

Факультет программной инженерии и компьютерной техники

Компьютерные сети

Лабораторная работа №4

Вариант №1

Преподаватель: Маркина Татьяна Анатольевна

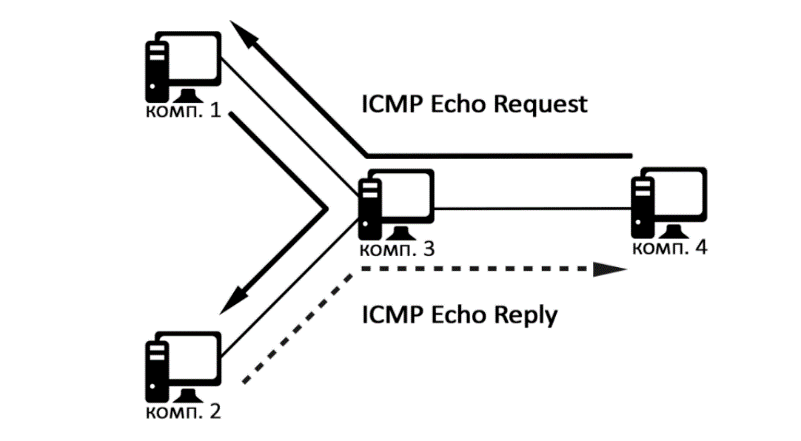
Выполнил: Кульбако Артемий Юрьевич Р33112

# Цель работы

**Цель работы – изучение основных методов настройки маршрутизируемых компьютерных сетей на примере сети, состоящей из компьютеров под управлением ОС Linux.**

В процессе выполнения работы изучается сетевой уровень модели OSI. Производится базовая настройка связности в сети, управление таблицами маршрутизации и правилами трансляции сетевых адресов. При помощи утилиты tcpdump выполняются наблюдения за передачей трафика по каналам связи в маршрутизируемой компьютерной сети. Применение утилиты tcpdump позволяет непосредственно в терминале (это основной метод управления сетевым оборудованием) наблюдать проходящие через интерфейсы компьютера пакеты и изучить их внутреннюю структуру. В данной работе изучаются методы маршрутизации в сетях IPv4 и IPv6, а также широко распространенная в компьютерных сетях технология NAT.

# Задание

На рисунке 4.5 изображена топология сети и требуемый путь прохождения сетевых пакетов. Необходимо так настроить хосты сети, чтобы ping-запрос (request) от компьютера 4 к компьютеру 2 шёл по пути, указанному сплошной стрелкой. При этом ping-ответ (reply) должен возвращаться другим путём: по пунктирной стрелке.

# Выполнение

Определим адреса хостов:

|  |  |  |
| --- | --- | --- |
| № | IPv4 | IPv6 |
| 1 | 7.8.1.2 | ::ffff:0708:0102 |
| 2 | 7.8.2.2 | ::ffff:0708:0202 |
| 3 | 7.8.1.1 | ::ffff:0708:0101 |
| 4 | 7.8.3.2 | ::ffff:0708:0302 |

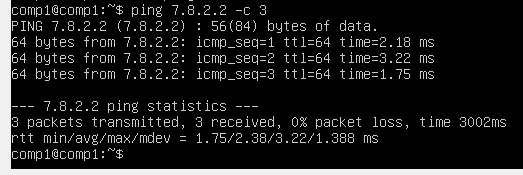
Все скрипты выполняются в sudo режиме. Виртуальный жёсткий диск машины можно просто продублировать, чтобы не повторять процесс установки 4 раза (именно поэтому на скриншотах имя пользователя и компьютера везде одинаковое).

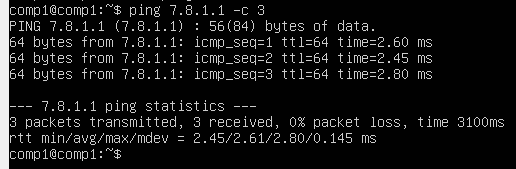
1. Формирование топологии, проверка утилитой ping.

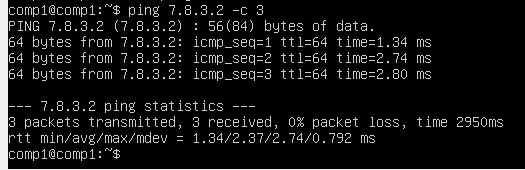
Для настройки интерфейсов будем использовать скрипт form\_topology.sh. Запускать в каждой виртуальной машине, передав в качестве аргумента её номер и режим адресации: IPV4 или IPV6.

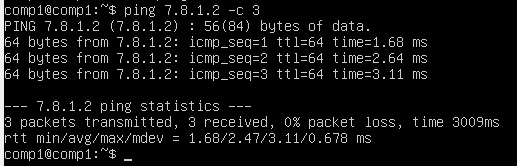
*function* configure\_eth() {  
 *echo* "Настройка интерфейсов"  
 *ip* "$net\_param""$mask" a add "$1" dev enp0s3  
}  
  
*function* add\_route\_to() {  
 *echo* "Добавление маршрута"  
 *ip* "$net\_param" ro add default via "$1"  
}  
  
*function* ping\_to() {  
 *echo* "Проверка утилитой ping"  
 *for* i *in* "*$*{1[@]}"  
 *do  
 ping* "*$*{hosts[$i]}" -c 2  
 *done*}  
  
*if* [ $# -lt 2 ]  
 *then  
 echo* "Необходимо задать номер хоста и режим адресации"  
 *exit  
fi  
# i-элемент == номеру хоста, а 0-вой оставим пустым для удобства обращения  
case $*{2} *in* "IPV4")  
 hosts=("" "7.8.1.2" "7.8.2.2" "7.8.1.1" "7.8.3.2")  
 net\_param=""  
 mask="/30"  
 *;;* "IPV6")  
 hosts=("" "::ffff:0708:0102" "::ffff:0708:0202" "::ffff:0708:0101" "::ffff:0708:0302")  
 net\_param="-6"  
 mask="/126"  
 *;;* \*)  
 *echo* "Не выбран режим адресации"  
 *exit  
 ;;  
esac  
sysctl* -w net.ipv4.ip\_forwarding = 1

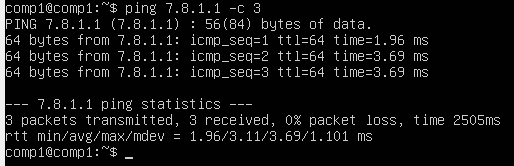
*sysctl* -w net.ipv4.conf.all.rp\_filter = 1  
*ip* link set enp0s3 up  
*ip* a flush enp0s3  
*case $*{1} *in* "1")  
 *configure\_eth* "*$*{hosts[1]}"  
 *add\_route\_to* "*$*{hosts[3]}"  
 dest=(2 3 4)  
 *ping\_to* "*$*{dest[@]}"  
 *;;* "2")  
 *configure\_eth* "*$*{hosts[2]}"  
 *add\_route\_to* "*$*{hosts[3]}"  
 dest=(1 3 4)  
 *ping\_to* "*$*{dest[@]}"  
 *;;* "3")  
 *configure\_eth* "*$*{hosts[3]}"  
 *add\_route\_to* "*$*{hosts[4]}"  
 dest=(1 2 4)  
 *ping\_to* "*$*{dest[@]}"  
 *;;* "4")  
 *configure\_eth* "*$*{hosts[4]}"  
 dest=(1 2 3)  
 *ping\_to* "*$*{dest[@]}"  
 *;;* \*)  
 *echo* "Некорректно задан номер хоста"  
 *;;  
esac*

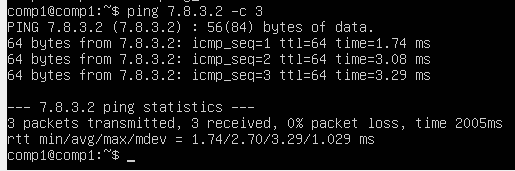


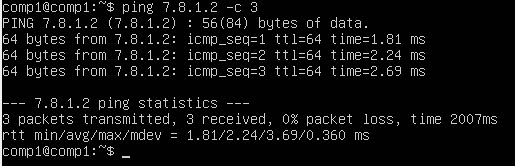


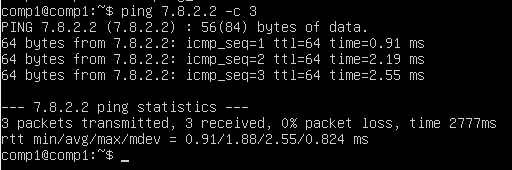


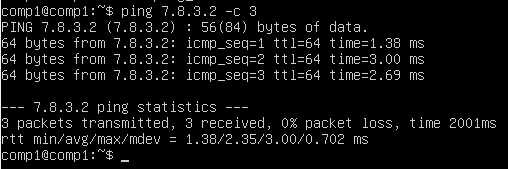


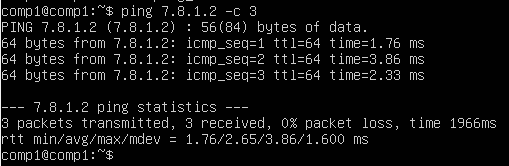


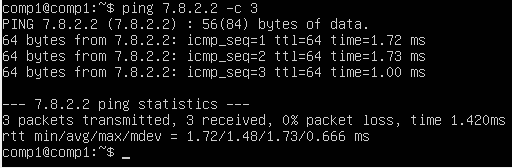


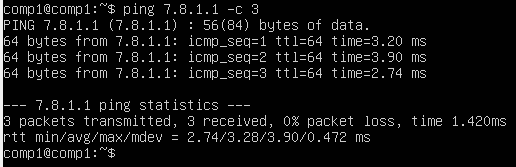












## Отправка пакетов утилитой nc.

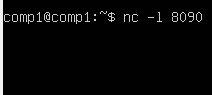
В качестве аргумента передаётся режим работы nc. Скрипт отправляет сообщение автоматически.

modes=("CLIENT" "SERVER")  
*case $*{1} *in* "*$*{modes[0]}")  
 *nc* 7.8.3.2 8090 << **EOF**Artemy Kulbako  
**EOF** *;;* "*$*{modes[1]}")  
 *nc* -l 8090  
 *;;* \*)  
 *echo* "Режим задан некорректно"  
*esac*

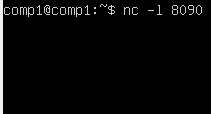
|  |  |
| --- | --- |
|  |  |

## Создание межсетевого экрана

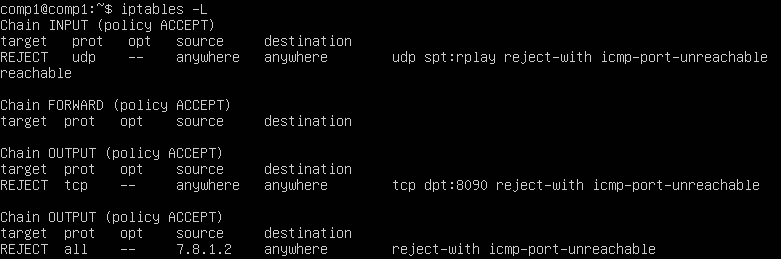
*iptables* -F  
*echo* "Запретить передачу только тех пакетов, которые отправлены на TCP-порт, заданный в настройках утилиты nc."  
*iptables* -A OUTPUT -j REJECT -o enp0s3 -p tcp --dport 8090



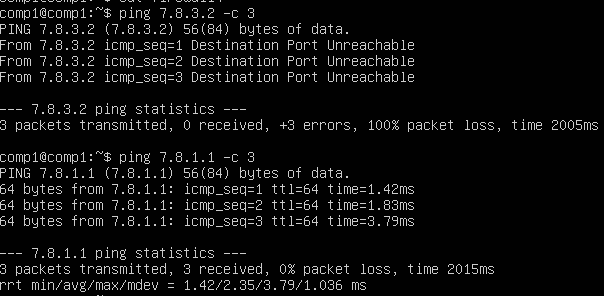
*echo* "Запретить приём только тех пакетов, которые отправлены с UDP-порта утилиты nc."  
*iptables* -A INPUT -j REJECT -i enp0s3 -p udp -sport 8090



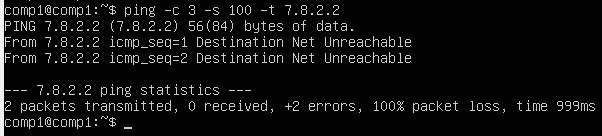
*echo* "Запретить передачу только тех пакетов, которые отправлены с IP-адреса компьютера А."  
*iptables* -A OUTPUT -j REJECT -o enp0s3 -s 7.8.1.2



*echo* "Запретить приём только тех пакетов, которые отправлены на IPадрес компьютера Б."  
*iptables* -A INPUT -j REJECT -i enp0s3 -d 7.8.3.2/30



*echo* "Запретить приём и передачу ICMP-пакетов, размер которых превышает 1000 байт, а поле TTL при этом меньше 10."  
*iptables* -A OUTPUT -o enp0s3 -p icmp -m length *!* --length 0:1000 -m ttl --ttl-lt 10 -j REJECT

****

## Задание по варианту 1 (IPv4)

Для настройки перенаправления используем скрипт forwarding.sh, который в качестве параметра принимает номер машины и тип адресации. Сложность задачи, что пакет от хоста 4, после попадания на хост 3, должен не сразу отправиться на хост2, а сначала обойти хост1. Для достижения этого, пакет будет передаваться через разные порты, чтобы мы могли определить, откуда он пришёл, и, следовательно, куда должен отправиться дальше.

Итоговая схема будет выглядеть так:

**С 4 на 2:**

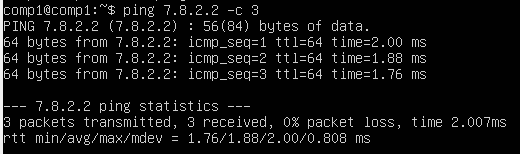
1. Пакеты, отправленные с хоста4, или полученные извне, перенаправляются на хост3 на порт 55004.
2. С хоста3 с порта 55004 на хост1.
3. С хоста 1 на хост3 порт 55001.
4. С хоста3 порт 55001 на хост2.

**C 2 на 4:**

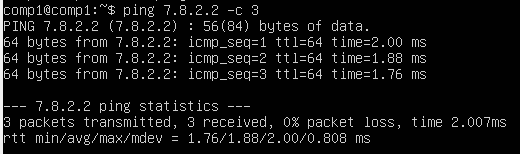
1. C хоста2 на хост3 порт 5002.
2. С хоста3 порт 5002 на хост4.

*function* forward\_icmp\_packets() {  
 *# 1-ый параметр: IP, куда направляется пакет  
 # 2-ой параметр: IP, куда перенаправить пакет  
 # 3-ой параметр: порт, куда перенаправить  
 # 4-ой параметр: порт, откуда перенаправить  
 case* "$#" *in* "2")  
 *echo* "Настройка переадресации"  
 *iptables* -t nat -A PREROUTING -i enp0s3 -p icmp -d "$1" -j DNAT --to-destination "$2"  
 *;;* "3")  
 *echo* "Настройка переадресации с заданием порта на который отправлять пакет"  
 *iptables* -t nat -A PREROUTING -i enp0s3 -p icmp -d "$1" -j DNAT --to-destination "$2":"$3"  
 *;;* "4")  
 *echo* "Настройка переадресации с заданием порта на который отправлять пакет и откуда он пришёл"  
 *iptables* -t nat -A PREROUTING -i enp0s3 -p icmp -d "$1" --dport="$4" -j DNAT --to-destination "$2":"$3"  
 *;;* \*)  
 *echo* "Неверно задано количество аргументов"  
 *exit  
 ;;  
 esac  
 iptables* -t filter -A FORWARD -p icmp -d "$2" -j ACCEPT  
}  
  
*case $*{2} *in* "IPV4")  
 hosts=("" "7.8.1.2" "7.8.2.2" "7.8.1.1" "7.8.3.2")  
 *;;* "IPV6")  
 hosts=("" "::ffff:0708:0102" "::ffff:0708:0202" "::ffff:0708:0101" "::ffff:0708:0302")  
 *;;* \*)  
 *echo* "Не выбран режим адресации"  
 *exit  
 ;;  
esac  
case $*{1} *in* "1")  
 *iptables* -t nat -F  
 *forward\_icmp\_packets\_from\_to* "*$*{hosts[1]}" "*$*{hosts[3]}" 55001  
 *;;* "2")  
 *iptables* -t nat -F  
 *forward\_icmp\_packets\_from\_to* "*$*{hosts[4]}" "*$*{hosts[3]}" 55002  
 *;;* "3")  
 *iptables* -t nat -F  
 *forward\_icmp\_packets\_from\_to* "*$*{hosts[3]}" "*$*{hosts[1]}" 55001 55004  
 *forward\_icmp\_packets\_from\_to* "*$*{hosts[3]}" "*$*{hosts[2]}" 55002 55001  
 *forward\_icmp\_packets\_from\_to* "*$*{hosts[3]}" "*$*{hosts[4]}" 55004 55002  
 *;;* "4")  
 *iptables* -t nat -F  
 *forward\_icmp\_packets\_from\_to* "*$*{hosts[2]}" "*$*{hosts[3]}" 55004  
 *;;* \*)  
 *echo* "Узел задан некорректно"  
 *;;  
esac*

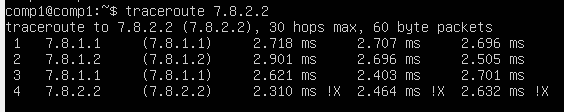
Запрос доходит до цели:



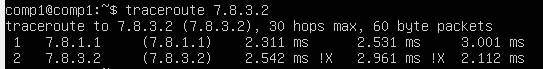
И обратно:



Трассировка пути туда:



И обратно:



## Задание по варианту 1 (IPv6)

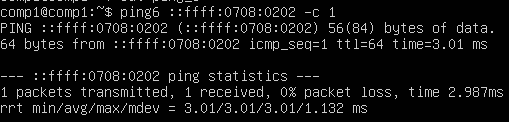
Для создания топологии запустить на каждом хосте скрипт из пункта 1, передав в качестве 2-го аргумента IPV6. Пример:

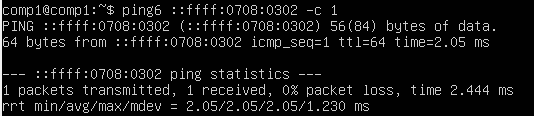
*./form\_topology* 1 IPV6

Для настройки маршрутизации, нужно запустить скрипт из пункта 4 с 2-ом параметром IPV6 для каждого хоста, т.к. выполняемые команды отличаются исключительно форматом адреса. Пример:

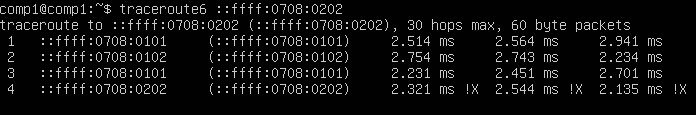
*./forwarding.sh* 1 IPV6

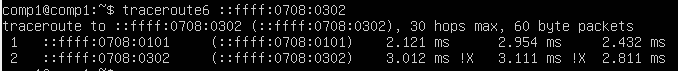
Проверки:





Трассировки:





# Выводы

В процессе выполнения были получены навыки управления вычислительной сетью из 4 хостов такие как: перенаправление трафика через нужные узлы, создание простейших правил фильтрации, настройка маршрутизации в локальных сетях. Основная сложность работы заключается в том, что предоставленные утилиты многофункциональны и сложны, разобраться в них непросто. К тому же, сложно было догадаться, как при маршрутизации настроить правила хождения трафика, когда у нас есть «узел-перекрёсток», было решено использовать правило пересылки через порт.

# Список использованной литературы

1. Т. И. Алиев, В. В. Соснин, Д. Н. Шинкарук – Компьютерные сети и телекоммуникации: задания и тесты – СПБ: СПБГУ ИТМО, 2018. – 112 с.
2. Т. И. Алиев – Сети ЭВМ и телекоммуникации – СПБ: СПБГУ ИТМО, 2011 – 400 с.