки запросов на узлы-сервисы и т. д. Далее рассмотрен пример локальной сети предприятия (рис. 4.1).

Сеть предприятия является общей для каждого узла сети. Ethernet на рисунке – абстракция коммутаторов и каналов связи между узлами. При данном построении без дополнительных настроек каждый узел виден внутри сети. Руководство, бухгалтерия и остальные (разработчики, сервисы и т. д.) имеют доступ друг к другу. С технической точки зрения все в порядке, у каждого есть доступ к сети Интернет и к любым ресурсам компании. Однако с точки зрения организации рабочего процесса данная настройка не очень удачна, например любой узел имеет доступ к ресурсам бухгалтерии, финансовым документам, персональным данным.

Данная проблема может быть решена различными способами. Вот несколько стандартных вариантов решения:

1. Настроить правила доступа к каждому узлу на всех узлах сети.
2. Физически разделить общую сеть на несколько сетей, после чего настроить управляемый доступ с помощью маршрутизатора.
3. Настроить виртуальные сети (VLAN).

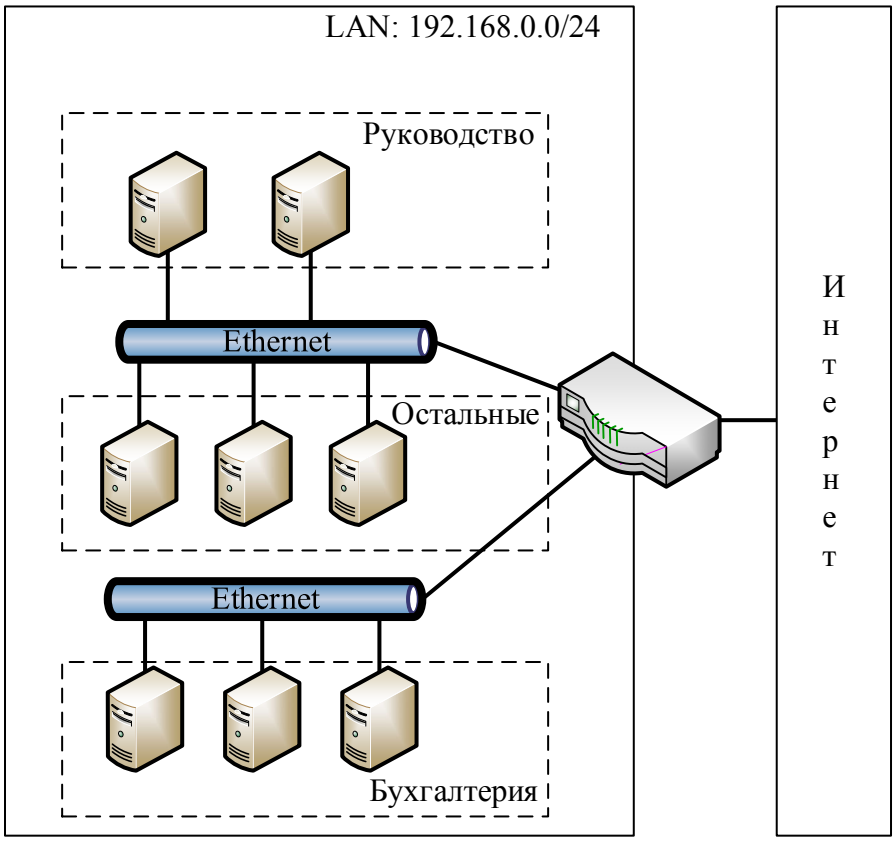

Рис. 4.1. Пример общей локальной сети предприятия

Первый вариант очень трудоемок – на каждом узле необходимо настроить права доступа (ACL – access control list). После первичной настройки обслуживание данной системы также затрудняется изменениями внутри сети: при добавлении нового узла доступ к нему необходимо настроить на каждом элементе сети; при изменении политики доступа к какому-то узлу необходимо перенастроить правила доступа на всех устройствах. Также сильно повышается риск человеческой ошибки при настройке подобного решения, поэтому в больших сетях (уже начиная с 10 компьютеров) он обычно не используется.

Второй и третий варианты подходят для решения задачи в общем виде, однако имеют свои достоинства и недостатки. Например, при физическом разделении сети необходимо проложить каналы связи, установить коммутаторы до каждого узла, который будет принадлежать конкретной подсети (рис. 4.2).

В случае с общей сетью все компьютеры подключены к любому коммутатору в любом порядке. Для того чтобы физически разделить логические группы компьютеров (А, Б, В), необходимо добавить в сеть большее количество коммутаторов (их число зависит от расположения конечных узлов каждой из групп), обеспечить физическое подключение от конечного узла к требуемой подсети. На рисунке представлен простейший пример из трех групп, компьютеры которых находятся недалеко друг от друга. Чем больше компьютеров

в сети и чем дальше они находятся друг от друга, тем большее количество сетевых элементов необходимо добавить к сети: установить коммутаторы и проложить новые линии связи и т. д. Подобное решение обычно используют, когда компания не слишком крупная либо необходимо обеспечить реальное физическое разделение между узлами разных групп.

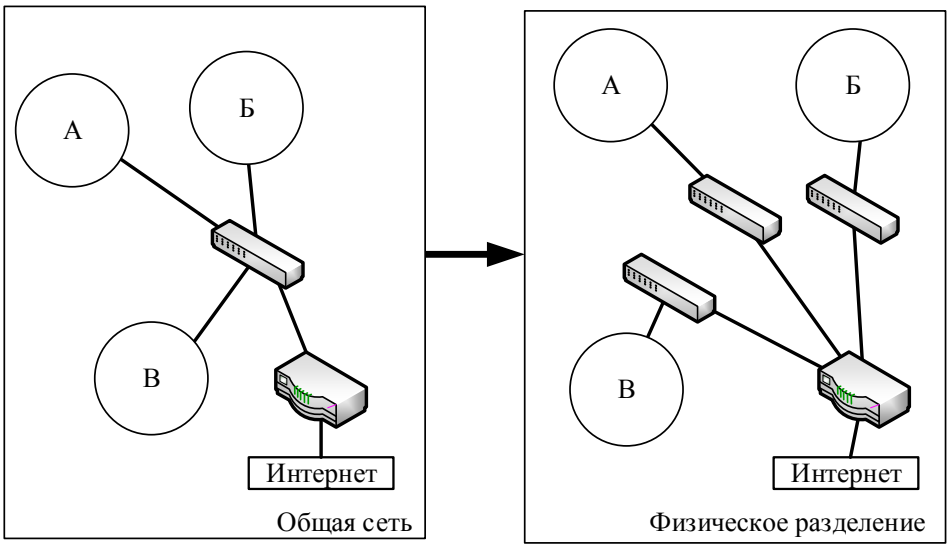

Рис. 4.2. Схема перехода от общей сети к физическому разделению на три подсети

Для снижения издержек можно использовать третий вариант – настроить виртуальные подсети в общей сети, что позволит логически разделить подсети. Если нет необходимости физического разделения подсетей, данный вариант позволит сэкономить на покупке нового оборудования и прокладки линий связи между узлами (рис. 4.3).

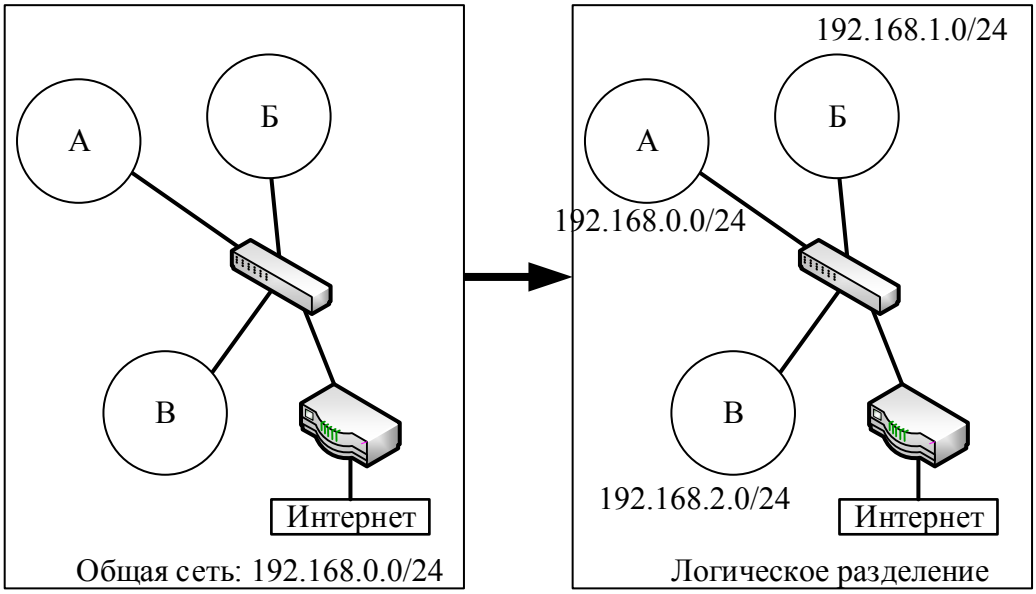

Рис. 4.3. Схема перехода от общей сети к виртуальным подсетям

Данный вариант включает только логическую настройку узлов. Таким образом можно обособить между собой подсети А, Б, В (руководство, бухгалтерия, остальные) без дополнительных финансовых затрат на закупку оборудования.

Виртуальной локальной сетью VLAN называется логическая группа узлов сети, трафик которой, в том числе и широковещательный, полностью изолирован от других узлов сети на канальном уровне. Это означает, что передача кадров между разными виртуальными сетями на основании MAC-адреса невозможна независимо от типа адреса – индивидуального, группового или широковещательного. В то же время внутри виртуальной сети кадры передаются по технологии коммутации, т. е. только на тот порт, который связан с MAC-адресом назначения кадра.

VLAN обладают следующими преимуществами:

1. Гибкость внедрения – VLAN являются эффективным способом группировки сетевых пользователей в виртуальные рабочие группы независимо от их физического размещения в сети.

2. Ограничивают распространение широковещательного трафика, что увеличивает полосу пропускания, доступную для пользователя.

3. Позволяют повысить безопасность сети, определив с помощью фильтров, настроенных на коммутаторе или маршрутизаторе, политику взаимодействия пользователей из разных виртуальных сетей.

Далее рассмотрена реализация данного решения. Она может происходить на уровне сетевых узлов (коммутаторов, маршрутизаторов) или на уровне каждого ПК. В обоих вариантах происходит следующее: пакеты дополнительно маркируются информацией о VLAN. Эта информация называется тегом и состоит из идентификатора протокола тега, приоритета, идентификатора канонического формата и самого идентификатора VLAN. Каждой такой виртуальной подсети выделяется номер, чтобы сетевой узел мог обеспечить обмен данными между узлами одного VLAN и не перенаправлял данные на узлы с другими номерами. Для новой подсети можно выбрать любое значение между 1 и 4094. VLAN ID 0 применяется в общей сети, которая не использует VLAN, однако в условиях стандартов, использующих VLAN (например, IEEE 802.1q, IEEE 802.1p), данным, исходящим из общей сети, назначается тег по умолчанию.

VLAN на основе сетевых узлов. При настройке VLAN на основе коммутаторов и маршрутизаторов ничего не настраивается на компьютерах и других

узлах, не отвечающих за сетевое окружение (оргтехника и т. д.). На коммутаторах создаются настройки для групп портов. Им назначаются различные VLAN ID в зависимости от того, компьютер какой логической сети к ним подключается. При проходе данных от компьютера к коммутатору пакеты тегируются (рис. 4.4).

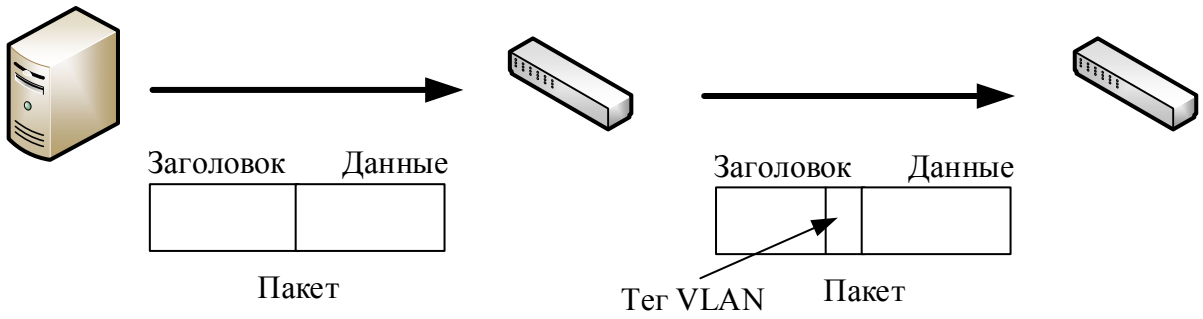

Рис. 4.4. Тегирование пакетов

Получая в результате пакет с тегом, следующие узлы разбирают его, анализируют и, если он соответствует настройкам на порте (VLAN ID), отбрасывают его либо отправляют нужному узлу далее. Благодаря этим настройкам работает логическое разграничение сети.

VLAN на основе сетевых карт узлов (ПК, оргтехника и др.). Настройку виртуальной подсети можно выполнить и на узлах компьютеров на сетевых картах. В этом случае на узле добавляется дополнительный виртуальный интерфейс, который подключается к физической сети через существующий сетевой интерфейс. Далее приведен пример настройки ОС Ubuntu.

Необходимо установить утилиту vlan:

```
sudo apt-get install vlan
```

В случае, когда физический сетевой интерфейс имеет имя eth0, в файле `/etc/network/interfaces` необходимо добавить следующие строки:

```
auto eth0.1001
iface eth0.1001 inet static
address 1.0.0.1
netmask 255.255.255.0
vlan_raw_device eth0
```

Для применения настроек необходимо перезагрузить ПК. Эти действия создадут виртуальный сетевой интерфейс с VLAN ID 1001 с IP-адресом 1.0.0.1, подключенный к сетевому интерфейсу eth0. После настройки данный ПК сможет общаться с другими компьютерами, у которых настроены такой же VLAN ID и IP-адрес из такой же подсети.

Данный вариант создания виртуальных подсетей является более сложным, так как необходимо выполнять настройки VLAN на каждом узле. Если коммутаторы не поддерживают стандарты, использующие VLAN, тогда настройку VLAN можно выполнить только за счет изменения конфигураций всех компьютеров сети.

4.3. Общая формулировка задач

Требуется создать три виртуальные машины Ub1, UbR, Ub3.

Необходимо решить следующие задачи:

1. Настроить VLAN между Ub1 и Ub3. VLAN ID, IP-адреса и маски подсети использовать согласно указанным ниже вариантам. Проверить выполнение ping между ПК, объяснить результат.

2. На машинах Ub1 и Ub3 запустить скрипты task2-v*.sh (предоставляет преподаватель), исправить ошибку в настройке сетевых адаптеров, после чего продемонстрировать успешный эхо-запрос от одного ПК к другому и обратно.

3. На трех ПК (Ub1, Ub3, UbR) запустить скрипт task3-v*.sh (предоставляет преподаватель), организовать подключение Ub1 к Ub3 и обратно через UbR, настроить UbR таким образом, чтобы эхо-запрос успешно проходил с Ub1 на Ub3.

4. На трех ПК запустить скрипт task4-v*.sh (предоставляет преподаватель). В данной задаче сеть настроена с ошибками. Необходимо исправить ошибку и показать выполнение эхо-запроса от Ub1 до Ub3.

4.4. Последовательность выполнения работы

Данную работу следует выполнять строго в последовательности, указанной в общей формулировке задач. Скрипты для задач необходимо получить у преподавателя в соответствии с вариантами. Варианты для первой задачи:

Вариант 1. Ub1: vlan id: 100, ip 10.0.0.1, netmask 255.255.255.0; Ub3: vlan id: 100, ip 10.0.0.2, netmask 255.255.255.0.

Вариант 2. Ub1: vlan id: 101, ip 10.168.16.1, netmask 255.255.240.0; Ub3: vlan id: 101, ip 10.168.30.220, netmask 255.255.240.0.

Вариант 3. Ub1: vlan id: 102, ip 1.7.0.2, netmask 255.192.0.0; Ub3: vlan id: 102, ip 1.60.60.60, netmask 255.192.0.0.

Вариант 4. Ub1: vlan id: 103, ip 220.23.12.7, netmask 255.255.248.0; Ub3: vlan id: 103, ip 220.23.8.34, netmask 255.255.248.0.

Вариант 5. Ub1: vlan id: 104, ip 8.0.0.7, netmask 255.255.224.0; Ub3: vlan id: 104, ip 8.0.31.222, netmask 255.255.224.0.

Вариант 6. Ub1: vlan id: 105, ip 110.10.12.54, netmask 255.128.0.0; Ub3: vlan id: 105, ip 110.1.13.67, netmask 255.128.0.0.

Вариант 7. Ub1: vlan id: 106, ip 18.18.18.35, netmask 255.255.255.224; Ub3: vlan id: 106, ip 18.18.18.60, netmask 255.255.255.224.

Вариант 8. Ub1: vlan id: 107, ip 78.98.178.198, netmask 255.255.255.224; Ub3: vlan id: 107, ip 78.98.179.47, netmask 255.255.254.0.

Вариант 9. Ub1: vlan id: 108, ip 77.97.99.10, netmask 255.255.255.248; Ub3: vlan id: 108, ip 77.97.99.12, netmask 255.255.255.248.

Вариант 10. Ub1: vlan id: 109, ip 77.97.99.10, netmask 255.255.255.248; Ub3: vlan id: 109, ip 77.97.99.12, netmask 255.255.255.248.

Вариант 11. Ub1: vlan id: 110, ip 154.137.12.8, netmask 255.255.255.224; Ub3: vlan id: 110, ip 154.137.12.27, netmask 255.255.255.224.

Вариант 12. Ub1: vlan id: 111, ip 255.255.192.0, netmask 255.255.192.0; Ub3: vlan id: 111, ip 250.250.190.12, netmask 255.255.192.0.

Вариант 13. Ub1: vlan id: 112, ip 12.13.14.15, netmask 255.255.255.128; Ub3: vlan id: 112, ip 12.13.14.120, netmask 255.255.255.128.

Вариант 14. Ub1: vlan id: 113, ip 222.12.45.76, netmask 255.255.248.0; Ub3: vlan id: 113, ip 222.12.43.12, netmask 255.255.248.0.

Схема подключения для задач 3 и 4 выглядит следующим образом:

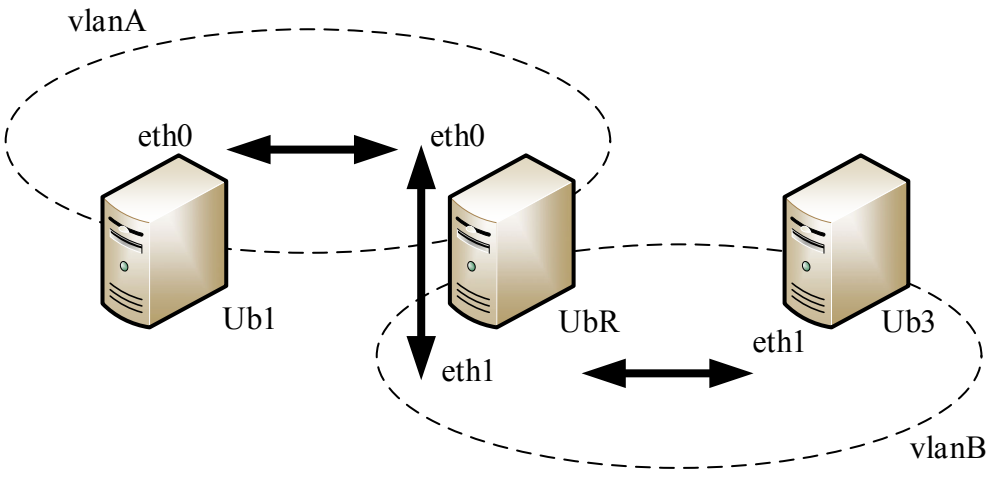

Рис. 4.5. Схема взаимодействия между узлами в третьей и четвертой задачах

В последних двух задачах (подраздел 4.3) настраивается маршрутизация пакетов с Ub1 на Ub3 и обратно. Для этого на Ub1 и Ub3 необходимо настроить маршрутизацию пакетов через UbR:

```
sudo route add default gw <ip-адрес UbR>
```

Для обеспечения возможности переадресации трафика между интерфейсами внутри UbR следует включить данную опцию в sysctl. Для этого в файле `/etc/sysctl.conf` задайте следующую переменную:

```
net.ipv4.ip_forward = 1
```

После задания переменной перезагрузите UbR. Затем на узле необходимо настроить два сетевых интерфейса, один из которых будет принадлежать одному VLAN, а второй – другому. Благодаря настройке переадресации трафика UbR будет выступать в качестве маршрутизатора между двумя виртуальными сетями.

В отчет необходимо включить настройки сетевых интерфейсов (имена, VLAN, IP-адреса, маски подсети, шлюз по умолчанию) – в том числе виртуальных – до исправлений и после.

4.5. Контрольные вопросы

1. Как настроить в Linux прием и передачу тегированного трафика?
2. Какие есть варианты разграничения компьютеров в сети?
3. Что такое VLAN? Для чего он нужен?
4. Как выполняется маршрутизация между разными VLAN?
5. На каких узлах может быть настроен VLAN?
6. Как настроить обмен данными между разными подсетями?

5. ЛАБОРАТОРНАЯ РАБОТА № 5. СЕТЕВЫЕ ЭКРАНЫ. IPTABLES

5.1. Цель и задачи

Целью работы является изучение принципов работы с сетевыми экранами. Необходимо решить следующие задачи:

1. Создать три виртуальные машины (лаб. работа № 1).

2. Научиться блокировать и разрешать прием и отправку пакетов с помощью iptables, настраивать логирование событий.

5.2. Основные теоретические сведения

Сетевой экран – это программный или аппаратный элемент компьютерной сети, осуществляющий контроль и фильтрацию проходящего через него сетевого трафика в соответствии с заданными правилами. Это необходимо для того, чтобы блокировать вредоносный трафик, направленный на различные сервисы, для обеспечения их бесперебойной работы и своевременного ответа на легитимные запросы.

Существует два основных подхода к настройке сетевых экранов:

1. Все, что не запрещено, разрешено.

2. Все, что явно не разрешено, запрещено.

В зависимости от требований безопасности и причин установки межсетевого экрана выбирается один из этих подходов. Например, если необходимо запретить доступ части узлов к другим, проще использовать первый подход (рис. 5.1). На маршрутизаторе можно запретить трафик, проходящий из подсети А в подсеть Б. После этих настроек узлы подсети А будут иметь доступ к ресурсам сети Интернет, но не смогут обмениваться данными с подсетью Б.

Если требуется максимально закрыть доступ к узлу извне, то используется второй подход. В его основе лежит запрет по умолчанию любого трафика с дальнейшим разрешением доступа с конкретных узлов, подсетей и др., а также разрешением доступа к ним.

На рис. 5.1 представлен пример, когда сетевой экран располагается на маршрутизаторе. Однако существуют и другие варианты расположения сетевых экранов.

Типы сетевых экранов по их местоположению. Существует два основных варианта расположения сетевого экрана: host-based и network-based.

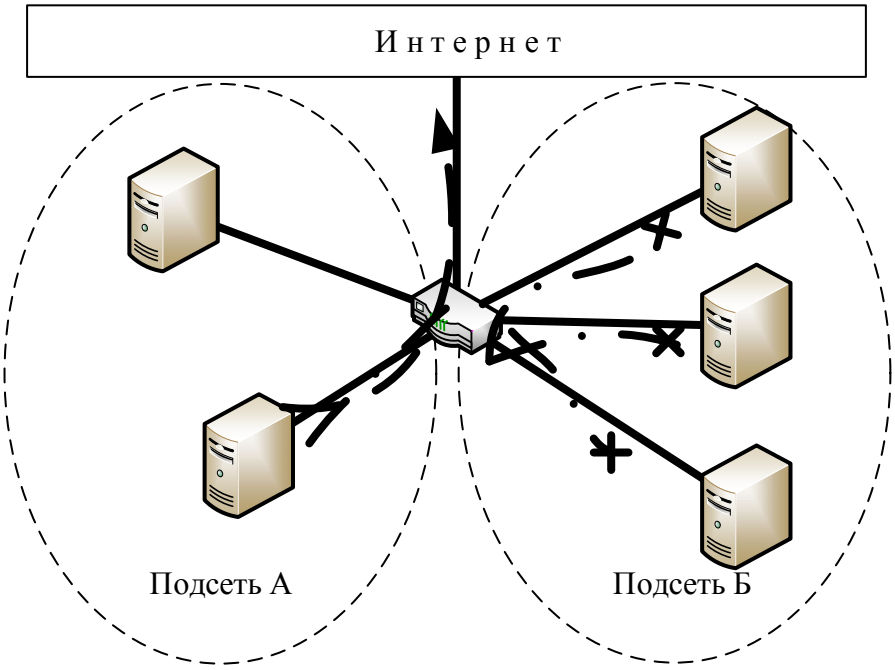

Рис. 5.1. Схема доступа к сети Интернет и подсети Б

Host-based сетевой экран – это экран, который располагается на самой системе, где необходимо фильтровать трафик. Network-based-экран устанавливается на промежуточных узлах между защищаемыми ресурсами и другими узлами.

Host-based сетевые экраны позволяют гибко и удобно настраивать правила на конкретном компьютере, упрощая тем самым процесс настройки и обслуживания системы фильтрации. Однако для защиты большого количества сетевых узлов их необходимо устанавливать на каждом узле, что может усложнить процесс администрирования системы. Кроме того, в корпоративной сети, обеспечивая защиту только на самих узлах, линии связи (коммутаторы и др.) остаются без защиты. Это может повлечь за собой отсутствие доступа к узлам при атаках на узлы связи.

Данная проблема может быть решена с помощью network-based сетевых экранов. Они устанавливаются на промежуточных узлах между защищаемыми узлами и внешним миром. Их можно настроить на фильтрацию всего трафика, проходящего через них во внутреннюю сеть и обратно, тем самым обеспечив централизованное администрирование сетевого экрана всех узлов внутренней сети. Однако следует отметить, что из-за дополнительных работ с проходящим трафиком длительность коммутации через подобные маршрутизаторы увеличивается, а нагрузка на них возрастает, при большом объеме циркулирующих данных это стоит учитывать. То же самое происходит и с защитой host-based – она использует ресурсы защищаемого узла. Внутренние узлы, по какой-то

причине начавшие генерировать вредоносный трафик на соседние узлы, в случае использования защиты network-based на концевом маршрутизаторе заблокированы не будут.

Есть еще один вариант использования сетевого экрана как промежуточной точки между внешней сетью и защищаемой – proxy-сервер. Трафик перед получением его защищаемым узлом перенаправляется на специальный узел. Этот узел является сетевым экраном и обеспечивает анализ данных и фильтрацию вредоносного трафика. Обычно подобное перенаправление устанавливается на концевом маршрутизаторе внутренней защищаемой сети, а сам сетевой экран может находиться внутри защищаемой сети и вне ее. Внешние узлы proxy-сервера обычно предоставляются сторонними компаниями, предлагающими «очистку данных». Такой подход позволяет снизить нагрузку на администрирование и настройку сетевого экрана, однако конфиденциальные данные при передаче их через внешний proxy-сервер могут быть перехвачены. Если подобную возможность нельзя допускать, можно установить внутренний proxy-сервер (рис. 5.2).

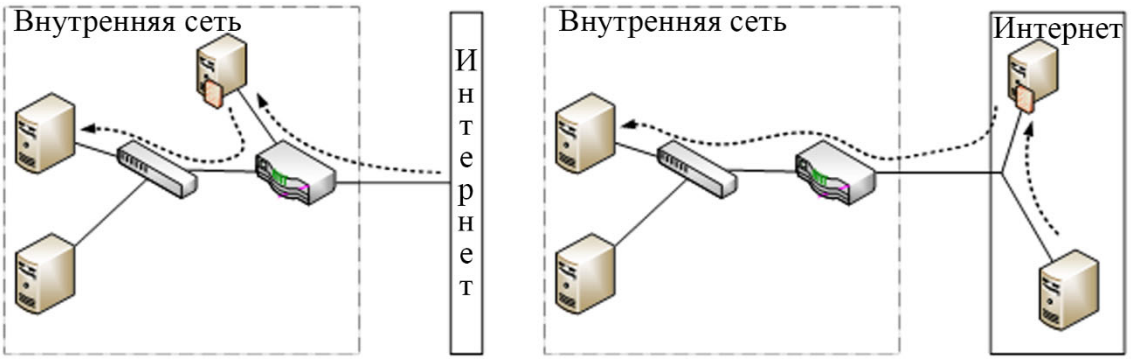

Рис. 5.2. Схемы расположения proxy-сервера

Типы сетевых экранов по глубине анализа трафика. Рассмотрим два наиболее часто используемых типа сетевых экранов.

Первый анализирует только заголовки пакетов и может работать на транспортном, сетевом и канальном уровнях иерархии DoD (TCP/IP). В данном варианте анализ входящих и исходящих пакетов осуществляется на основе информации, содержащейся в следующих полях TCP- и IP-заголовков пакетов: IP-адрес отправителя; IP-адрес получателя; порт отправителя; порт получателя. В зависимости от отслеживания активных соединений подобные сетевые экраны делятся на два типа: stateless (простая фильтрация) и stateful (фильтрация с учетом контекста). Экраны stateless фильтруют потоки данных на основе

статических данных о пакете, stateful включают в себя анализ логики работы протоколов и приложений.

К преимуществам фильтрации на основе анализа заголовков относятся невысокая стоимость подобных экранов, гибкость в задании правил фильтрации и небольшая задержка при прохождении трафика за счет простых правил анализа потоков данных. Однако данные экраны не могут анализировать фрагментированные пакеты и не отслеживают взаимосвязи между пакетами в силу отсутствия анализа данных в пакете, относящихся к уровню представления.

Второй тип межсетевого экрана может анализировать данные в пакете и работает на прикладном уровне иерархии DoD (TCP/IP). Благодаря работе на прикладном уровне можно организовать большое число проверок, которые будут использовать особенности работы протоколов прикладного уровня. Например, можно добавить проверку взлома известных «дыр» в программном обеспечении и протоколах. Но подобный функционал увеличивает время анализа данных, что сказывается на производительности. Обычно такие сетевые экраны являются host-based-экранами или proxy-серверами, так как работают под управлением более сложных ОС и требуют бóльших ресурсов по сравнению с экранами, работающими только с заголовками пакетов.

В данной лабораторной работе используется первый вариант сетевого экрана Linux – Netfilter. Для его настройки будет использован iptables.

Утилита управления сетевым экраном iptables. iptables – это утилита командной строки, управляющая межсетевым экраном. С ее помощью создаются правила, которые управляют фильтрацией и перенаправлением пакетов. Работа с iptables возможна только от имени суперпользователя. Есть несколько основных базовых понятий для работы с iptables: правило, таблица и цепочка.

Правило состоит из критерия, действия и счетчика. Если пакет соответствует критерию (требованию), то к нему принимается действие и этот пакет учитывается счетчиком. Критерий анализирует свойство пакета, например, условия совпадения источника назначения с заданным в условии. Действие определяет, что необходимо сделать с пакетом, если он попадает под критерий. В случае срабатывания правила счетчик учитывает этот пакет и также учитывает суммарный объем пакетов, подходящих под этот критерий, в байтах.

Таблица – совокупность базовых и пользовательских цепочек, объединенных общим функциональным назначением. Имена таблиц (как и модулей критериев) записываются в нижнем регистре, так как не могут конфликтовать с именами пользовательских цепочек.

Цепочкой является упорядоченная последовательность правил. Правила в цепочке срабатывают в порядке их указания, поэтому при их настройке важен порядок. Существует пять базовых цепочек:

1. PREROUTING – для изначальной обработки входящих пакетов.

2. INPUT – для входящих пакетов, адресованных непосредственно локальному процессу (клиенту или серверу).

3. FORWARD – для входящих пакетов, перенаправленных на выход (заметьте, что перенаправляемые пакеты проходят сначала цепь PREROUTING, затем FORWARD и POSTROUTING).

4. OUTPUT – для пакетов, генерируемых локальными процессами.

5. POSTROUTING – для окончательной обработки исходящих пакетов.

В случае необходимости можно создавать дополнительные пользовательские цепочки. Также существуют приложения, добавляющие свои цепочки в iptables, например, docker или Open vSwitch.

Цепочки организованы в четыре таблицы:

1. raw – просматривается до передачи пакета системе определения состояний. Используется редко, например, для маркировки пакетов, которые НЕ должны обрабатываться системой определения состояний. Для этого в правиле указывается действие NOTRACK. Содержит цепочки PREROUTING и OUTPUT.

2. mangle – содержит правила модификации (обычно заголовка) IP-пакетов. Среди прочего поддерживает действия TTL (Time to Live), TOS (Type of Service) и MARK (для изменения полей TTL и TOS и для изменения маркеров пакета). Содержит все пять стандартных цепочек.

3. nat – просматривает только пакеты, создающие новое соединение (согласно системе определения состояний). Поддерживает действия DNAT, SNAT, MASQUERADE, REDIRECT. Содержит цепочки PREROUTING, OUTPUT и POSTROUTING. Данная таблица используется в лаб. работе № 3.

4. filter – основная таблица, используется по умолчанию, если название таблицы не указано. Содержит цепочки INPUT, FORWARD и OUTPUT.

На рис. 5.3 представлена упрощенная диаграмма прохождения таблиц и цепочек.

Сначала пакет проходит через цепочку PREROUTING. В случае, если пакет должен быть передан другому узлу, далее используется цепочка FORWARD и после нее – POSTROUTING. В случае, когда пакет отправлялся

текущей системе, он попадает в цепочку INPUT. После этого он обрабатывается соответствующим приложением. В случае генерации ответа или запроса из самой системы к пакету сначала применяется цепочка OUTPUT, а после – POSTROUTING. В случае, если на каком-то из правил данный пакет отбрасывается, к нему применяются два варианта: DROP или REJECT. Действие DROP просто останавливает дальнейшую работу с пакетом. REJECT генерирует ответ отправителю о блокировке пакета.

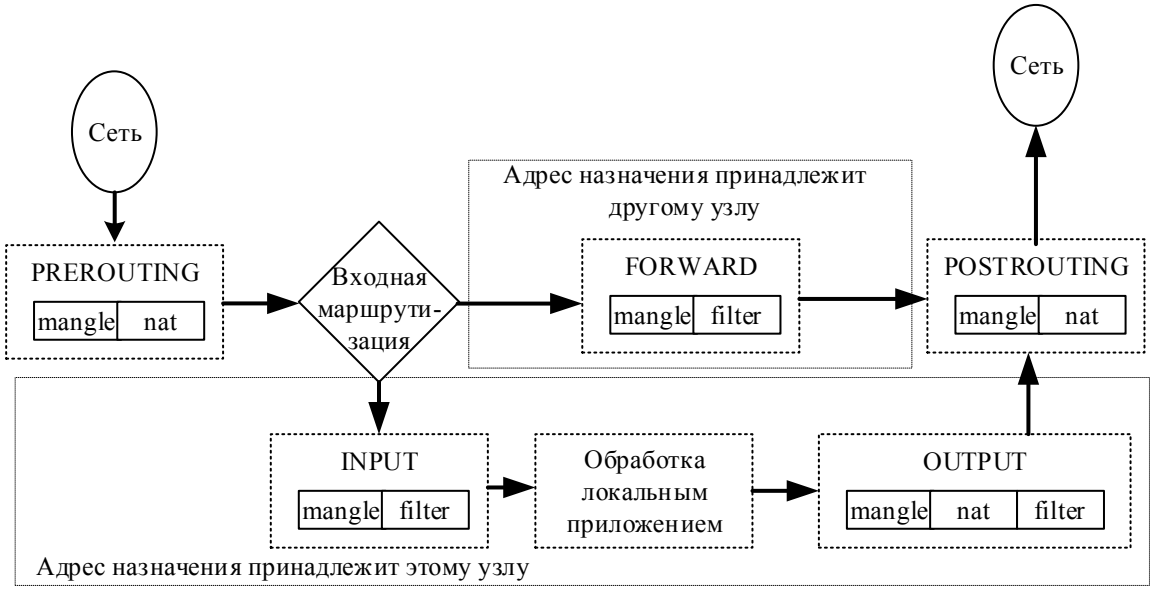

Рис. 5.3. Диаграмма процесса анализа пакетов в iptables

Далее представлены примеры использования iptables. Данная команда запрещает весь входящий трафик:

```
iptables -A INPUT -j DROP
```

Ключ –A означает, что правило добавляется в конец таблицы, в данном примере INPUT. Так как правила к пакетам применяются по порядку, то это необходимо учитывать. Чтобы вставить правило в конкретное место в цепочке, необходимо использовать ключ –I:

```
iptables -I INPUT 3 –s 192.168.0.32 -j DROP
```

Данная команда вставляет правило на 3-ю строчку, и оно блокирует все пакеты, у которых IP-адрес источника равен 192.168.0.32.

В следующем примере блокируются все пакеты, кроме ssh-соединений:

```
iptables -A INPUT -p tcp --dport 22 -j ACCEPT
iptables -A INPUT -j DROP
```

Так как ssh-сервер слушает 22-й порт, то первая команда разрешает для TCP-протокола (ключ –p) 22-й порт назначения, а второе правило блокирует

все пакеты. Важен порядок применения правил. Второе правило добавляется в конце таблицы, поэтому первое правило применяется сначала.

Вывести существующие правила можно следующей командой:

```
iptables –nvL
```

Примеры, касающиеся перенаправления трафика для NAT, рассмотрены в лаб. работе № 3. Если необходимо заблокировать исходящие пакеты, то используется таблица OUTPUT:

```
iptables -A OUTPUT -d 10.10.10.10 -j DROP
```

Это правило блокирует все пакеты, у которых IP-адрес назначения 10.10.10.10. Для отладки работы iptables можно настроить логирование:

```
iptables -A INPUT -I enp0s3 -s 1.0.0.0/24 -j LOG --log-prefix
"Logging info”
iptables -A INPUT -i enp0s3 -s 1.0.0.0/24 -j DROP
```

Первое правило сначала записывает событие, означающее, что на интерфейс enp0s3 пришел входящий пакет с IP-адресом источника 1.0.0.0/24. Второе правило блокирует его. Посмотреть логи можно в этом файле: `/var/log/messages`.

Также iptables может работать с MAC-адресами:

```
iptables -A INPUT -p tcp –d port 22 -m mac --mac-source
00:04:BA:67:04:37 -j ACCEPT
```

Данное правило разрешит прием пакетов для ssh-соединения от узла с указанным MAC-адресом.

Кроме добавления правил в цепочку их можно также удалять поштучно либо целиком очистить все правила. Если нужно удалить конкретное правило, это можно выполнить, указав его порядковый номер либо полностью указав спецификацию правила:

```
iptables –D INPUT 3
iptables –D INPUT –s 1.1.1.1 –j DROP
```

Первое правило удаляет из INPUT третье правило, а второе удаляет правило, которое отбрасывает пакеты, приходящие от IP-адреса 1.1.1.1.

Следующая команда удаляет все правила:

```
iptables –F
```

Чтобы удалить правила для конкретной цепочки для конкретной таблицы, необходимо дополнительно указать нужную информацию:

```
iptables –t nat –F PREROUTING
```

Данный пример стирает все правила из цепочки PREROUTING для таблицы nat.

5.3. Общая формулировка задач

Для выполнения лабораторной работы необходимо настроить три виртуальные машины Ub1, Ub2 и Ub3 так, чтобы они находились в одной подсети. Кроме того, для некоторых пунктов необходимо установить дополнительные службы на виртуальные машины: apache2, ftpd – и выполнить следующие задачи:

1. Заблокировать доступ по IP-адресу ПК Ub1 к Ub3. Продемонстрировать результаты с попыткой подключения Ub1 и Ub2 к Ub3.

2. Заблокировать доступ по 21-му порту на Ub1. Продемонстрировать возможность доступа по ssh на Ub1 и невозможность доступа по 21-му порту.

3. Разрешить доступ только по ssh на Ub2. Предоставить результат.

4. Запретить ICMP-запросы на IP-адрес 8.8.8.8 двумя способами. Необходимо создать два правила: в цепочке INPUT и цепочке OUTPUT. С помощью Wireshark на хосте нужно продемонстрировать разницу между двумя способами блокировки и сделать вывод о том, какой вариант эффективнее.

5. Полностью запретить доступ к Ub3. Разрешить доступ по ICMP-протоколу.

6. Запретить подключение к Ub1 по порту 80. Настроить логирование попыток подключения по 80-му порту. Продемонстрировать результаты логирования.

7. Заблокировать доступ по 80-му порту к Ub3 с Ub1 по его MAC-адресу. Продемонстрировать результат, сменить MAC-адрес на Ub3 и продемонстрировать успешное подключение к Ub3 по 80-му порту.

8. Полностью закрыть доступ к Ub1. Разрешить доступ для Ub3 к Ub1, используя диапазон портов 20–79. В результате необходимо показать невозможность подключения к 80 порту и возможность – к ssh или ftp.

9. Разрешить только одно ssh-подключение к Ub3. Продемонстрировать результат попытки подключения с Ub2 при наличии открытой ssh-сессии с Ub1 к Ub3.

5.4. Последовательность выполнения работы

Перед выполнением задач необходимо установить следующее ПО:

```
apt-get update && apt install apache2 ftpd –y
```

Для того, чтобы протестировать успешность выполнения различных заданий, потребуется использовать следующие приемы: для проверки, связанной с ICMP-протоколом ping, для проверок, связанных с TCP-протоколом, можно использовать команду telnet:

```
telnet 192.168.0.2 80
```

Успешное выполнение данной команды будет выглядеть таким образом:

```
Trying 192.168.0.2...
Connected to localhost.
Escape character is '^]'.
```

При работе с задачей «Запретить ICMP-запросы на IP-адрес 8.8.8.8 двумя способами» на хосте установите Wireshark. Запустите его на нужном интерфейсе для того, чтобы он отобразил необходимый трафик либо его отсутствие. Для фильтрации ненужного трафика вы можете добавить ключ «icmp», чтобы отображались пакеты, которые принадлежат только ICMP-протоколу.

После выполнения каждой задачи необходимо удалять все правила из цепочки. Если так случилось, что были добавлены неверные правила и доступ к виртуальной машине потерян, можно перезагрузить ее. После этого правила сотрутся и доступ будет восстановлен.

5.5. Контрольные вопросы

1. Что позволяет делать сетевой экран?
2. Какие бывают типы сетевых экранов и чем они различаются?
3. Приведите примеры, когда лучше использовать сетевые экраны host-based.
4. Каким типом сетевого экрана является iptables?
5. Каковы минусы использования proxy-сервера в качестве сетевого экрана?
6. Для чего нужна таблица NAT в iptables?
7. Чем DROP отличается от REJECT?
8. Какую цепочку лучше использовать, чтобы заблокировать доступ с ПК на ресурс во внешней сети?
9. Можно ли использовать несколько типов сетевых экранов для защиты корпоративных сетей и узлов и почему/для чего?

ЗАКЛЮЧЕНИЕ

В результате выполнения всех лабораторных работ студент должен научиться выполнять сетевые настройки на узлах сети, понимать принципы переадресации пакетов, создания виртуальных машин, отладки и проверки различных сетевых настроек, понимать процесс создания и обслуживания виртуальных сетей, а также контроля проходящего через узлы трафика.

После успешного выполнения всех лабораторных работ необходимо предоставить преподавателю отчеты по всем лабораторным работам, а в ходе защиты работ уметь выполнять различные настройки, относящиеся к теме текущей и ранее выполненных лабораторных работ.

Список рекомендуемой литературы

Олифер В., Олифер Н. Компьютерные сети. Принципы, технологии, протоколы. СПб.: Питер, 2016.

Таненбаум Э., Уэзеролл Д. Компьютерные сети. 5-е изд. СПб.: Питер, 2014.

СОДЕРЖАНИЕ

Борисенко Константин Алексеевич,
Беляев Сергей Алексеевич

Сети и телекоммуникации

Учебно-методическое пособие

Редактор Н. В. Кузнецова

Подписано в печать 31.03.21. Формат 60×84 1/16.
Бумага офсетная. Печать цифровая. Печ. л. 2,75.
Гарнитура «Times New Roman». Тираж 82 экз. Заказ .

Издательство СПбГЭТУ «ЛЭТИ»
197376, С.-Петербург, ул. Проф. Попова, 5