

ОТЧЕТ
ЗАЩИЩЕН С ОЦЕНКОЙ
ПРЕПОДАВАТЕЛЬ

ст. преподаватель

Н.В. Матвеев

должность, уч. степень, звание

подпись, дата

инициалы, фамилия

ОТЧЕТ О ЛАБОРАТОРНОЙ РАБОТЕ №1

АНАЛИЗ И ЭМУЛИРОВАНИЕ СЕТЕВОГО ТРАФИКА В ЛОКАЛЬНЫХ СЕТЯХ

по курсу: ОСНОВЫ КОНСТРУИРОВАНИЯ ТЕХНОЛОГИИ И ЭКСПЛУАТАЦИИ
ТЕЛЕКОММУНИКАЦИОННОГО ОБОРУДОВАНИЯ

РАБОТУ ВЫПОЛНИЛА

СТУДЕНТКА ГР. № 5721

А. Е. Ковалева

подпись, дата

инициалы, фамилия

Санкт-Петербург 2020

Цель работы: Освоение базовых навыков настройки и анализа трафика в локальных сетях передачи данных стандарта 802.3. Для анализа работы различных протоколов используется утилита Wireshark. Изучение основных методов настройки маршрутизируемых компьютерных сетей на примере сети, состоящей из компьютеров под управлением ОС Linux и маршрутизаторов на примере cisco. В процессе выполнения работы изучаются различные уровни стека протоколов TCP/IP. Производится базовая настройка связности в сети, управление таблицами маршрутизации и правилами трансляции сетевых адресов.

1 Допуск к лабораторной работе

Вариант 2 схема 2.

1.1 Постановка задачи

В эмуляторе GNS3 необходимо собирать схему из одного маршрутизатора cisco (R1) и трех виртуальных компьютеров (PC1, PC2, PC3). С помощью утилиты iperf организовать передачу данных на максимальной скорости по протоколу TCP, с использованием дополнительной опции AFFINITY CPU0. По заданию TCPDUMP перехватить все ACK пакеты.

1.2 Настройка оборудования

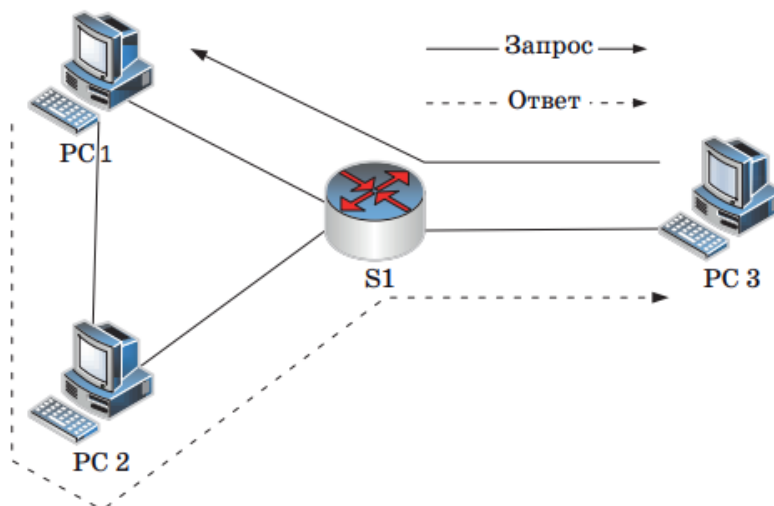


Рисунок 1 – Схема сети

В эмуляторе GNS 3 собирается схема из маршрутизатора cisco и компьютеров, которые являются виртуальными.

Ниже на рисунке 2 представлена схема построенной сети:

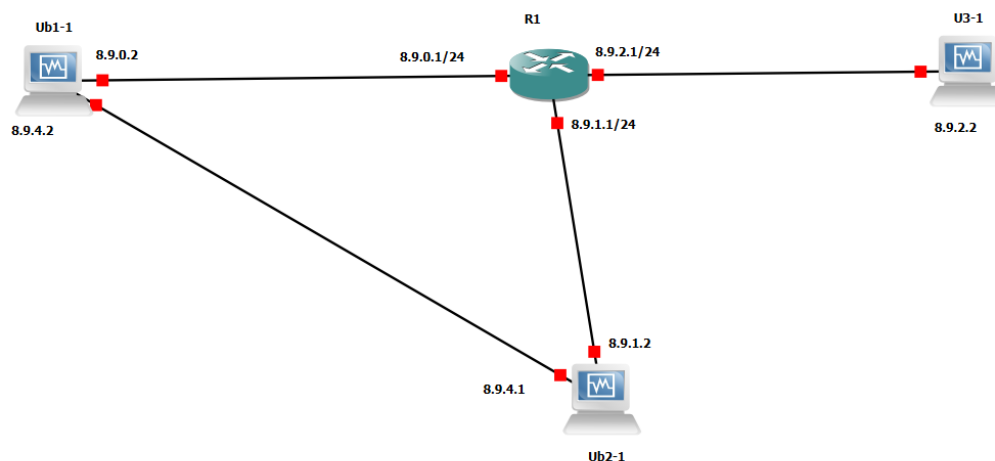


Рисунок 2 – Схема построенной сети

На всех адаптерах всех маршрутизаторов и компьютеров в топологии, настраиваем IP адреса (все интерфейсы компьютеров должны быть из разных подсетей).

IPv4 адрес выбирается следующим образом: А.В.Х.У/М,

где А – количество букв в имени студента;

В – количество букв в фамилии студента;

Х, У – числа, выбираемые студентом самостоятельно;

М – маска подсети (выбирается максимально длинная маска для обеспечения связности в сети).

Ниже на рисунке 3 представлен настроенный интерфейс маршрутизатора:

```
R1#conf t
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#int f0/1
R1(config-if)#ip addr 8.9.0.2 255.255.255.0
% 8.9.0.0 overlaps with FastEthernet0/0
R1(config-if)#ip addr 8.9.1.1 255.255.255.0
R1(config-if)#
R1#w
*Mar 1 00:09:24.027: %SYS-5-CONFIG_I: Configured from console by console
R1#write
Building configuration...
[OK]
R1#conf t
Enter configuration commands, one per line. End with CNTL/Z.
R1(config)#int f1/1
R1(config-if)#ip addr 8.9.2.1 255.255.255.0
R1(config-if)#
R1#w
*Mar 1 00:10:30.175: %SYS-5-CONFIG_I: Configured from console by console
R1#write
Building configuration...
[OK]
R1#sh ip int br
Interface                IP-Address      OK? Method Status      Protocol
FastEthernet0/0          8.9.0.1         YES manual up          up
FastEthernet0/1          8.9.1.1         YES manual up          up
FastEthernet1/0          8.9.2.1         YES manual up          up
R1#
```

Рисунок 3 – Настройка интерфейса маршрутизатора

На маршрутизаторах настраивается статическая маршрутизация, чтобы данные передавались согласно направлениям, обозначенным на схеме сети по варианту.

На всех компьютерах необходимо настроить сеть, используя консольную утилиту ip или ifconfig;

```
Ub1 [Работаer] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
To check for new updates run: sudo apt update
Last login: Sat Oct 24 08:25:43 UTC 2020 on tty1
kova@kova:~$ sudo ifconfig enp0s3 8.9.0.2 netmask 255.255.255.0
[sudo] password for kova:
kova@kova:~$ sudo route add default gw 8.9.0.1
kova@kova:~$ sudo ifconfig enp0s8 8.9.4.2 netmask 255.255.255.0
kova@kova:~$ sudo route add -net 8.9.2.0/24 gw 8.9.4.1
kova@kova:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 8.9.0.2 netmask 255.255.255.0 broadcast 8.9.0.255
    inet6 fe80::a00:27ff:fe5b:bbc2 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:5b:bb:c2 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 24 bytes 4896 (4.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 8.9.4.2 netmask 255.255.255.0 broadcast 8.9.4.255
    inet6 fe80::a00:27ff:fe1e:812 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:1e:08:12 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 9 bytes 726 (726.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 1144 bytes 81712 (81.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1144 bytes 81712 (81.7 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Рисунок 4 – Настройка сети на ПК 1

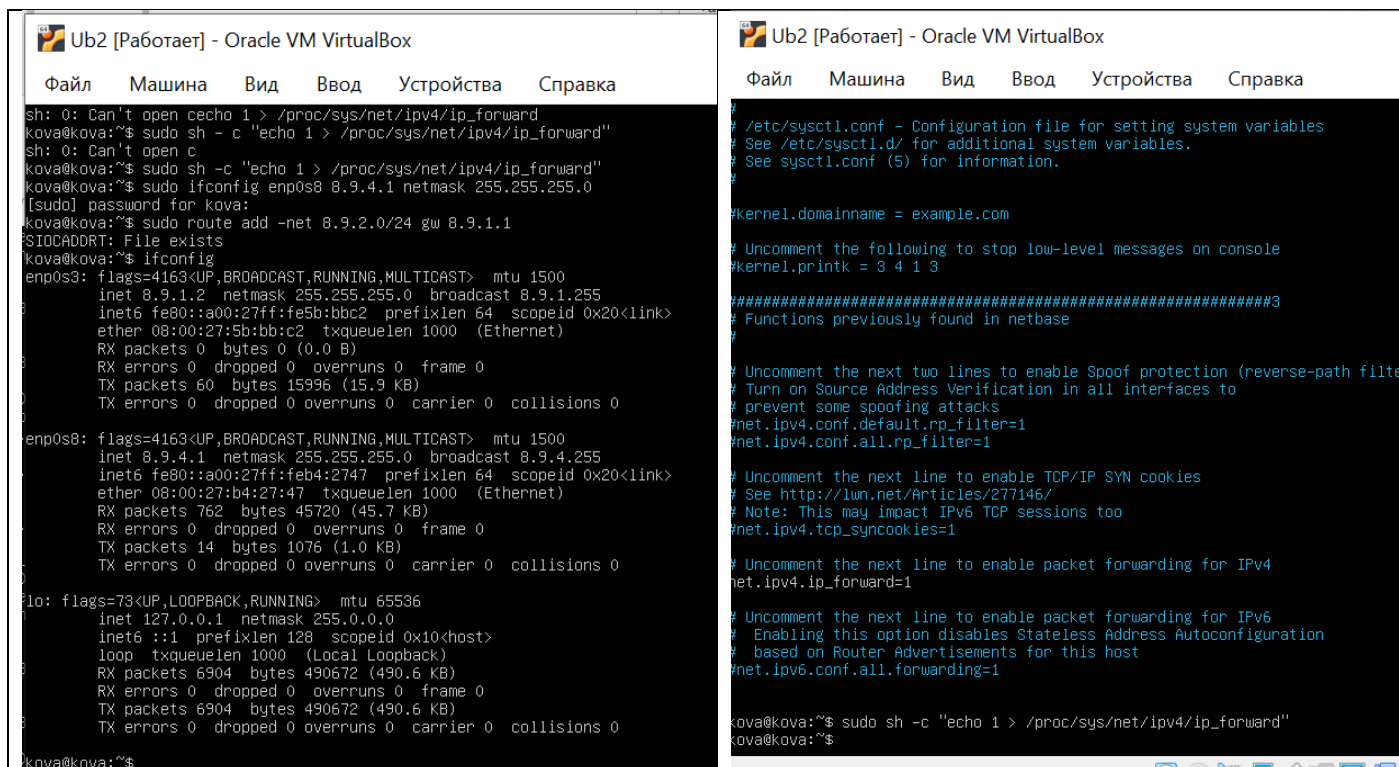


Рисунок 5 – Настройка сети на ПК2

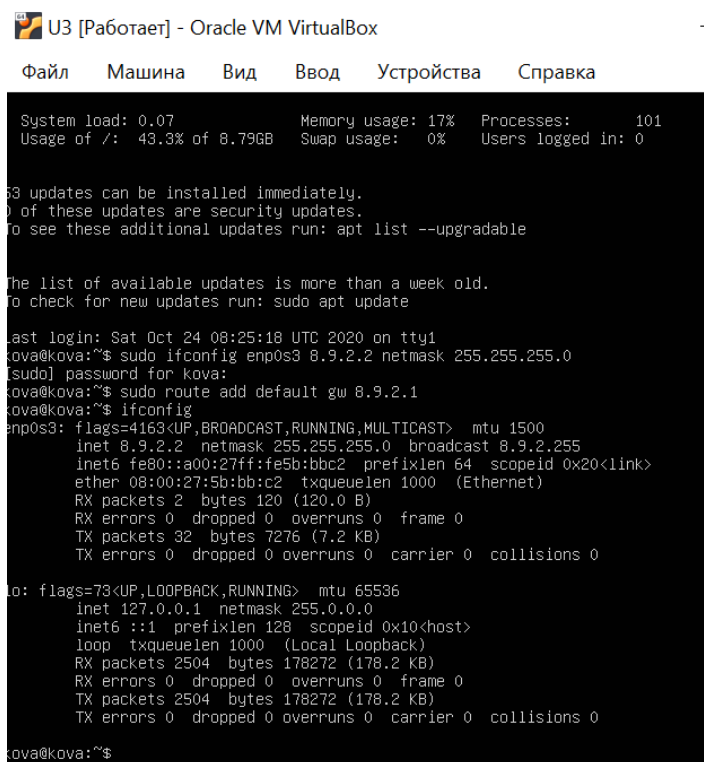


Рисунок 6 – Настройка сети на ПК3

Проверяем установленное подключение, а именно пингуем с третьего компьютера первый:

```
U3 [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка

inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10<host>
loop txqueuelen 1000 (Local Loopback)
RX packets 6344 bytes 450912 (450.9 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 6344 bytes 450912 (450.9 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

kova@kova:~$ ping 8.9.0.2
PING 8.9.0.2 (8.9.0.2) 56(84) bytes of data.
64 bytes from 8.9.0.2: icmp_seq=1 ttl=62 time=49.1 ms
64 bytes from 8.9.0.2: icmp_seq=2 ttl=62 time=18.1 ms
64 bytes from 8.9.0.2: icmp_seq=3 ttl=62 time=20.0 ms
64 bytes from 8.9.0.2: icmp_seq=4 ttl=62 time=22.0 ms
64 bytes from 8.9.0.2: icmp_seq=5 ttl=62 time=22.0 ms
64 bytes from 8.9.0.2: icmp_seq=6 ttl=62 time=13.3 ms
64 bytes from 8.9.0.2: icmp_seq=7 ttl=62 time=20.0 ms
64 bytes from 8.9.0.2: icmp_seq=8 ttl=62 time=21.0 ms
64 bytes from 8.9.0.2: icmp_seq=9 ttl=62 time=17.1 ms
64 bytes from 8.9.0.2: icmp_seq=10 ttl=62 time=20.9 ms
64 bytes from 8.9.0.2: icmp_seq=11 ttl=62 time=15.1 ms
64 bytes from 8.9.0.2: icmp_seq=12 ttl=62 time=16.0 ms
64 bytes from 8.9.0.2: icmp_seq=13 ttl=62 time=20.6 ms
64 bytes from 8.9.0.2: icmp_seq=14 ttl=62 time=14.2 ms
64 bytes from 8.9.0.2: icmp_seq=15 ttl=62 time=16.9 ms
64 bytes from 8.9.0.2: icmp_seq=16 ttl=62 time=19.9 ms
64 bytes from 8.9.0.2: icmp_seq=17 ttl=62 time=22.0 ms
64 bytes from 8.9.0.2: icmp_seq=18 ttl=62 time=14.0 ms
64 bytes from 8.9.0.2: icmp_seq=19 ttl=62 time=13.8 ms
64 bytes from 8.9.0.2: icmp_seq=20 ttl=62 time=18.3 ms
64 bytes from 8.9.0.2: icmp_seq=21 ttl=62 time=18.1 ms
64 bytes from 8.9.0.2: icmp_seq=22 ttl=62 time=21.4 ms
^C
--- 8.9.0.2 ping statistics ---
22 packets transmitted, 22 received, 0% packet loss, time 21041ms
rtt min/avg/max/mdev = 13.265/19.709/49.070/7.007 ms
kova@kova:~$
```

Рисунок 7 – ping ПК3 и ПК1

Команда Ping – одна из самых используемых сетевых утилит интерпретатора командной строки, присутствующая в стандартном наборе программных средств любой операционной системы с поддержкой сетевого взаимодействия.

Утилита Ping является самым простым и удобным средством проверки доступности удаленного узла.

Iperf – утилита, состоящая из клиентской и серверной части для тестирования скорости соединения между различными узлами

```
U3 [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка

kova@kova:~$ ping 8.9.0.2
PING 8.9.0.2 (8.9.0.2) 56(84) bytes of data.
64 bytes from 8.9.0.2: icmp_seq=1 ttl=62 time=49.1 ms
64 bytes from 8.9.0.2: icmp_seq=2 ttl=62 time=18.1 ms
64 bytes from 8.9.0.2: icmp_seq=3 ttl=62 time=20.0 ms
64 bytes from 8.9.0.2: icmp_seq=4 ttl=62 time=22.0 ms
64 bytes from 8.9.0.2: icmp_seq=5 ttl=62 time=22.0 ms
64 bytes from 8.9.0.2: icmp_seq=6 ttl=62 time=13.3 ms
64 bytes from 8.9.0.2: icmp_seq=7 ttl=62 time=20.0 ms
64 bytes from 8.9.0.2: icmp_seq=8 ttl=62 time=21.0 ms
64 bytes from 8.9.0.2: icmp_seq=9 ttl=62 time=17.1 ms
64 bytes from 8.9.0.2: icmp_seq=10 ttl=62 time=20.9 ms
64 bytes from 8.9.0.2: icmp_seq=11 ttl=62 time=15.1 ms
64 bytes from 8.9.0.2: icmp_seq=12 ttl=62 time=16.0 ms
64 bytes from 8.9.0.2: icmp_seq=13 ttl=62 time=20.6 ms
64 bytes from 8.9.0.2: icmp_seq=14 ttl=62 time=14.2 ms
64 bytes from 8.9.0.2: icmp_seq=15 ttl=62 time=16.9 ms
64 bytes from 8.9.0.2: icmp_seq=16 ttl=62 time=19.9 ms
64 bytes from 8.9.0.2: icmp_seq=17 ttl=62 time=22.0 ms
64 bytes from 8.9.0.2: icmp_seq=18 ttl=62 time=14.0 ms
64 bytes from 8.9.0.2: icmp_seq=19 ttl=62 time=13.8 ms
64 bytes from 8.9.0.2: icmp_seq=20 ttl=62 time=18.3 ms
64 bytes from 8.9.0.2: icmp_seq=21 ttl=62 time=18.1 ms
64 bytes from 8.9.0.2: icmp_seq=22 ttl=62 time=21.4 ms
^C
--- 8.9.0.2 ping statistics ---
22 packets transmitted, 22 received, 0% packet loss, time 21041ms
rtt min/avg/max/mdev = 13.265/19.709/49.070/7.007 ms
kova@kova:~$ iperf -c 8.9.0.2
Client connecting to 8.9.0.2, TCP port 5001
TCP window size: 85.0 KByte (default)
[ 3] local 8.9.2.2 port 36648 connected with 8.9.0.2 port 5001
[ ID] Interval           Transfer     Bandwidth
[ 3] 0.0-10.7 sec      416 KBytes    318 Kbits/sec
kova@kova:~$

Ub1 [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод

TX packets 9 bytes 726 (726.0 B)
TX errors 0 dropped 0 overruns

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10<host>
loop txqueuelen 1000 (Local Loopback)
RX packets 1144 bytes 81712 (81.7 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 1144 bytes 81712 (81.7 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

kova@kova:~$ iperf -s
Server listening on TCP port 5001
TCP window size: 128 KByte (default)
^Ckova@kova:~$ iperf -s
Server listening on TCP port 5001
TCP window size: 128 KByte (default)
```

Рисунок 8 – Демонстрация работы iperf

Задание под вариантом 2

Iperf это клиент-серверная утилита которая позволять производить замеры пропускной способности канала. Клиент-серверная утилита подразумевает под собой, что для проверки скорости между двумя ПК необходимо будет запустить iperf на одном ПК в режиме «сервер», а на другом ПК в режиме «клиент».

Скорость измеряется от клиента к серверу, т.е. если вы на своем компьютере запустили iperf в режиме «клиент», то результатом будет «исходящая» скорость.

Чтобы измерить входящую, необходимо запустить iperf в режиме сервер, либо воспользоваться ключиком -d для измерения скорости входящей+исходящей.

Важно отметить, что генерирует трафик только клиентская часть.

-s, —server, запустить в режиме сервера

-c, —client <host>, запустить в режиме клиента, при подключении к <host>

-t, —time n, время передачи в секундах (по умолчанию 10 секунд)

-i, —interval n, пауза секунд между периодическими отчётами

```
Client connecting to 8.9.1.2, TCP port 5001
TCP window size: 85.0 KByte (default)

[ 3] local 8.9.2.2 port 55848 connected with 8.9.1.2 port 5001
[ ID] Interval      Transfer      Bandwidth
[ 3] 0.0-10.0 sec    2.75 MBytes    2.31 Mbits/sec
[ 3] 10.0-20.0 sec   1.36 MBytes    1.14 Mbits/sec
[ 3] 20.0-30.0 sec   1.18 MBytes    993 Kbits/sec
[ 3] 30.0-40.0 sec   1.12 MBytes    938 Kbits/sec
[ 3] 40.0-50.0 sec   1.24 MBytes    1.04 Mbits/sec
[ 3] 50.0-60.0 sec   1.86 MBytes    1.56 Mbits/sec
[ 3] 60.0-70.0 sec    636 KBytes     521 Kbits/sec
[ 3] 70.0-80.0 sec   1.30 MBytes    1.09 Mbits/sec
[ 3] 80.0-90.0 sec   1.30 MBytes    1.09 Mbits/sec
[ 3] 90.0-100.0 sec  1.30 MBytes    1.09 Mbits/sec
[ 3] 100.0-110.0 sec 1.24 MBytes    1.04 Mbits/sec
[ 3] 110.0-120.0 sec 1.30 MBytes    1.09 Mbits/sec
[ 3] 0.0-120.5 sec  16.6 MBytes    1.16 Mbits/sec
kova@kova:~$
```

Рисунок 9 – С помощью утилиты iperf организована передача данных на максимальной скорости по протоколу TCP

С помощью утилиты wireshark посмотрим трафик между маршрутизаторами прямо в GNS 3:

No.	Time	Source	Destination	Protocol	Length	Info
1422	18.273694	8.9.2.2	8.9.1.2	TCP	1514	55852 → 5001 [PSH, ACK] Seq=1271345 Ack=1 Win=64256 Len=1448 TSval=2064253775 TSecr=3087802
1423	18.274671	8.9.1.2	8.9.2.2	TCP	66	5001 → 55852 [ACK] Seq=1 Ack=1272793 Win=577792 Len=0 TSval=3087803836 TSecr=2064253775
1424	18.284469	8.9.2.2	8.9.1.2	TCP	1514	55852 → 5001 [ACK] Seq=1272793 Ack=1 Win=64256 Len=1448 TSval=2064253790 TSecr=3087802186
1425	18.285414	8.9.1.2	8.9.2.2	TCP	66	5001 → 55852 [ACK] Seq=1 Ack=1274241 Win=580736 Len=0 TSval=3087803847 TSecr=2064253790
1426	18.295212	8.9.2.2	8.9.1.2	TCP	1514	55852 → 5001 [ACK] Seq=1274241 Ack=1 Win=64256 Len=1448 TSval=2064253799 TSecr=3087802199
1427	18.296145	8.9.1.2	8.9.2.2	TCP	66	5001 → 55852 [ACK] Seq=1 Ack=1275689 Win=583552 Len=0 TSval=3087803858 TSecr=2064253799
1428	18.305904	8.9.2.2	8.9.1.2	TCP	1514	55852 → 5001 [PSH, ACK] Seq=1275689 Ack=1 Win=64256 Len=1448 TSval=2064253799 TSecr=3087802
1429	18.306880	8.9.1.2	8.9.2.2	TCP	66	5001 → 55852 [ACK] Seq=1 Ack=1277137 Win=586496 Len=0 TSval=3087803868 TSecr=2064253799
1430	18.316638	8.9.2.2	8.9.1.2	TCP	1514	55852 → 5001 [ACK] Seq=1277137 Ack=1 Win=64256 Len=1448 TSval=2064253809 TSecr=3087802210
1431	18.317617	8.9.1.2	8.9.2.2	TCP	66	5001 → 55852 [ACK] Seq=1 Ack=1278585 Win=589312 Len=0 TSval=3087803879 TSecr=2064253809
1432	18.327375	8.9.2.2	8.9.1.2	TCP	1514	55852 → 5001 [ACK] Seq=1278585 Ack=1 Win=64256 Len=1448 TSval=2064253820 TSecr=3087802221
1433	18.328353	8.9.1.2	8.9.2.2	TCP	66	5001 → 55852 [ACK] Seq=1 Ack=1280033 Win=592256 Len=0 TSval=3087803890 TSecr=2064253820
1434	18.338110	8.9.2.2	8.9.1.2	TCP	1514	55852 → 5001 [PSH, ACK] Seq=1280033 Ack=1 Win=64256 Len=1448 TSval=2064253820 TSecr=3087802
1435	18.338110	8.9.1.2	8.9.2.2	TCP	66	5001 → 55852 [ACK] Seq=1 Ack=1281481 Win=595200 Len=0 TSval=3087803900 TSecr=2064253820

> Frame 1: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface -, id 0

> Ethernet II, Src: c2:01:1c:e8:00:01 (c2:01:1c:e8:00:01), Dst: c2:01:1c:e8:00:01 (c2:01:1c:e8:00:01)

> Configuration Test Protocol (loopback)

> Data (40 bytes)

Рисунок 10 – Анализ работы протокола с использованием утилиты Wireshark

Утилита tcpdump относится к числу так называемых «снифферов» — программ, предназначенных для перехвата сетевого трафика. Одним словом, tcpdump предназначен для подслушивания.

Он позволяет просматривать все входящие и исходящие из определенного интерфейса пакеты и работает в командной строке.

Формат команды tcpdump следующий:

tcpdump [-опции] [фильтры]

Для поиска точных значений используется оператор: != Не равно

```
kova@kova:~$ sudo tcpdump "cp-ak != 0"
[sudo] password for kova:
tcpdump: can't parse filter expression: syntax error
kova@kova:~$ sudo tcpdump "tcp-ak != 0"
tcpdump: can't parse filter expression: syntax error
kova@kova:~$ sudo tcpdump "tcp-ack != 0"
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
20:16:01.631758 CDPv2, ttl: 180s, Device-ID 'R1', length 342
20:16:03.370788 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:16:13.666755 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:16:23.987821 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:16:34.283666 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:16:41.927570 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP, Request from 08:00:27:5b:bb:c2 (oui Unknown), length 288
20:16:44.558158 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:16:54.832458 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:17:03.088310 CDPv2, ttl: 180s, Device-ID 'R1', length 342
20:17:04.827650 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:17:15.123436 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:17:25.419282 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:17:35.715061 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:17:45.070833 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP, Request from 08:00:27:5b:bb:c2 (oui Unknown), length 288
20:17:46.083187 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:17:56.437148 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:18:04.766746 CDPv2, ttl: 180s, Device-ID 'R1', length 342
20:18:06.514474 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
```

Рисунок 11 – Перехват всех АСК пакетов

48	326.395447	c2:01:1c:e8:00:00	c2:01:1c:e8:00:00	LOOP	60	Reply	
49	336.500994	c2:01:1c:e8:00:00	c2:01:1c:e8:00:00	LOOP	60	Reply	
50	345.106345	c2:01:1c:e8:00:00	CDP/VTP/DTP/PagP/U...	CDP	364	Device ID: R1 Port ID:	
51	346.727478	c2:01:1c:e8:00:00	c2:01:1c:e8:00:00	LOOP	60	Reply	
52	357.023344	c2:01:1c:e8:00:00	c2:01:1c:e8:00:00	LOOP	60	Reply	
53	359.913240	0.0.0.0	255.255.255.255	DHCP	330	DHCP Discover - Transac	
54	367.276181	c2:01:1c:e8:00:00	c2:01:1c:e8:00:00	LOOP	60	Reply	
55	377.572007	c2:01:1c:e8:00:00	c2:01:1c:e8:00:00	LOOP	60	Reply	
56	387.846402	c2:01:1c:e8:00:00	c2:01:1c:e8:00:00	LOOP	60	Reply	
57	397.905989	c2:01:1c:e8:00:00	c2:01:1c:e8:00:00	LOOP	60	Reply	
58	406.505524	c2:01:1c:e8:00:00	CDP/VTP/DTP/PagP/U...	CDP	364	Device ID: R1 Port ID:	
59	408.126658	c2:01:1c:e8:00:00	c2:01:1c:e8:00:00	LOOP	60	Reply	
60	418.422485	c2:01:1c:e8:00:00	c2:01:1c:e8:00:00	LOOP	60	Reply	
61	424.280435	0.0.0.0	255.255.255.255	DHCP	330	DHCP Discover - Transac	
62	428.687073	c2:01:1c:e8:00:00	c2:01:1c:e8:00:00	LOOP	60	Reply	

```
l 3] 40.0-50.0 sec 1.24 Mbytes 1.04 Mb/s/sec
l 3] 50.0-60.0 sec 1.80 Mbytes 1.51 Mb/s/sec
l 3] 60.0-70.0 sec 1020 Kbytes 836 Kbits/sec
[2]+ Stopped iperf -c 8.9.1.2 -t 120 -i 10
kova@kova:~$ sudo tcpdump "tcp-ack != 0"
[sudo] password for kova:
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on enp0s3, link-type EN10MB (Ethernet), capture size 262144 bytes
20:56:48.031243 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:56:56.529317 CDPv2, ttl: 180s, Device-ID 'R1', length 342
20:56:58.257636 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:57:08.553606 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:57:13.344837 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP, Request from 08:00:27:5b:bb:c2 (oui Unknown), length 288
20:57:18.806357 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:57:29.102263 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:57:39.376681 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:57:49.436232 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:57:57.328343 CDPv2, ttl: 180s, Device-ID 'R1', length 342
20:57:59.658817 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:58:09.352795 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
20:58:17.384822 IP 0.0.0.0.bootpc > 255.255.255.255.bootps: BOOTP/DHCP, Request from 08:00:27:5b:bb:c2 (oui Unknown), length 288
20:58:20.217282 Loopback, skipCount 0, Reply, receipt number 0, data (40 octets)
```

Рисунок 12 – Анализ работы протокола с использованием утилиты Wireshark

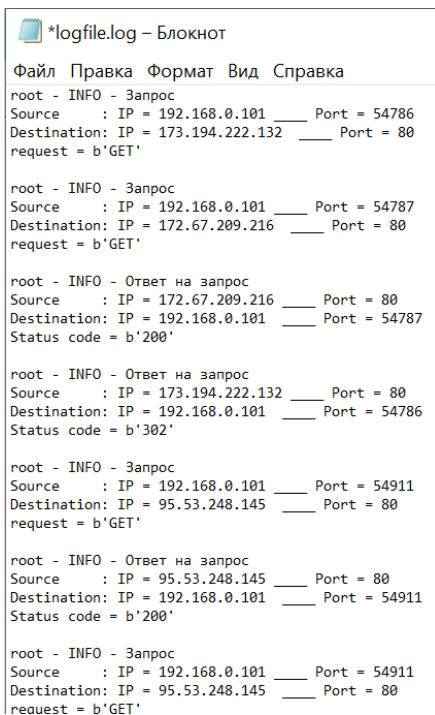
Лабораторная работа

Вариант 2

Перехватить пакеты по протоколу http, записать в лог файл IP адрес и PORT источника и получателя для запросов (GET, POST, и т. д.) и ответов на запрос (200, 300 и т. д.).

Листинг программы

```
from scapy.all import *
import logging
from scapy.layers.http import HTTPResponse, HTTPRequest, HTTP
from scapy.layers.inet import IP, TCP
#распаковываем пакет
def parse_packet(packet1):
    ret = ""
    if not HTTP in packet1:
        return ret
    #вернуть результат вместо печати (по умолчанию False)
    ret = "***GET PACKET***"
    ip_source = 'None' #источник
    ip_destination = 'None' #получатель
    if IP in packet1:
        ip_source = packet1[IP].src
        ip_destination = packet1[IP].dst
    tcp_source_port = '0'
    tcp_destination_port = '0'
    if TCP in packet1:
        tcp_source_port = str(packet1[TCP].sport)
        tcp_destination_port = str(packet1[TCP].dport)
    if HTTPRequest in packet1:
        method = str(packet1[HTTPRequest].Method)
        logging.info('Запрос\nSource      : IP = ' + ip_source + ' ____ Port = ' + tcp_source_port + \
                    '\nDestination: IP = ' + ip_destination + ' ____ Port = ' + tcp_destination_port + \
                    '\nrequest = ' + method + '\n\n')
    if HTTPResponse in packet1:
        reply_code = str(packet1[HTTPResponse].Status_Code)
        logging.info('Ответ на запрос\nSource      : IP = ' + ip_source + ' ____ Port = ' + tcp_source_port + \
                    '\nDestination: IP = ' + ip_destination + ' ____ Port = ' + tcp_destination_port + \
                    '\nStatus code = ' + reply_code + '\n\n')
    return ret
logging.basicConfig(filename='logfile.log', filemode='w+', format='%(name)s - %(levelname)s - %(message)s', \
                    level=logging.INFO)
sniff(iface="Беспроводная сеть", prn=parse_packet)
```



```
*logfile.log - Блокнот
Файл  Правка  Формат  Вид  Справка
root - INFO - Запрос
Source      : IP = 192.168.0.101 ____ Port = 54786
Destination: IP = 173.194.222.132 ____ Port = 80
request = b'GET'

root - INFO - Запрос
Source      : IP = 192.168.0.101 ____ Port = 54787
Destination: IP = 172.67.209.216 ____ Port = 80
request = b'GET'

root - INFO - Ответ на запрос
Source      : IP = 172.67.209.216 ____ Port = 80
Destination: IP = 192.168.0.101 ____ Port = 54787
Status code = b'200'

root - INFO - Ответ на запрос
Source      : IP = 173.194.222.132 ____ Port = 80
Destination: IP = 192.168.0.101 ____ Port = 54786
Status code = b'302'

root - INFO - Запрос
Source      : IP = 192.168.0.101 ____ Port = 54911
Destination: IP = 95.53.248.145 ____ Port = 80
request = b'GET'

root - INFO - Ответ на запрос
Source      : IP = 95.53.248.145 ____ Port = 80
Destination: IP = 192.168.0.101 ____ Port = 54911
Status code = b'200'

root - INFO - Запрос
Source      : IP = 192.168.0.101 ____ Port = 54911
Destination: IP = 95.53.248.145 ____ Port = 80
request = b'GET'
```

Рисунок 13 – Записи в log файле

3 Выводы

В ходе лабораторной работы:

1. Были освоены базовые навыки настройки и анализа трафика в локальных сетях передачи данных стандарта 802.3. Для анализа работы различных протоколов была использована утилита Wireshark.
2. Были изучены основные методы настройки маршрутизируемых компьютерных сетей на примере сети, состоящей из компьютеров и маршрутизаторов на примере cisco.
3. В процессе выполнения работы были изучены различные уровни стека протоколов TCP/IP.
4. Была произведена базовая настройка связности в сети, управление таблицами маршрутизации и правилами трансляции сетевых адресов
5. Была настроена схема, состоящая из 3 PC и 1 маршрутизатора. В качестве PC использовались виртуальные машины;
6. Между ПК были настроены статистические маршруты;
7. Была протестирована утилита iperf;
8. Была организована передача данных на максимальной скорости по протоколу TCP;
9. По заданию TCPDUMP были перехвачены все ACK пакеты.
10. Была написана программа перехвата сетевого трафика по заданному протоколу http;