МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра космического приборостроения и систем связи

УТВЕЖДАЮ
Проректор по учебной работе
____________О.Г.Локтионова
«_9_» _04_________2024 г.

ПРОЕКТИРОВАНИЕ КАБЕЛЬНЫХ СИСТЕМ ДОСТУПА

Методические указания
по выполнению практических работ для студентов, обучающихся по направлению подготовки
11.04.02 «Инфокоммуникационные технологии и системы связи»
по курсу «Проектирование кабельных систем доступа»

Курск 2024

УДК 621.391

Составители: И.Г. Бабанин, Е.Ю. Бабанина

Рецензент
Доктор технических наук, старший научный сотрудник,
заведующий кафедрой *В.Г. Андроноа*

Проектирование кабельных систем доступа: методические указания по выполнению практических работ по курсу «Проектирование кабельных систем доступа» / Юго-Зап. гос. ун-т; сост.: И.Г. Бабанин, Е.Ю. Бабанина.– Курск, 2024. – 84 с.

Методические указания по выполнению практических работ содержат краткие теоретические сведения о сетевых установках, правила выполнения практических работ по курсу «Проектирование кабельных систем доступа», требования к оформлению отчёта.

Методические указания полностью соответствуют учебному плану по направлению подготовки 11.04.02 «Инфокоммуникационные технологии и системы связи», а также рабочей программы дисциплины «Проектирование кабельных систем доступа».

Предназначены для студентов, обучающихся по направлению подготовки 11.04.02 «Инфокоммуникационные технологии и системы связи» очной и заочной форм обучения.

Текст печатается в авторской редакции

Подписано в печать 9.04.24. Формат 60x84 1/16.
Усл. печ. л. 4,88.Уч.-изд. л. 4.42. Тираж 100 экз. Заказ 604 .Бесплатно.
Юго-Западный государственный университет.
305040, г. Курск, ул. 50 лет Октября, 94.

1. Анализ трафика локальной сети на примере протоколов ARP, DNS и HTTP

Цель работы: Исследование основных типов трафика в реальной локальной сети на примере наиболее распространенных протоколов стека TCP/IP.

Краткая теоретическая справка

В работе исследуется два типа трафика: broadcast (широковещательный, на примере протокола ARP) и unicast (на примере протокола http). Предполагается, что к моменту выполнения лабораторной работы из курса лекций известна структура и основные принципы работы протоколов ARP, DNS и http, а также структура кадра Ethernet. Лабораторная работа проводится с использованием свободно распространяемого (лицензия GNU GPL) анализатора трафика Wireshark.

В настоящее время существует достаточно большой выбор анализаторов трафика – специальных программ, позволяющих получить информацию о пакетах, передающихся по исследуемой сети. Все они, несмотря на различных разработчиков, объединены одной идеей: предоставить возможность не только определить тип пакета, но и его структуру. Информация о структуре пакета в Wireshark выводится в виде таблиц анализа (рис. 1).

Таблица анализа представляет собой три поля, которые заполняются динамически по мере поступления пакетов. В основной части представлена информация о соединении: МАС-адреса источника и получателя, IP-адреса источника и получателя, протокол, порт отправителя и порт получателя. Обратите внимание – зарезервированные порты часто обозначаются именем протокола: например, порты 80 и 8080 могут обозначаться как http. Одна строка в этой таблице относится к одному пакету. При выделении строки в двух других окнах появляется информация о структуре пакета.

Поле со структурой пакета позволяет определить, как заполнены поля протоколов в соответствии со стандартом. Это позволяет определить адреса отправителя и получателя, номер порта, корректность контрольной суммы и т.п.

Поле с представлением пакета в ASCII кодах и 16-ричной системе дает представление о реальном виде пакета при передаче по сети. В большинстве анализаторов предусмотрена возможность выделения пункта в поле со структурой пакета и одновременное выделение соответствующих знаков в поле с представлением пакета в 16-ричной системе.

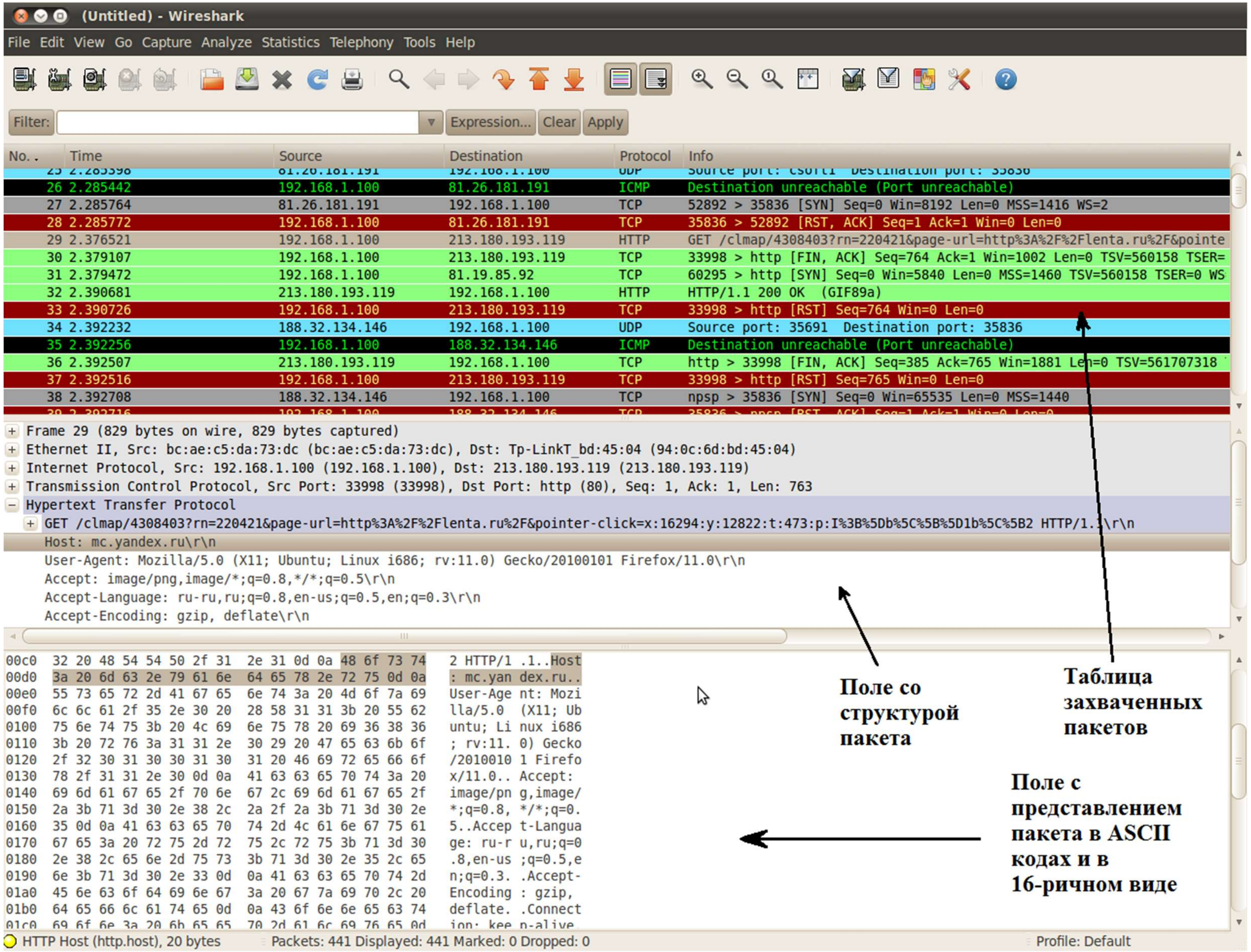
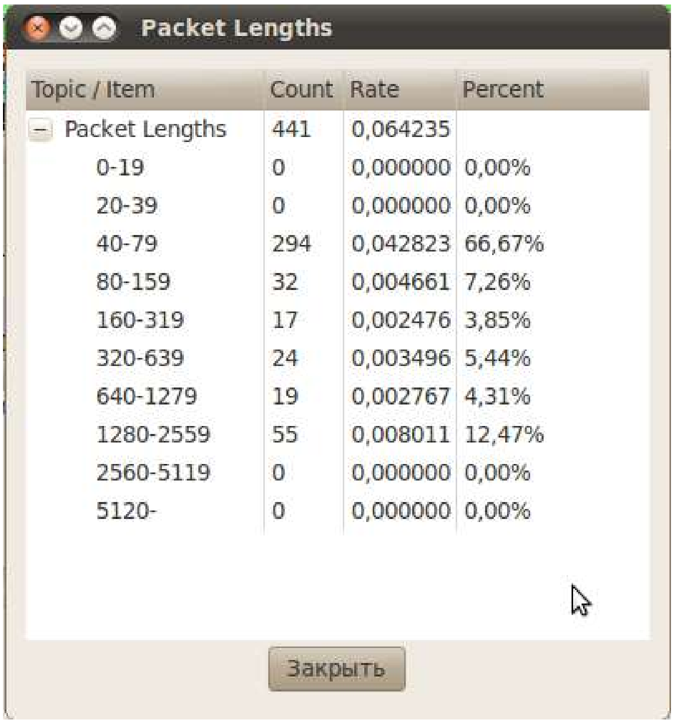


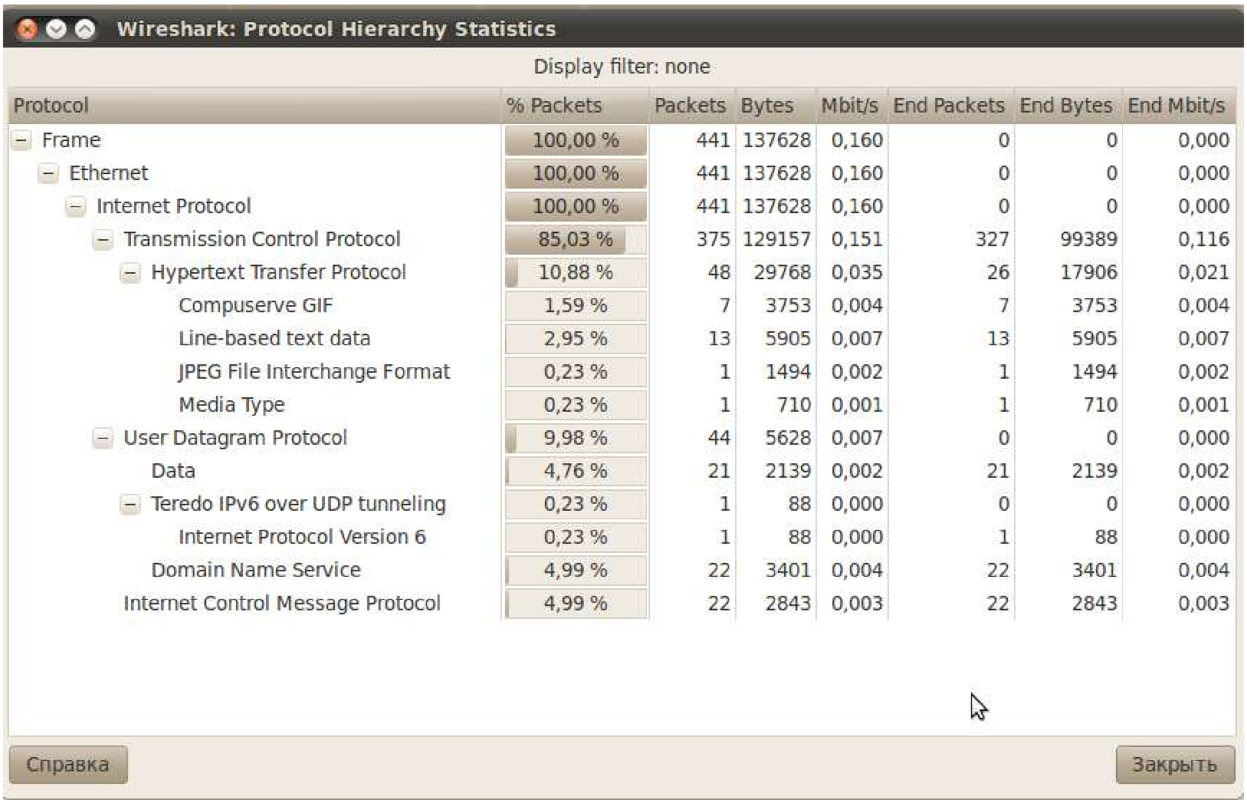
Рис. 1 – Пример таблицы анализа: пакет http, запрос к серверу

Также анализатор трафика позволяет собрать статистику о пакетах, проходящих по сети на различных уровнях модели TCP/IP. В качестве примера на рисунке 2 представлена статистика по размерам пакетов (Statistics → Packet Lengths), а на рисунке 3 – по типам протоколов и видам данных, встречающихся в захваченном трафике (Statistics → Protocol Hierarchy).



Topic / Item	Count	Rate	Percent
Packet Lengths	441	0,064235	
0-19	0	0,000000	0,00%
20-39	0	0,000000	0,00%
40-79	294	0,042823	66,67%
80-159	32	0,004661	7,26%
160-319	17	0,002476	3,85%
320-639	24	0,003496	5,44%
640-1279	19	0,002767	4,31%
1280-2559	55	0,008011	12,47%
2560-5119	0	0,000000	0,00%
5120-	0	0,000000	0,00%

Рис. 2 – Пример статистики по размерам пакетов



Protocol	% Packets	Packets	Bytes	Mbit/s	End Packets	End Bytes	End Mbit/s
Frame	100,00 %	441	137628	0,160	0	0	0,000
Ethernet	100,00 %	441	137628	0,160	0	0	0,000
Internet Protocol	100,00 %	441	137628	0,160	0	0	0,000
Transmission Control Protocol	85,03 %	375	129157	0,151	327	99389	0,116
Hypertext Transfer Protocol	10,88 %	48	29768	0,035	26	17906	0,021
Compuserve GIF	1,59 %	7	3753	0,004	7	3753	0,004
Line-based text data	2,95 %	13	5905	0,007	13	5905	0,007
JPEG File Interchange Format	0,23 %	1	1494	0,002	1	1494	0,002
Media Type	0,23 %	1	710	0,001	1	710	0,001
User Datagram Protocol	9,98 %	44	5628	0,007	0	0	0,000
Data	4,76 %	21	2139	0,002	21	2139	0,002
Teredo IPv6 over UDP tunneling	0,23 %	1	88	0,000	0	0	0,000
Internet Protocol Version 6	0,23 %	1	88	0,000	1	88	0,000
Domain Name Service	4,99 %	22	3401	0,004	22	3401	0,004
Internet Control Message Protocol	4,99 %	22	2843	0,003	22	2843	0,003

Рис. 3 – Пример статистики по типам протоколов.

При необходимости можно визуально оценить на общем графике интенсивность интересующих видов трафика по протоколам (до пяти протоколов одновременно), настроив соответствующим образом фильтры в окне IO Graphs (как показано на рисунке 4). Данный инструмент находится в меню Statistics → IO Graphs.

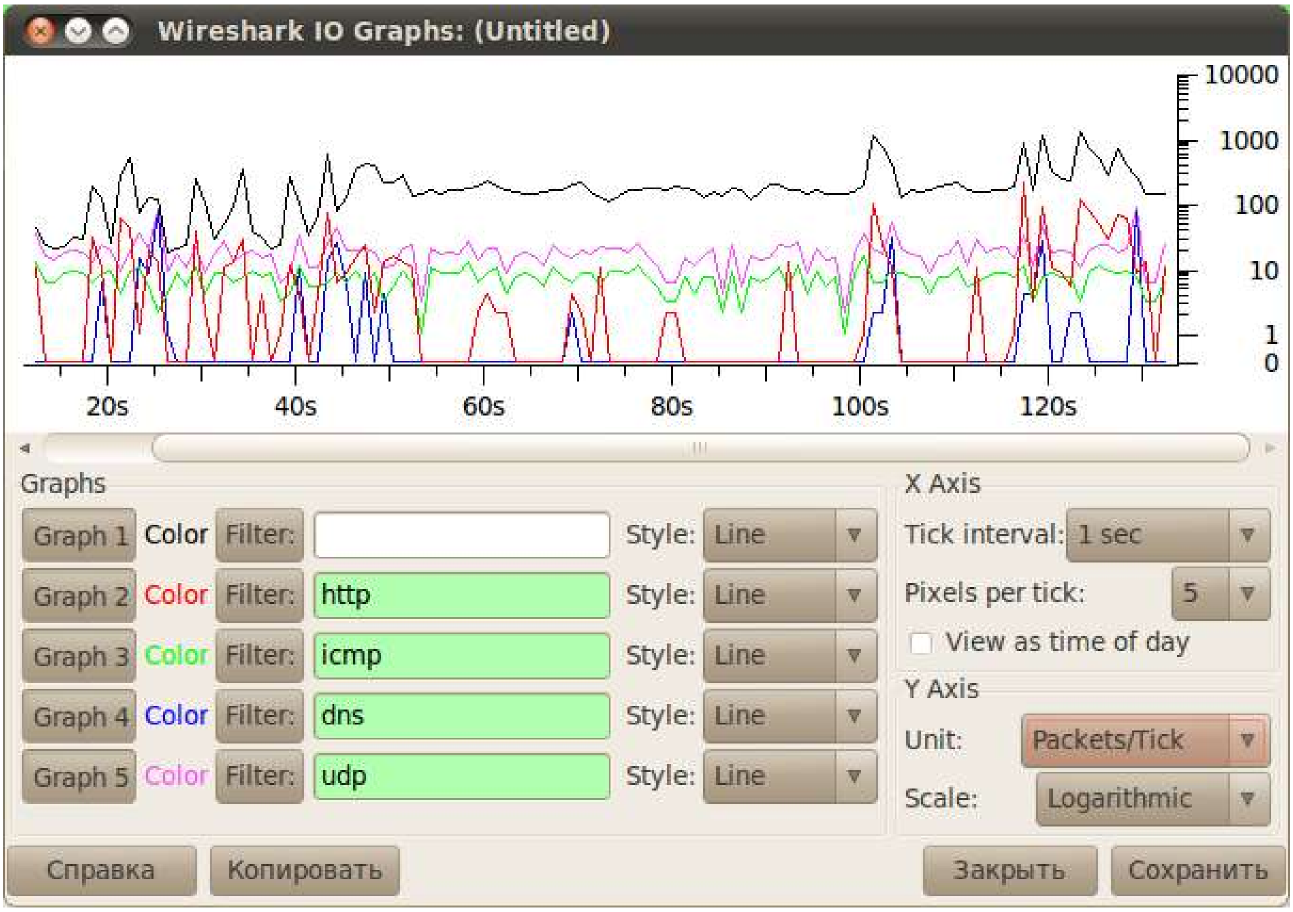


Рис. 4 – Графики интенсивности захваченного трафика.

Кроме того, в большинстве анализаторов можно увидеть статистику по хостам. На сетевом уровне такая статистика (Statistics → IP Destinations) позволяет оценить долю трафика соответствующую конкретным IP-адресам и используемым ими протоколам (см. рис. 5).

Инструмент Graph Analysis из меню Statistics, окно которого показано на рисунке 6, позволяет в графическом виде представлять процедуры обмена данными на основании захваченного трафика.

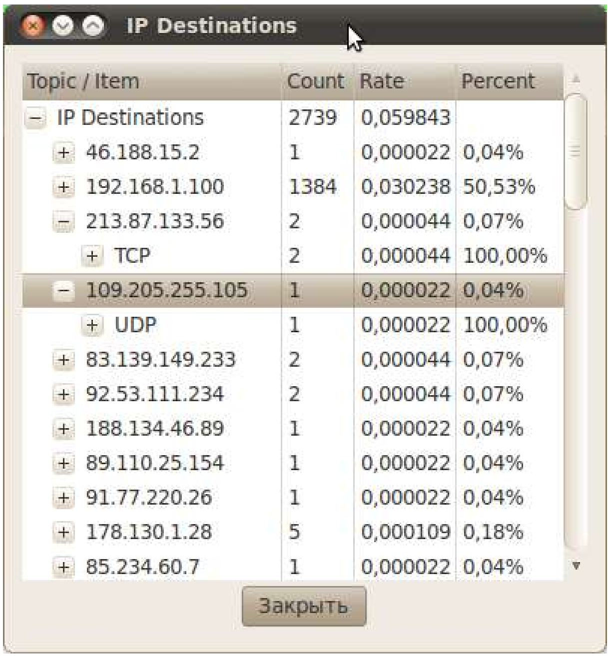


Рис. 5 – Статистика IP-адресов в захваченном трафике

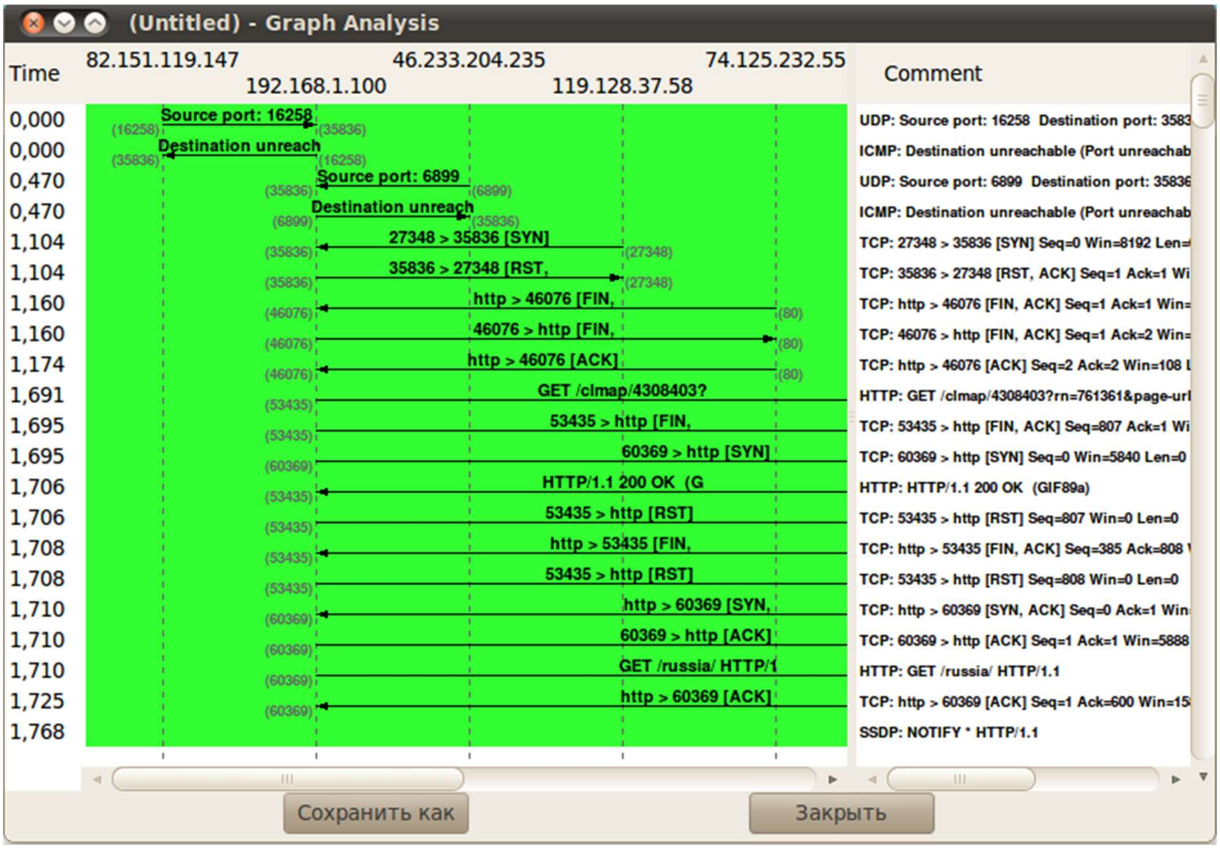


Рис. 6 – Диаграммы соединений.

Задание на работу:

1. Запустить анализатор трафика Wireshark. После выбора требуемого сетевого интерфейса кнопкой Start начать сбор трафика (см. рис. 7).

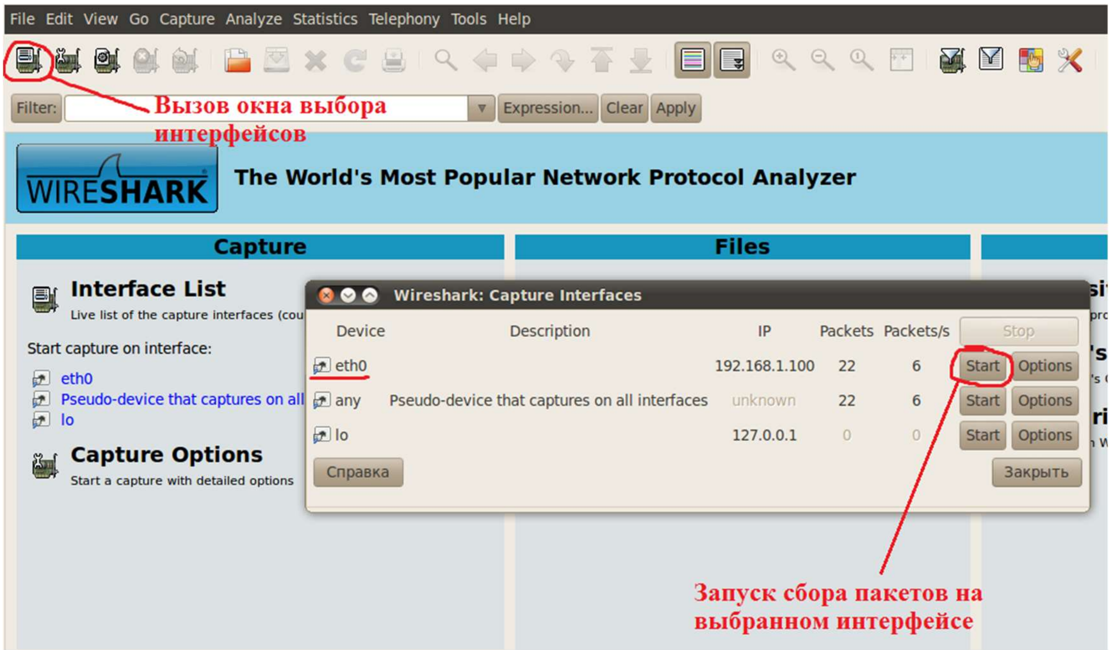


Рис. 7 – Запуск процедуры сбора пакетов

2. Настроить фильтр на широковещательный трафик (см. рис. 8): нажатием на кнопку Filter в панели инструментов Wireshark запустить окно выбора/создания фильтра, ввести любое имя для нового фильтра в поле Filter Name, а в качестве фильтрующего выражения (Filter String) выставить название протокола «arp». Для применения фильтра нажмите кнопку «Apply» на панели инструментов.

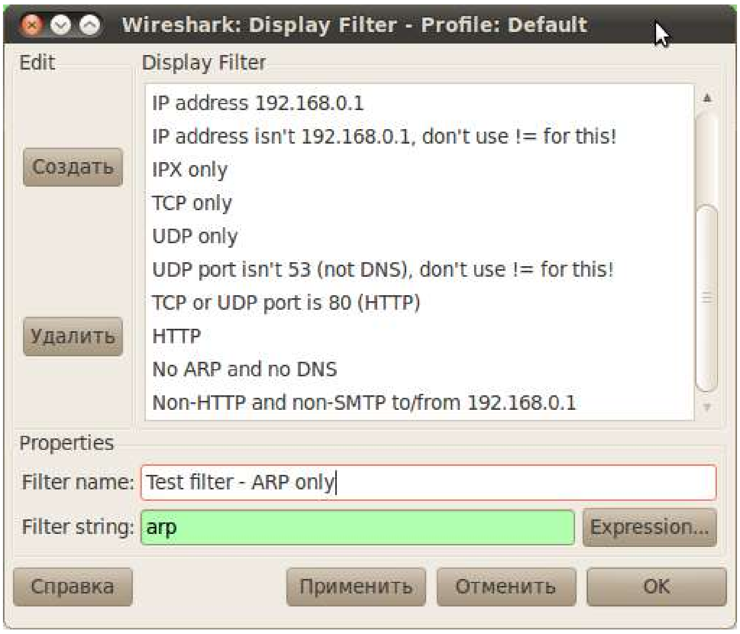


Рис. 8 – Настройка фильтра для анализа ARP-пакетов

3. Разобрать пакет ARP: запрос и ответ. Снять скриншоты экранов с таблицами анализа протокола ARP.

4. Убрать настройку фильтров. Запустить браузер, набрать URL какого-либо веб-ресурса (по желанию студента или заданию преподавателя). Отследить и разобрать пакеты DNS (запрос и ответ).

5. Разобрать пакеты http двух типов: запрос (GET) и ответ сервера. Снять скриншоты экранов: таблицы анализа, график интенсивности трафика HTTP в общем трафике, диаграмму соединений.

6. На основании данных об IP-адресах в полученном трэйсе и инструмента Graph Analysis, построить карту сети (см. рис. 9), указав с помощью соединительных линий логические связи (т.е. наличие соединений) между хостами.

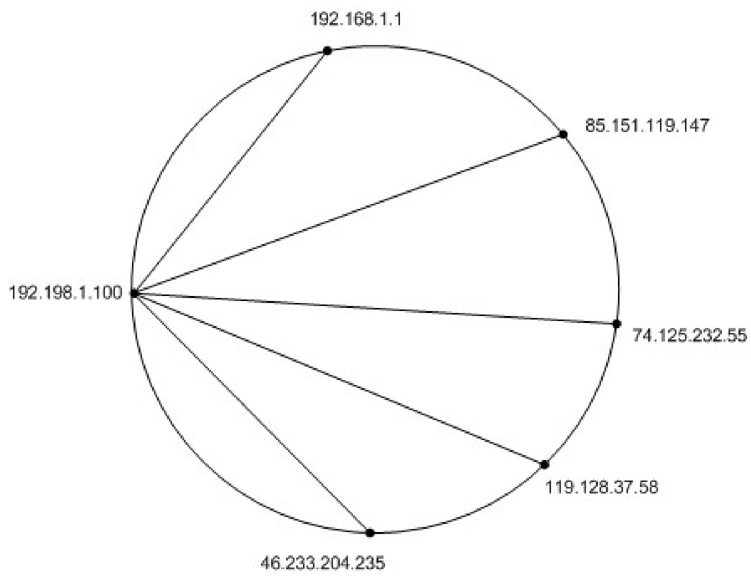


Рис. 9 – Пример карты сети

К защите:

1. Знать принципы формирования пакетов в локальных сетях (технологии IP и Ethernet).

2. Иметь представление о функциях и процессе формирования пакетов протоколов ARP и HTTP, запросов/ответов DNS, понимать особенности широковещательного трафика.

3. Представить отчет, содержащий скриншоты для всех исследуемых протоколов: таблицы с анализом трафика, графики, диаграммы.

Рекомендуемая литература:

1. Wireshark User Guide, U.Lamping, R.Sharpe, E.Warnicke. http://www.wireshark.org/docs/wsug_html_chunked.

2. Олифер Н. А., Олифер В.Г. Компьютерные сети. Принципы, технологии, протоколы. Издание 4-е. Питер, 2010.

2. Использование генератора трафика для создания нагрузки в сети

Цель работы: Изучение принципов генерации трафика с помощью программы Ostinato на примере наиболее распространенных протоколов стека TCP/IP.

Краткая теоретическая справка

Генераторы трафика являются необходимыми инструментами для проверки работоспособности сетей как на этапе тестовых испытаний, так и при разработке новых услуг и технологий. Генераторы трафика бывают как программно-аппаратными, так и программными. Программно-аппаратные генераторы трафика обычно крайне дороги, являются закрытыми решениями, но при этом обладают специальным функционалом, ориентированным на тип сети. Что касается программных генераторов, то среди них есть достаточно большое количество предложений с открытым кодом, но позволяющих конструировать только пакетный трафик (в этом случае их иногда называют генераторами пакетов). Для ознакомления с принципами генерации трафика в лабораторной работе предлагается использовать одно из таких решений Ostinato [1].

Ostinato представляет собой кроссплатформенный open-source генератор пакетного трафика с графическим пользовательским интерфейсом, построенный на базе клиент-серверной архитектуры. Данный генератор позволяет передавать данные несколькими потоками и имеет широкие возможности для настройки поведения трафика и различных опций используемых сетевых протоколов. Генератор трафика Ostinato предустановлен на компьютере управления mnlin. Для начала работы с генератором, подключитесь к mnlin с помощью KVM-переключателя и запустите Ostinato из терминала с правами суперпользователя:

```
root@mnlin# ostinato &
```

Повышенные привилегии необходимы для получения доступа к сетевым устройствам mnlin.

Основной интерфейс программы разделен на три окна (рис.1): порты (ports list), потоки (stream list) и статистика (statistics).

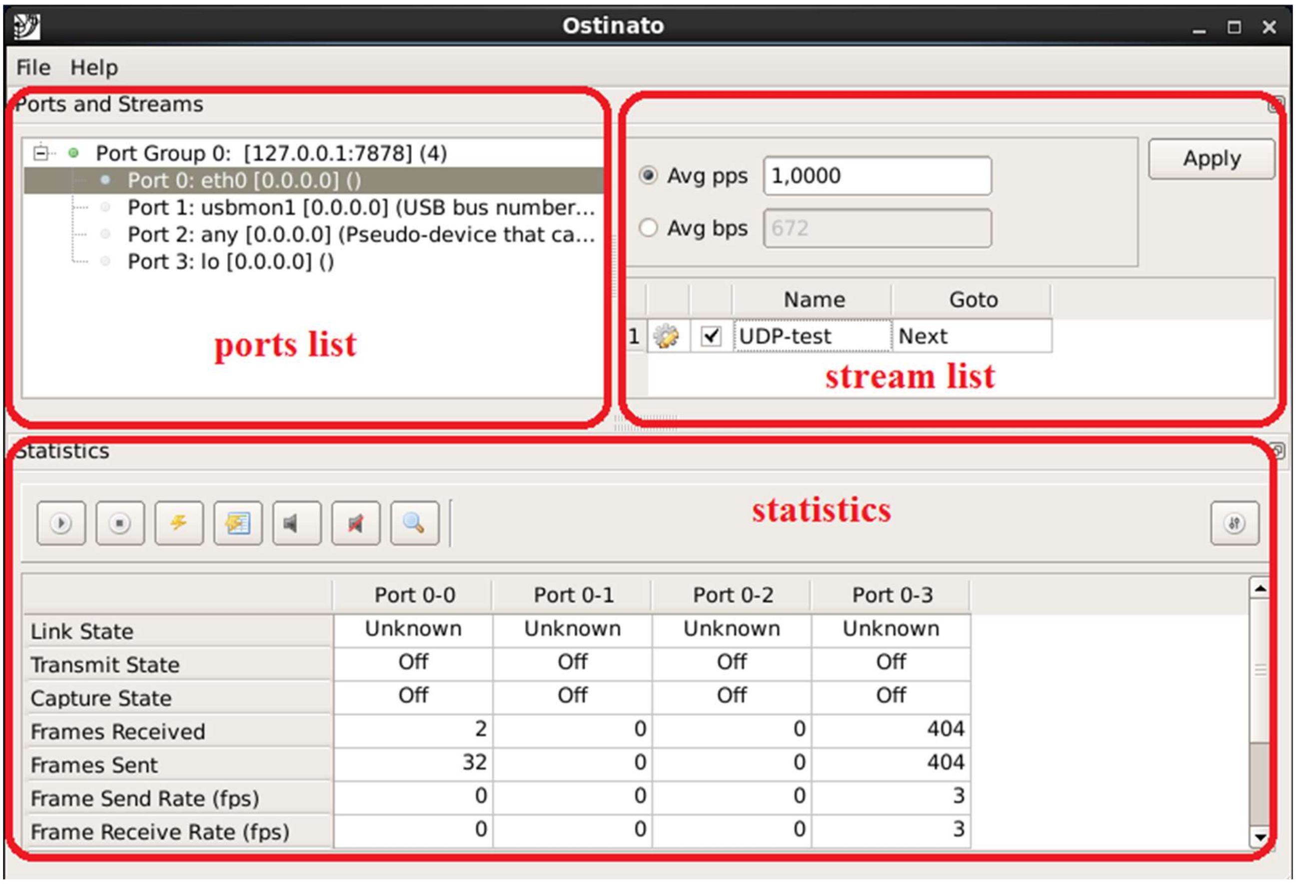


Рис. 1 – Интерфейс Ostinato

В окне портов представлена группа портов [127.0.0.1] и зеленый индикатор напротив нее, что обозначает доступность группы портов. Ostinato является клиентским приложением и может работать с несколькими различными серверами (Drone). В данном случае клиент и сервер расположены на одном компьютере и обмениваются данными через сокет 127.0.0.1:7878.

Открыв группу портов (нажав на +) можно увидеть все доступные на данной машине порты. Выберите порт, с которого будет производиться отправка пакетов в сеть. Нажатием правой кнопки мыши в окне потоков откройте меню и создайте новый поток (new stream). Дважды кликнув на значке созданного потока, вы получите доступ к конфигуратору потока.

На первой вкладке Protocol Selection (рис.2) конструируйте будущий пакет в соответствии с моделью OSI. На рисунке 2 необходимые пункты помечены красным.

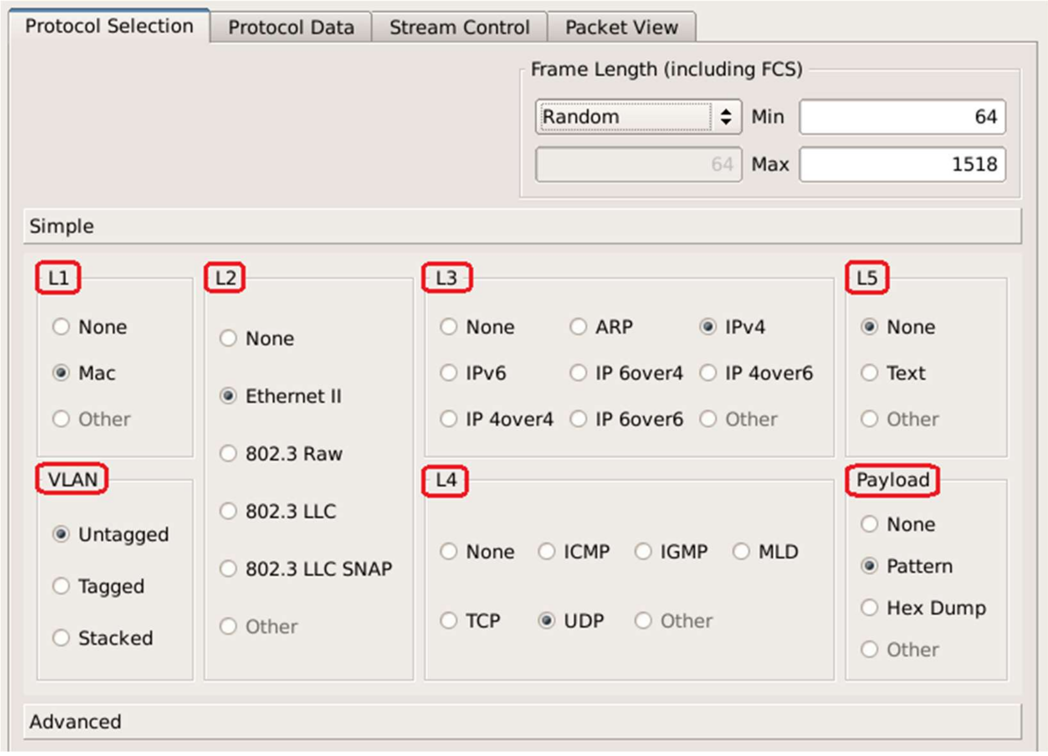


Рис. 2 – Вкладка Protocol Selection

Вкладка Protocol Data (рис. 3) позволяет настроить некоторые специфичные поля выбранных протоколов, такие как, например, адреса источника и получателя в протоколе IP, порты в UDP и т.п.

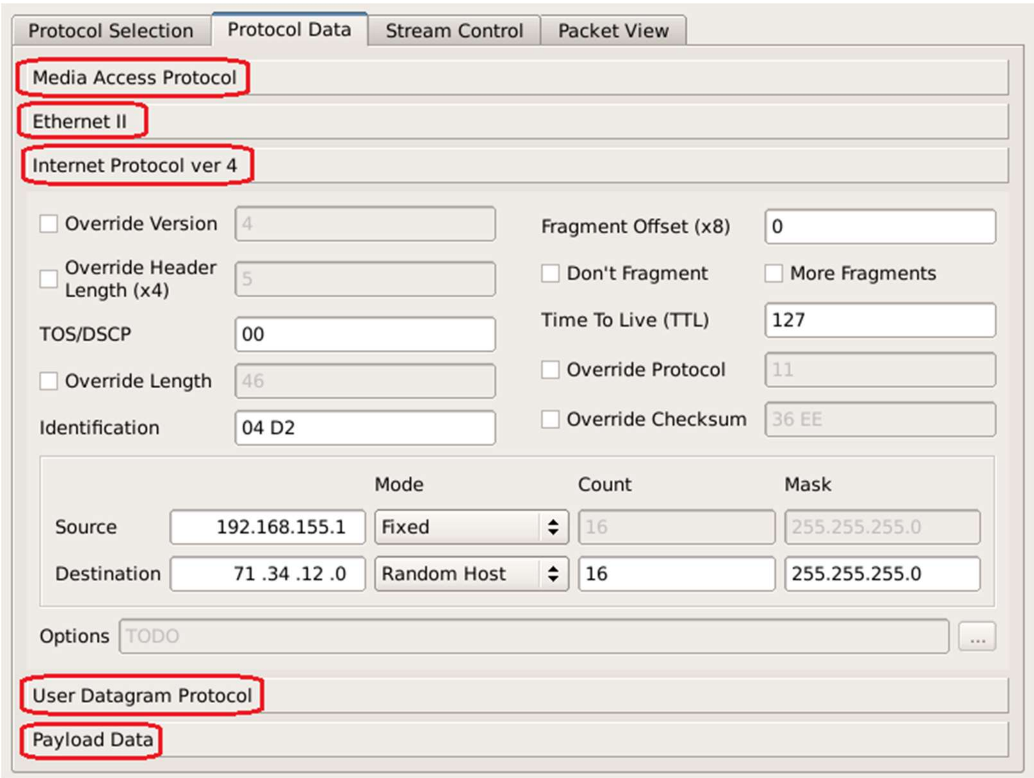


Рис. 3 – Вкладка Protocol Data

Примечание.: При конструировании пакета в реальной сети задавайте реальный MAC-адрес назначения в поле Media Access Protocol!

На вкладке Stream Control (рис. 4) выставляются опции потока. Например, можно настроить генерацию трафика пачками или отдельными пакетами, количество пакетов в секунду/битовую скорость потока, поведение потока и т.д.

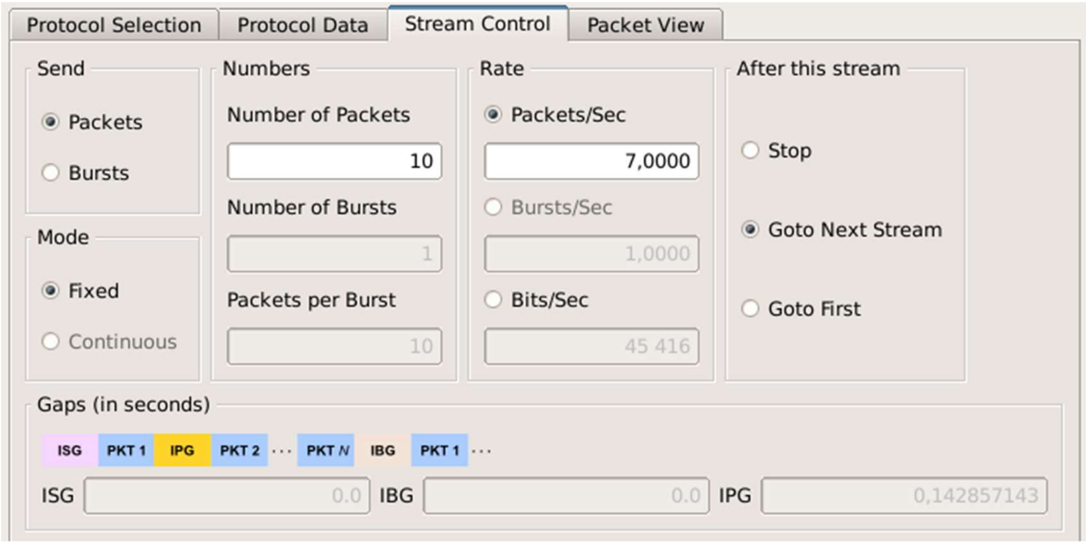


Рис. 4 – Вкладка Stream Control

Создав несколько различных потоков для данного порта и выставив для каждого из них опцию Goto Next Stream в поле After this stream, можно организовать последовательную передачу этих потоков в сеть.

Также возможно организовать одновременную смешанную передачу. Для этого необходимо в окне портов выбрать требуемый, правой кнопкой мыши открыть контекстное меню и в пункте Port Configuration изменить режим работы передачи порта на Interleaved Streams (смешанные потоки). Теперь все созданные для данного порта потоки будут передаваться на сетевой интерфейс одновременно (рис. 5).

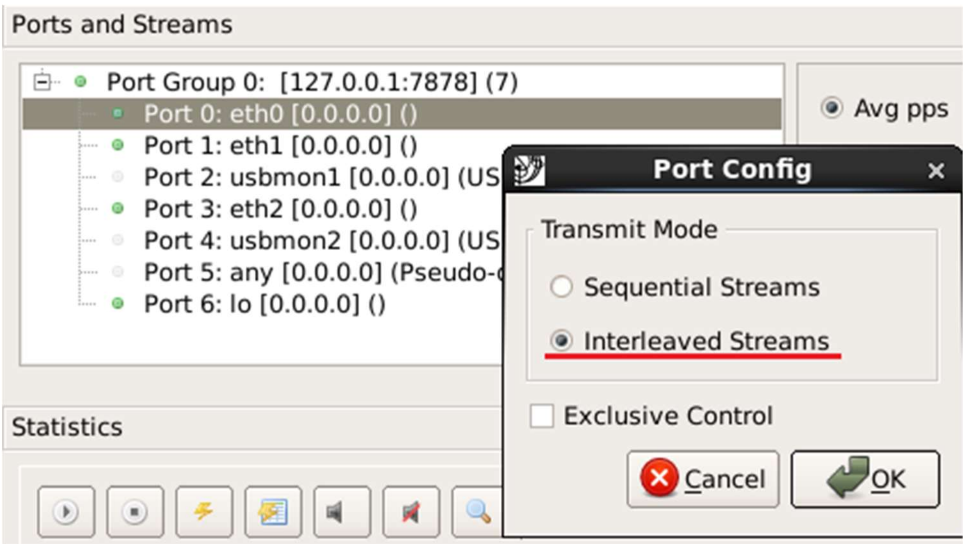


Рис. 5 – Перевод порта в смешанный режим передачи

Примечание: В смешанном режиме вместо количества передаваемых пакетов указывается скорость передачи пакетов/пачек или битовая скорость потока. Соответственно, передача трафика будет продолжаться до тех пор, пока не будет остановлена нажатием кнопки Stop Tx.

Вкладка Packet View позволяет просмотреть получившийся пакет в «собранном» виде (рис. 6).

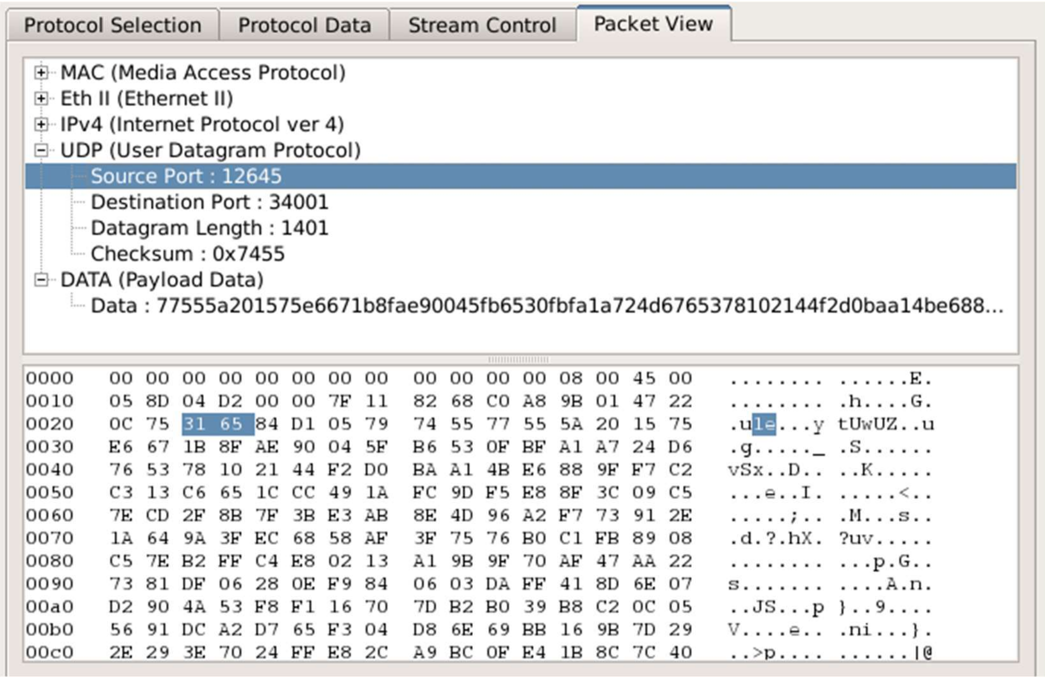


Рис. 6 – Вкладка Packet View

После окончания настройки потока нажмите Ok и передайте изменения на сервер кнопкой Apply. Если этого не сделать, генерирующий демон Drone не получит информацию о произошедших изменениях.

Для проверки работы генератора, предварительно запустите анализатор трафика Wireshark, настроив его на прослушивание того порта, на который вы передаете сгенерированный трафик. Теперь можно начинать генерацию (рис.7).

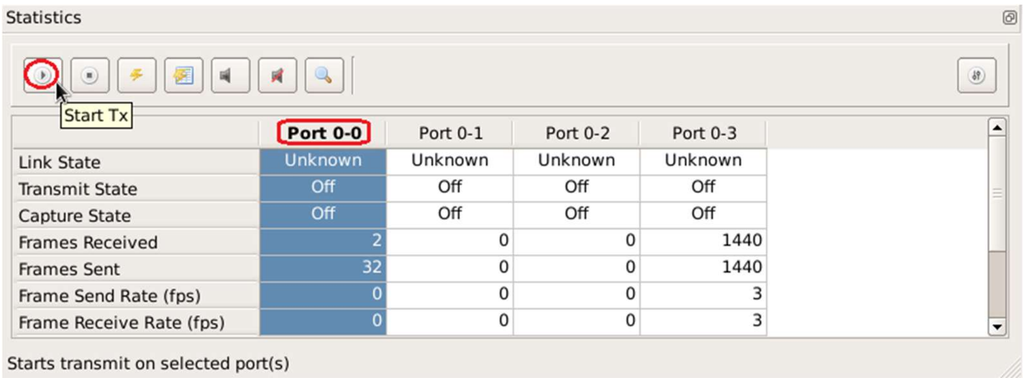


Рис. 7 – Запуск генератора

В окне статистики выберите тот же порт, для которого вы конструировали поток (например, нажмите Port 0-0, если вы настроили Port Group 0 и в ней Port 0) и кнопкой Start Tx запустите генерацию трафика.

Задание на работу

1. Создайте 3 произвольных потока данных, состоящих из пакетов различных протоколов, заданных общим количеством пакетов и пакетной скоростью потока.
2. Создайте 3 произвольных потока данных, состоящих из пакетов различных протоколов, заданных числом пачек (bursts).
3. Запустите анализатор трафика Wireshark и начните захват на том порту, на котором предполагается генерировать трафик с помощью Ostinato.
4. Организуйте последовательную передачу сконструированных потоков в произвольном порядке. Сохраните полученный трэйс-файл.
5. Организуйте смешанную передачу (interleaved) сконструированных потоков трафика. Сохраните полученный трэйс-файл.
6. Изучив содержимое полученных трэйс-файлов, убедитесь в корректной работе генератора трафика. Используйте инструменты из меню Statistics:
   a. IO Graph
   b. Packet Lenghts
   c. Protocol Hierarchy

К защите:

1. Знать основные принципы конструирования пакетов согласно модели TCP/IP, характеристики трафика пакетных сетей, процессы формирования потоков, принципы инкапсуляции.
2. Уметь использовать генератор трафика для создания потоков с различными характеристиками.
3. Представить отчет, содержащий характеристики сконструированных потоков трафика согласно п. 1 и 2 задания на лабораторную работу, примеры полученных пакетов (см. рис. 6), трэйс-файлы, статистику работы согласно меню Statistics.

Рекомендуемая литература:

1. Ostinato: Packet/Traffic Generator and Analyzer. https://code.google.com/p/ostinato/
2. Олифер Н. А., Олифер В.Г. Компьютерные сети. Принципы, технологии, протоколы. Издание 4-е. Питер, 2010.

3. Мониторинг в сетях связи: протокол ICMP, специализированные утилиты

Цель работы: овладение основными инструментами мониторинга сетей.

Краткая теоретическая справка

1. Протокол ICMP (Internet Control Message Protocol) является протоколом сообщений об ошибках. Несмотря на то, что ICMP считается протоколом сетевого уровня, его сообщения инкапсулируются в IP. Сообщения ICMP довольно разнообразны [1], но имеют единый формат (рис. 1). На этом протоколе базируются средства мониторинга сетей связи, а также сообщения о недоступности узла в некоторых протоколах прикладного уровня, например, ошибка 404 в HTTP.

Тип сообщения	Код сообщения	Контрольная сумма
Параметры		
Данные		

Тип сообщения – согласно классификации
Код сообщения – дополнительные сведения об ошибке
Параметры – например, IP-адрес узла
Данные – например, IP-заголовок и первые 64 бита пакета, переадресованного на другой узел.

Рис. 1 – Формат сообщений ICMP

Протокол ICMP для IPv4 и его сообщения описаны в RFC 792, работа с масками сетей – в RFC 950 [2]. Протокол ICMP для IPv6 описан в RFC 4443 [3].

Инструменты мониторинга утилиты *ping* и *traceroute* описаны в RFC 2529 [4].

2. Утилита *ping* (Packet Internet Groper – одно из возможных прочтений) является одним из главных средств, используемых для отладки сетей, и служит для принудительного вызова ответа конкретной машины. Она позволяет проверять работу приложений TCP/IP (по портам) на удаленных машинах, адреса устройств в локальной сети, адрес удаленного сетевого устройства. В выполнении команды *ping* участвуют система маршрутизации, схемы разрешения адресов и сетевые шлюзы. Это утилита низкого уровня, которая не требует наличия серверных процессов на зондируемой машине,

поэтому успешный результат при прохождении запроса вовсе не означает, что выполняются какие-либо сервисные программы высокого уровня, а говорит о том, что сеть находится в рабочем состоянии, питание зондируемой машины включено, и машина не отказала. Утилита *ping* входит во все реализации TCP/IP не зависимо от операционной системы.

Получив эхо-запрос *ping*, программное обеспечение, реализующее протокол IP у адресата, посылает эхо-ответ. Эхо-запросы посылаются заданное количество раз (ключ *-n*) или по умолчанию до тех пор, пока пользователь не введет команду прерывания (Ctrl+C или Del), после чего выводятся статистические данные. В некоторых реализациях количество посылок эхо-запросов ограничено, например, в Windows их 4. В некоторых случаях можно в целях обеспечения безопасности (защита от DDOS-атак) выставить запрет на эхо-ответы. Список ключей и формат команды можно посмотреть самостоятельно, набрав в командной строке ping.

На практике большинство опций в формате команды можно опустить, тогда в командной строке может быть: `ping имя_узла` или `ping IP-адрес`.

Пример:

```
C:\Users\admin>ping yandex.ru
Обмен пакетами с yandex.ru [77.88.21.11] с 32 байтами данных:
Ответ от 77.88.21.11: число байт=32 время=1651мс TTL=57
Ответ от 77.88.21.11: число байт=32 время=154мс TTL=57
Ответ от 77.88.21.11: число байт=32 время=79мс TTL=57
Ответ от 77.88.21.11: число байт=32 время=77мс TTL=57

Статистика Ping для 77.88.21.11:
 Пакетов: отправлено = 4, получено = 4, потеряно = 0
 (0% потерь)
Приблизительное время приема-передачи в мс:
 Минимальное = 77мсек, Максимальное = 1651 мсек, Среднее = 490 мсек
```

3. Утилита *traceroute* (в реализациях Windows используется написание *tracert*) позволяет выявлять последовательность шлюзов, через которые проходит IP-пакет на пути к пункту своего назначения. У этой команды есть много опций, большинство из которых применяются крайне редко. Традиционно используется формат `traceroute имя_узла`, которое может быть задано в символической или числовой форме. Выходная информация представляет собой список машин, начиная с первого шлюза и кончая пунктом назначения.

Принцип работы *traceroute* основан на установке поля времени жизни (TTL) исходящего пакета таким образом, чтобы это время истекало до достижения пакетом пункта назначения. При получении пакета с обнуленным полем TTL текущий шлюз отправит сообщение об ошибке на машину-источник. Каждое приращение поля времени жизни позволяет пакету пройти на один шлюз дальше.

Утилита *traceroute* посылает для каждого значения поля TTL три пакета. Если промежуточный шлюз распределяет трафик по нескольким маршрутам, то эти пакеты могут возвращаться разными машинами. Некоторые системы не посылают уведомлений о пакетах, время жизни которых истекло, а некоторые посылают уведомления, которые поступают обратно с задержкой, превышающей время ожидания на машине-источнике. Эти шлюзы обозначаются рядом звездочек. Если конкретный шлюз определить нельзя, все равно с помощью *traceroute* можно увидеть следующие за ним узлы маршрута. Заметим, что в связи с использованием на сетях динамической маршрутизации, в разные моменты времени можно получить различные маршруты прохождения пакетов. Это также относится к зеркалированным узлам.

Пример:

```
admin@ddd:~$ traceroute lenta.ru
traceroute to lenta.ru (81.19.85.92), 30 hops max, 60 byte packets
 1 172.24.255.254            (172.24.255.254)  13.218 ms 14.129 ms 14.019 ms
 2 84.204.14.254             (84.204.14.254)   13.848 ms 15.145 ms 15.040 ms
 3 46.47.255.33              (46.47.255.33)    13.910 ms 13.815 ms 14.594 ms
 4 mx960-spb.peterstar.net   (82.196.95.169)   15.117 ms 25.568 ms 26.261 ms
 5 ix-j-mx240.m9.ramtel.ru   (193.232.244.118) 39.993 ms 39.870 ms 39.750 ms
 6 s193-mx240.vr.rambler.ru  (81.19.64.93)     39.672 ms 17.704 ms 19.828 ms
 7 81.19.94.132              (81.19.94.132)    19.772 ms 19.708 ms 19.590 ms
 8 81.19.85.92               (81.19.85.92)     19.529 ms 19.424 ms 28.634 ms
```

Пример:

```
C:\Users\admin>tracert yandex.ru
Трассировка маршрута к yandex.ru [87.250.251.11]
с максимальным числом прыжков 30:

 1   100 ms   104 ms   106 ms HS2-1-16.xG.SPb.SkyLink.RU [89.253.1.16]
 2   119 ms   99 ms   106 ms HS2-0-1.xG.SPb.SkyLink.RU [89.253.0.1]
 3   107 ms   106 ms   106 ms 212.129.96.227
 4   112 ms   104 ms   106 ms aurora-spb-ix.yandex.net [194.226.100.90]
 5   128 ms   198 ms   110 ms 213.180.213.134
 6 *   *   * Превышен интервал ожидания для запроса.
 7   124 ms   108 ms   105 ms s650-eto2c1.yandex.net [213.180.213.65]
 8   121 ms   132 ms   133 ms l3-s550-s650.yandex.net [213.180.213.28]
 9   116 ms   106 ms   133 ms yandex.ru [87.250.251.11]

Трассировка завершена.
```

4. Существует комбинированная диагностическая утилита *mtr* (My Traceroute), сочетающая в себе функциональность рассмотренных выше *traceroute* и *ping*. Данная утилита основана на библиотеке *libncurses* (консольная версия) или на базе GTK+ (оконная версия), позволяет в реальном времени отслеживать маршрут до заданного узла и изменяющееся время ответа каждого из промежуточных узлов, а также процент потерянных пакетов. Консольный вывод утилиты *mtr* представлен на рисунке 2. На данный момент *mtr* включена практически во все дистрибутивы Linux.

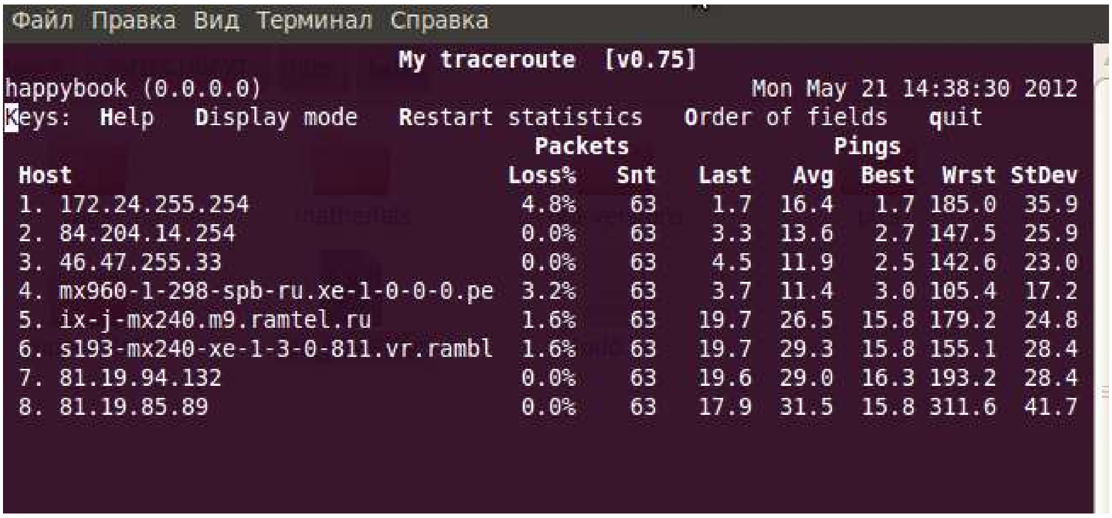


Рис.2 – Пример работы утилиты *mtr* в консольном режиме

5. Утилита *netstat* выводит информацию о локальной сети и средствах TCP/IP. Она реализована непосредственно в операционной системе и занимается сбором статистики об ошибках, текущих соединениях, состоянии портов и соединений. Содержание и форма выходной информации зависят от операционной системы, но обычно выводятся следующие данные: список соединений, статистика сетевых интерфейсов, статистика по буферам данных, содержание таблицы маршрутизации, статистика работы протоколов. Характер выводимой информации можно выбирать с помощью опций командной строки. Рассмотрим основные возможности мониторинга с помощью утилиты *netstat.*

5.1. Список соединений

Утилита *netstat* обладает набором ключей для отображения портов, находящихся в активном и/или пассивном состоянии. Таким образом, можно получить список всех серверных приложений, работающих на данном компьютере. Отметим, что формат списка соединений для сервера с системой NAT и для клиентской машины будет разным.

Информация выводится столбцами. В первом из них указан протокол, затем размеры очередей приема и передачи для установленного соединения на данной машине (на другом конце соединения размеры очередей могут быть другими), локальный и удаленный адреса и текущее состояние соединения.

Пример:

```
admin@ddd:~$ netstat -ta
Активные соединения с интернетом (servers and established)
Proto Recv-Q Send-Q Local Address Foreign Address State
tcp   0 0 *:29011               *:*                     LISTEN
tcp   0 0 localhost:ipp   *:*                           LISTEN
tcp   0 0 172.24.0.157:35608    213.199.179.141:40041   ESTABLISHED
tcp   0 0 172.24.0.157:37760    163-247.static.quie:www TIME_WAIT
tcp   0 0 172.24.0.157:36441    95-28-49-237.broa:26647 ESTABLISHED
tcp   0 0 172.24.0.157:38541    172.24.0.170:55554      TIME_WAIT
tcp   0 0 172.24.0.157:54369    bos-w031b-rdr1.bl:https ESTABLISHED
tcp   0 0 172.24.0.157:38543    172.24.0.170:55554      TIME_WAIT
tcp   0 0 172.24.0.157:55651    broadband-95-84-1:17654 ESTABLISHED
tcp   0 0 172.24.0.157:43546    178.204.199.167:26639   ESTABLISHED
tcp   0 0 172.24.0.157:44763    agama.yande:xmpp-client ESTABLISHED
tcp   0 0 172.24.0.157:43715    chat-p01c-rdr1.bl:https ESTABLISHED
tcp   0 0 172.24.0.157:53870    broadband-109-173:20143 ESTABLISHED
tcp   0 0 172.24.0.157:43067    h178-129-218-223.d:8740 ESTABLISHED
tcp   0 0 172.24.0.157:53809    89.189.134.198.dy:22113 ESTABLISHED
tcp   0 0 172.24.0.157:44762    agama.yande:xmpp-client ESTABLISHED
tcp   0 0 172.24.0.157:53246    212.8.166.36:https      ESTABLISHED
tcp   0 0 172.24.0.157:55257    bart-w04b.blue.ic:https ESTABLISHED
tcp   0 0 172.24.0.157:33021    dialin.customers.:26770 ESTABLISHED
tcp   0 0 172.24.0.157:58275    140.222.81.95.chtt:4078 ESTABLISHED
tcp   0 0 172.24.0.157:46402    91.190.216.24:12350     ESTABLISHED
tcp6   0 0 localhost:ipp        [::]:*                  LISTEN
```

Состояние соединения имеет значение только для протокола TCP. Протокол UDP факта установления соединения не проверяет.

5.2. Содержание таблицы маршрутизации

Каждое соединение машины с сетью называется сетевым интерфейсом. Машина, имеющая более одного интерфейса, может принимать данные по одному интерфейсу и передавать их по другому, осуществляя пересылку данных между сетями. Эта функция называется маршрутизацией, а машина, выполняющая ее – шлюзом.

Данные маршрутизации хранятся в так называемых таблицах маршрутизации, которые могут быть статическими и динамическими в зависимости от уровня сети и протокола маршрутизации. Для направления пакета по конкретному адресу подбирается наилучший маршрут согласно метрике. Если такой маршрут отсутствует, и нет маршрута по умолчанию, то отправителю возвращается сообщение об ошибке.

Утилита `netstat -r` позволяет отображать таблицу маршрутизации.

Пункты назначения и шлюзы могут показываться в виде имен машин или в виде их IP-адресов. Флаги дают оценку маршрута.

Пример:

```
admin@ddd:~ > netstat -r
Kernel IP routing table
Destination        Gateway            Genmask           Flags Ifac
ddd.sut.ru         *                  255.255.255.255   UH    eth1
195.19.219.120     *                  255.255.255.248   U     eth0
195.19.219.128     *                  255.255.255.192   U     eth1
192.168.1.0        *                  255.255.255.0     U     eth0
195.19.221.0            lgw.ccs.sut.ru      255.255.255.0     UG    eth1
193.125.0.0        lgw.ccs.sut.ru     255.255.0.0       UG    eth1
loopback           *                  255.0.0.0         U     lo
default                 lgw.ccs.sut.ru      0.0.0.0           UG    eth1
```

5.3. Статистика сетевых интерфейсов

При использовании ключа *-e* на экран будут выведены статистические данные всех используемых Ethernet-интерфейсов. Исходя из них, можно выяснить, исправно ли соединение с сетью.

Пример:

```
admin@ddd:~ > netstat -e
Kernel Interface table
Iface MTU  Met  RX-OK  RX-ERR RX-DRP RX-OVR  TX-OK   TX-ERR TX-DRP TX-OVR Flg
eth0  1000 0    844904  0        17     0      1454454  5        0      0     BRU
eth1  1500 0    590844  0        7      0      434438   59       0      0     BRU
lo    3924 0    45754   0        0      0      45754    0        0      0     LRU
```

Ошибки являются следствием проблем в кабельной системе или следствием неисправности платы сетевого адаптера. В нормально работающей сети количество конфликтов (RX-OVR, TX-OVR) не должно превышать 3% от числа пакетов, а другие ошибки не должны составлять более 0,5% от общего числа пакетов.

5.4. Статистика передачи данных

Использование *netstat -s* позволяет вывести содержимое счетчиков сетевых программ. В выходной информации есть разделы, относящиеся к различным протоколам: IP, ICMP, TCP, UDP. С ее помощью можно определить место появления ошибки в принятом пакете.

Пример:

```
admin@ddd:~$ netstat -s
Ip:
    всего пакетов принято 17058
    2 с неверными адресами
    0 перенаправлено
    0 входящих пакетов отклонено
    входящих пакетов доставлено: 4364
    запросов отправлено: 3936
Icmp:
    ICMP сообщений получено: 306
    неудачных входящих ICMP сообщений: 0
```

```
    Гистограмма входа ICMP
        пункт назначения недоступен: 8
        потери при прохождении: 263
        эхо-ответы: 35
    послано сообщений ICMP: 280
    неудачные сообщения ICMP: 0
    Гистограмма выхода ICMP
        пункт назначения недоступен: 8
        эхо-запросов: 14
IcmpMsg:
        InType0: 35
        InType3: 8
        InType11: 263
        OutType3: 8
        OutType8: 14
        OutType69: 258
Tcp:
    открытия активных соединений: 148
    открытия пассивных соединений: 0
    неудачные попытки соединения: 4
    получено сбросов соединений: 2
    соединений установлено: 0
    сегментов получено: 3386
    отправлено сегментов: 2895
    повторно передано сегментов: 39
    плохих сегментов получено: 0
    сбросов послано: 8
Udp:
    пакетов принято: 661
    принято пакетов на неизвестный порт: 8
    ошибок приема пакетов: 0
    пакетов послано: 723
UdpLite:
TcpExt:
    пакеты, вырезанные из очереди приема по причине переполнения буфера
сокета: 1
    67 TCP sockets finished time wait in fast timer
    задержанных подтверждений послано: 119
    Редим быстрого подтверждения приема был активирован 69 раз
    11 packets directly queued to recvmsg prequeue.
    3635 bytes directly received in process context from prequeue
    ожидаемых заголовков пакетов: 1745
    ожидаемых заголовков пакетов, непосредственно стоявших в очереди к
пользователю: 5
    311 acknowledgments not containing data payload received
    ожидаемые подтверждения: 299
    9 congestion windows recovered without slow start after partial ack
    1 timeouts in loss state
    19 retransmits in slow start
    других TCP тайм-аутов: 19
    22 packets collapsed in receive queue due to low socket buffer
    получено DSACKs: 9
    1 соединения сброшены из-за неожиданных данных
    2 connections reset due to early user close
    TCPDSACKIgnoredNoUndo: 3
```

```
    TCPSackShiftFallback: 9
IpExt:
    InMcastPkts: 15
    OutMcastPkts: 19
    InOctets: 4353792
    OutOctets: 405434
    InMcastOctets: 2714
    OutMcastOctets: 2990
```

Изучаемые в процессе выполнения лабораторной работы средства мониторинга относятся к универсальным в сетях IP. Они являются встроенными во все операционные системы с поддержкой IP, работают как с IPv4, так и с IPv6. Для своей работы утилита *netstat* использует статистику, собранную при помощи ICMP. Так как ICMP для IPv4 и IPv6 имеет отличия, связанные непосредственно с изменением формата заголовка, то и результат для этих протоколов будет отличаться. Современные операционные системы поддерживают обе версии ICMP.

Задание на работу:

1. Провести трассировку трех узлов по заданию преподавателя. По результатам построить графики зависимости времени прохождения пакета от номера узла. Указать шлюзы перехода из одной сети в другую. Листинги трассировки привести в отчете.
2. Провести оценку работоспособности узлов: узлов в подсети лаборатории, шлюза подсети, 5 узлов из ранее сделанных трассировок. Оценить TTL для каждого из них.
3. Запустить несколько сетевых приложений на клиентской машине (например, несколько сайтов, интернет-мессенджер и т.п.). Снять с клиентской машины при помощи утилиты *netstat* таблицу маршрутизации, список соединений, статистику передачи данных, состояние интерфейса Ethernet. На основании списка соединений построить карту сети (см. лаб.1). На основе таблицы маршрутизации зарисовать архитектуру сети.

К защите:

1. Знать основные принципы мониторинга сетей, характеристики сетей (TTL, время приема-передачи и т.п.), принципы работы средств мониторинга (всех используемых в лабораторной работе утилит).
2. Уметь использовать средства мониторинга IP-сетей.
3. Представить отчет, содержащий листинги работоспособности узлов, результаты трассировки, графики зависимости времени прохождения пакета

от номера узла, статистику работы сети согласно netstat, карту сети, архитектуру сети на основе таблицы маршрутизации.

Рекомендуемая литература:

1. RFC 792 Internet control message protocol. September, 1981.
2. RFC 950 Internet Standard Subnetting Procedure. August 1985
3. RFC 4443 Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification. March, 2006
4. RFC 2925 Definitions of Managed Objects for Remote Ping, Traceroute, and Lookup Operations. September, 2000.
5. Олифер Н. А., Олифер В.Г. Компьютерные сети. Принципы, технологии, протоколы. Издание 4-е. Питер, 2010.

4. Статическая маршрутизация в сетях IPv4 и IPv6

Цель работы: получение навыков конфигурации локальной сети на основе протоколов IPv4 и IPv6 с использованием статической маршрутизации.

Краткая теоретическая справка

Базовое иерархическое построение сети предполагает наличие максимум трех уровней иерархии: ядро сети, уровень агрегации и уровень доступа (рисунок 1). В ядре как минимум находятся опорные маршрутизаторы, могут быть расположены граничные маршрутизаторы и серверы услуг.

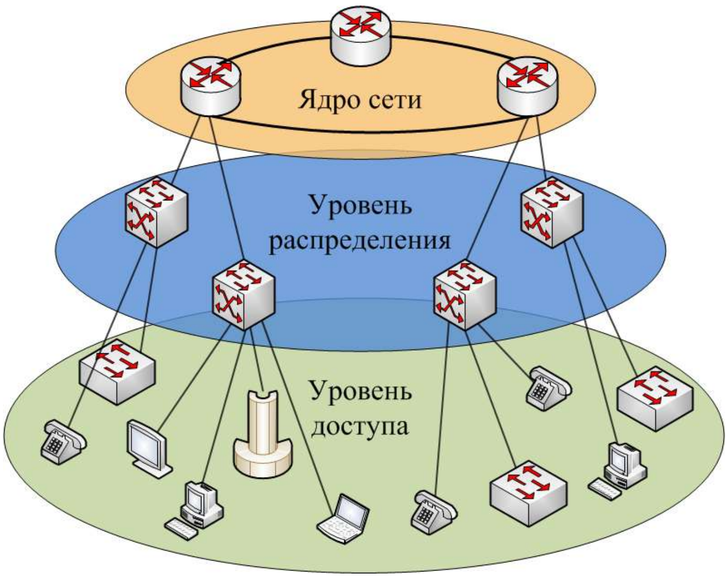


Рис. 1 – Трехуровневая базовая модель построения сети

Уровень агрегации обеспечивает возможность агрегации и распределения трафика внутри сети оператора, может выполнять роль интегратора отдельных сегментов сети. Уровень доступа организует возможность физического доступа пользователя к сети оператора. Подобный подход не зависит от используемой оператором технологии, но позволяет эффективно выстроить архитектуру, повысив устойчивость сети, и сделать процессы управления сетью прозрачными.

Отметим, что в небольших сетях может происходить вырождение иерархии до глубины в два уровня: доступ и ядро. Ядро небольшой сети представляет собой один маршрутизатор, выполняющий функции шлюза и, иногда, DNS-сервера. В этом случае чаще всего используется статическая маршрутизация, а управление потоками заключается в задании маршрутного правила.

В обобщенном виде запись маршрутного правила (далее маршрута) можно представить так:

```
Route network netmask gateway
```

Например, конкретная запись может быть представлена как:

```
Route 12.5.7.0 255.255.255.0 78.3.65.1,
```

Где 12.5.7.0 – это адрес подсети (network), 255.255.255.0 – маска данной подсети (netmask), а 78.3.65.1 – адрес шлюза (gateway). Шлюз представляет собой маршрутизатор, на который посылается весь трафик, удовлетворяющий данному маршруту, т.е. имеющий адрес получателя пакетов входящий в указанную подсеть.

Отметим, что существуют также многие другие способы маршрутизации пакетов, учитывающие различные параметры трафика (policy routing), адаптирующиеся под изменяющуюся топологию сети и т.д., однако они используются в крупных сетях с динамической маршрутизацией и будут рассмотрены позднее.

Задание на работу:

Часть 1

1. Соберите тестовую схему сети согласно рис. 1.
2. Подготовьте программные маршрутизаторы, войдите под суперпользователем и установите соединение по ssh.
3. Сконфигурируйте программные маршрутизаторы согласно заданию (табл. 1). Проверьте взаимную доступность программных маршрутизаторов.
4. Сконфигурируйте маршрутизатор Cisco. Проверьте взаимную доступность подсетей.
5. Снимите ARP-таблицы маршрутизаторов.

Часть 2

6. Соберите тестовую схему сети согласно рис. 6.
7. Выполните действия согласно п. 2-4 части 1, но только для IPv6.
8. Снимите таблицу канального уровня ND-протокола.

Для успешного выполнения лабораторной работы ознакомьтесь с методикой выполнения работы на конкретном примере.

Методика выполнения работы

Часть 1. Статическая маршрутизация на базе протокола IPv4

Ход выполнения работы проиллюстрирован на примере настройки маршрутизации в тестовой сети, приведенной на рисунке 2.

Для успешного выполнения лабораторной работы необходимо подготовить программные маршрутизаторы. На каждом программном маршрутизаторе следует запустить пакет маршрутизации Quagga. Применяемая в Quagga система команд очень близка к системе команд Cisco.

Включите программные маршрутизаторы и с помощью KVM-переключателя подсоединитесь к каждому из них, войдите под суперпользователем:

```
login: root
passw: simulator
```

и запустите Quagga:

```
root@soft-core# service zebra start
```

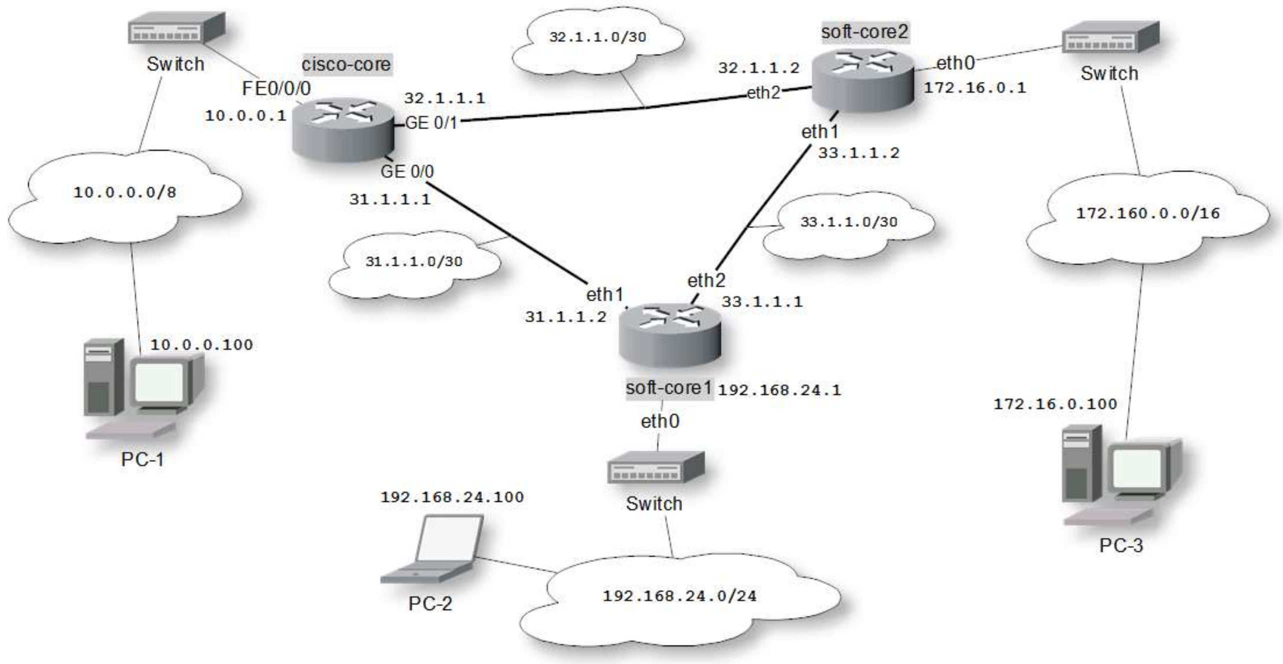


Рис. 2 – Схема тестовой сети IPv4

Примечание: Обратите внимание, что при запуске пакета маршрутизации Quagga, он перехватывает управление сетевыми ресурсами и заменяет существующую сетевую конфигурацию своей, хранящейся в файле /etc/quagga/zebra.conf. По умолчанию в данной конфигурации на soft-core1 (программном маршрутизаторе ядра1 1) настроен интерфейс eth0=192.168.24.1/24, а на soft-core2 (программном маршрутизаторе ядра 2) eth0=172.16.0.1/16.

Теперь на программные маршрутизаторы можно зайти посредством протокола SSH из тех локальных сетей, шлюзами которых они являются, т.е. на soft-core1 через интерфейс eth0 с адресом 192.168.24.1, а на soft-core2 – через eth0 с адресом 172.16.0.1.

Примечание: Для того, чтобы осуществить удаленное соединение по протоколу ssh, на машинах с ОС Linux необходимо запустить терминал и набрать команду:

```
ssh логин@IP-адрес_удаленного_хоста
```

после чего ввести запрашиваемый пароль. Логин и пароль на программных маршрутизаторах стенда по умолчанию:

login: admin

passw: admin

Важно: необходимо очистить таблицу правил фаейрвола на каждом программном маршрутизаторе. Это действие должно выполняться от суперпользователя:

```
pc2$ ssh admin@192.168.24.1
admin@192.168.24.1's password:
[admin@localhost ~]$ su
[root@localhost ~]$ iptables -F
[root@localhost ~]$ ip6tables -F
```

Для получения доступа к консоли управления Quagga, необходимо, установив удаленное подключение по ssh, набрать следующую команду:

```
telnet localhost 2601
```

Таким образом, мы подключаемся к процессу, ожидающему соединения на порту 2601 программного маршрутизатора (маршрутизирующий демон zebra).

Примечение: Пароль при подключении к консоли Quagga по умолчанию: softcore.

Процесс конфигурирования программного маршрутизатора может выглядеть следующим образом:

```
pc2$ ssh admin@192.168.24.1
admin@192.168.24.1's password:
[admin@localhost ~]$ telnet localhost 2601
soft-core1.lab> enable
soft-core1.lab# configure terminal
soft-core1.lab(config)# ip forwarding /включаем маршрутизацию IP
soft-core1.lab(config)# interface eth1 /настраиваем интерфейс eth1
soft-core1.lab(config-if)# ip address 31.1.1.2/30      /присваиваем адрес
IPv4
soft-core1.lab(config-if)# description to Cisco       /добавляем
описание
soft-core1.lab(config-if)# no shutdown     /включаем интерфейс
soft-core1.lab(config-if)# exit/выходим из режима конфигурирования
интерфейса
soft-core1.lab(config)# interface eth2
soft-core1.lab(config-if)# ip address 33.1.1.1/30
soft-core1.lab(config-if)# description to SC-2
soft-core1.lab(config-if)# no shutdown
soft-core1.lab(config-if)# exit
```

```
/Добавляем статические маршруты в формате "dst_network/netmask next-hop"
soft-core1.lab(config)# ip route 10.0.0.0/8 31.1.1.1
soft-core1.lab(config)# ip route 172.16.0.0/16 33.1.1.2

/Просмотреть таблицу маршрутизации можно следующим образом
soft-core1.lab# show ip route
Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF,
       I - ISIS, B - BGP, > - selected route, * - FIB route

S>* 10.0.0.0/8 [1/0] via 31.1.1.1, eth1
C>* 31.1.1.0/30 is directly connected, eth1
C>* 33.1.1.0/30 is directly connected, eth2
C>* 127.0.0.0/8 is directly connected, lo
S>* 172.16.0.0/16 [1/0] via 33.1.1.2, eth2
C>* 192.168.24.0/24 is directly connected, eth0
```

Подобную настройку необходимо произвести для обоих программных маршрутизаторов в соответствии с полученным заданием.

Далее необходимо убедиться, что программные маршрутизаторы «видят» друг друга. Для этого запустите еще один сеанс удаленного доступа по ssh и утилитами ping и traceroute проверьте доступность подключенного порта другого маршрутизатора и путь прохождения пакетов до него.

```
pc2$ ssh admin@192.168.24.1
admin@192.168.24.1's password:
Last login: Thu Jul 19 20:18:53 2012 from 192.168.24.100
[admin@localhost ~]$ ping 33.1.1.2
PING 33.1.1.2 (33.1.1.2) 56(84) bytes of data.
64 bytes from 33.1.1.2: icmp_seq=1 ttl=64 time=0.323 ms
64 bytes from 33.1.1.2: icmp_seq=2 ttl=64 time=0.095 ms
64 bytes from 33.1.1.2: icmp_seq=3 ttl=64 time=0.110 ms

--- 33.1.1.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.09/0.16/0.30/0.10 ms
[admin@localhost ~]$ traceroute 33.1.1.2
traceroute to 33.1.1.2 (33.1.1.2), 30 hops max, 60 byte packets
 1  33.1.1.2 (33.1.1.2)  0.262 ms  0.074 ms  0.058 ms
```

Сконфигурируем маршрутизатор Cisco. С компьютера управления зайдем на консольный порт Cisco:

```
picocom /dev/ttyS0
```

и включим маршрутизатор. После загрузки на экране вы увидите приглашение, теперь можно начинать настройку.

```
cisco-core>enable                     /переходим в режим exec
cisco-core#configure terminal
cisco-core(config)#ip classless       /включаем CIDR
cisco-core(config)#ip routing         /включаем IP-маршрутизацию

/Настраиваем сетевые интерфейсы
cisco-core(config)#interface gigabitEthernet 0/0 /выбираем интерфейс для
настройки
cisco-core(config-if)#ip address 31.1.1.1 255.255.255.252    /присваиваем
интерфейсу IP-адрес и маску подсети
cisco-core(config-if)#description to soft-core1 /добавляем описание
интерфейса (необязательно)
cisco-core(config-if)#no shutdown     /включаем интерфейс
```

```
cisco-core(config-if)#exit           /выходим из режима конфигурирования
данного интерфейса
cisco-core(config)#interface gigabitEthernet 0/1
cisco-core(config-if)#ip address 32.1.1.1 255.255.255.252
cisco-core(config-if)#description to soft-core2
cisco-core(config-if)#no shutdown
cisco-core(config-if)#end      /выходим из режима конфигурации

/Создаем статические маршруты
cisco-core#configure terminal
/В маршруте нужно указать в качестве шлюза next-hop маршрутизатор
cisco-core(config)#ip route 192.168.24.0 255.255.255.0 31.1.1.2
cisco-core(config)#ip route 172.16.0.0 255.255.0.0 32.1.1.2
cisco-core(config)#exit

/Просмотрим состояние интерфейсов и маршрутов
cisco-core#show ip interface brief
cisco-core#show ip route

/Проверим доступность LAN-интерфейсов шлюзов
cisco-core#ping 192.168.24.1
cisco-core#ping 172.16.0.1
```

Если интерфейсы доступны, значит маршрутизация настроена верно.

После настройки маршрутизаторов ядра нужно проверить взаимную доступность локальных подсетей. Для этого с различных машин, находящихся в разных подсетях произведите несколько проверок доступности подсетей.

```
pc1#ping 192.168.24.100
pc1#ping 172.16.0.100
pc2#ping 10.0.0.100
pc2#ping 172.16.0.100
pc3#ping 192.168.24.100
pc3#ping 10.10.0.100
```

Если *ping* проходит успешно, утилитой *traceroute* отследите пути продвижения пакетов по сети (рис.3).

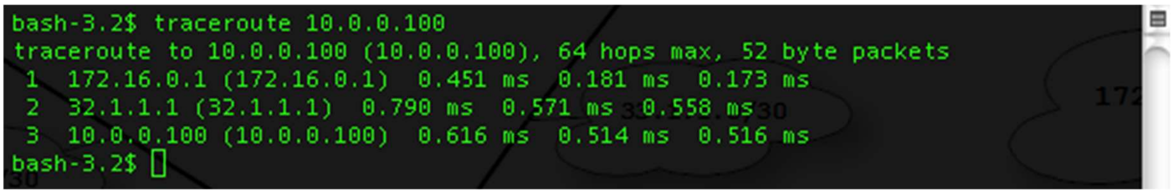


Рис. 3 – Прохождение пакетов через сеть.

Снимите ARP-таблицы с маршрутизаторов сети. Для того чтобы просмотреть ARP-таблицу программного маршрутизатора, подключитесь к нему по SSH и введите команду *ip neighbor* (рис.4).

```
[admin@localhost ~]$ ip neighbor
33.1.1.2 dev eth2 lladdr 00:1b:21:cc:0f:4e DELAY
31.1.1.1 dev eth1 lladdr 64:00:f1:19:6a:60 STALE
[admin@localhost ~]$ ping -c1 33.1.1.2
PING 33.1.1.2 (33.1.1.2) 56(84) bytes of data.
64 bytes from 33.1.1.2: icmp_seq=1 ttl=64 time=0.093 ms

--- 33.1.1.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.093/0.093/0.093/0.000 ms
[admin@localhost ~]$ ip neighbor
33.1.1.2 dev eth2 lladdr 00:1b:21:cc:0f:4e REACHABLE
31.1.1.1 dev eth1 lladdr 64:00:f1:19:6a:60 STALE
[admin@localhost ~]$
```

Рис. 4 – ARP-таблица программного маршрутизатора

Обратите внимание на то, как изменяется статус записи об узле 33.1.1.2 после проверки связи с ним. Просмотр ARP-таблицы в маршрутизаторе Cisco осуществляется командой *show arp* (рис. 5).

```
cisco-core#show arp
Protocol  Address          Age (min)  Hardware Addr   Type   Interface
Internet  10.0.0.1                -   6400.f119.6a60  ARPA   Vlan2
Internet  10.0.0.100              6   001d.0fc0.bcb3  ARPA   Vlan2
Internet  31.1.1.1                -   6400.f119.6a60  ARPA   GigabitEthernet0/0
Internet  31.1.1.2               15   001b.21cc.10fb  ARPA   GigabitEthernet0/0
Internet  32.1.1.1                -   6400.f119.6a61  ARPA   GigabitEthernet0/1
Internet  32.1.1.2                0   001b.21cc.10d9  ARPA   GigabitEthernet0/1
cisco-core#
```

Рис. 5 – ARP-таблица маршрутизатора Cisco

Часть 2. Статическая маршрутизация на базе протокола IPv6

Эта часть лабораторной работы аналогична рассмотренной в первой части, за исключением сетевого протокола: в данном случае сеть построена на IPv6. Схема тестовой сети приведена на рисунке 6.

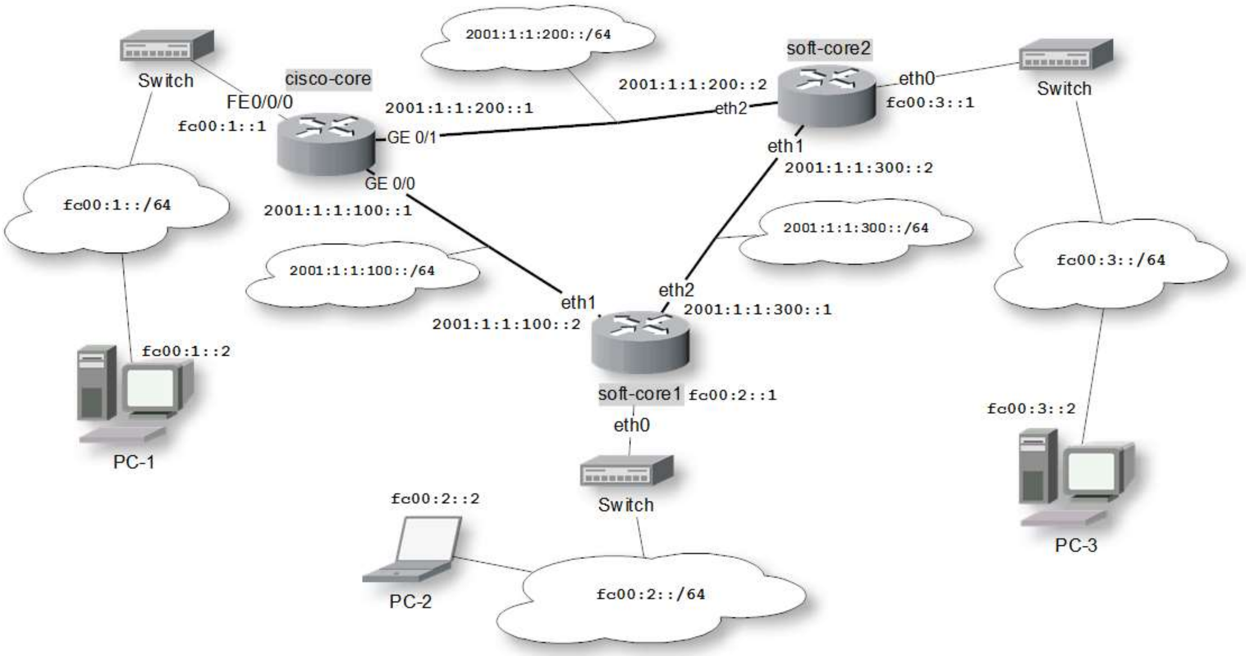


Рис. 6 – Схема тестовой сети IPv6

Настройка программных маршрутизаторов для работы в сети с IPv6-адресацией осуществляется аналогично проведенной для сетей IPv4.

1. Осуществляем необходимые подключения.

2. Конфигурируем программные маршрутизаторы. Обратите внимание на то, что интерфейсам eth0 по умолчанию присвоены только IPv4-адреса!

```
pc2$ ssh admin@192.168.24.1
admin@192.168.24.1's password:
[admin@localhost ~]$ telnet localhost 2601
soft-core1.lab> enable
soft-core1.lab# configure terminal
soft-core1.lab(config)# ipv6 forwarding
soft-core1.lab(config)# interface eth0
soft-core1.lab(config-if)# ipv6 address fc00:2::1/64
soft-core1.lab(config-if)# exit
soft-core1.lab(config)# interface eth1
soft-core1.lab(config-if)# ipv6 address 2001:1:1:100::2/64
soft-core1.lab(config-if)# exit
soft-core1.lab(config)# interface eth2
soft-core1.lab(config-if)# ipv6 address 2001:1:1:300::1/64
soft-core1.lab(config-if)# exit
soft-core1.lab(config)# ipv6 route fc00:1::/64 2001:1:1:100::1
soft-core1.lab(config)# ipv6 route fc00:3::/64 2001:1:1:300::2
soft-core1.lab(config)# exit
```

Данную настройку необходимо произвести для обоих программных маршрутизаторов в соответствии с заданием.

3. Проверяем взаимную доступность программных маршрутизаторов (рис.7).

```
TonyMac:~ tony$ ssh admin@fc00:2::1
admin@fc00:2::1's password:
Last login: Thu Jul 26 18:55:49 2012 from fc00:2::2
[admin@localhost ~]$ ping6 -c2 2001:1:1:300::2
PING 2001:1:1:300::2(2001:1:1:300::2) 56 data bytes
64 bytes from 2001:1:1:300::2: icmp_seq=1 ttl=64 time=0.174 ms
64 bytes from 2001:1:1:300::2: icmp_seq=2 ttl=64 time=0.117 ms

--- 2001:1:1:300::2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.117/0.145/0.174/0.030 ms
[admin@localhost ~]$ traceroute6 2001:1:1:300::2
traceroute to 2001:1:1:300::2 (2001:1:1:300::2), 30 hops max, 80 byte packets
 1  2001:1:1:300::2 (2001:1:1:300::2)  0.165 ms  0.121 ms  0.096 ms
[admin@localhost ~]$
```

```
TonyMac:~ tony$ ssh admin@fc00:3::1
admin@fc00:3::1's password:
Last login: Thu Jul 26 18:58:21 2012 from fc00:2::2
[admin@localhost ~]$ ping6 -c2 2001:1:1:300::1
PING 2001:1:1:300::1(2001:1:1:300::1) 56 data bytes
64 bytes from 2001:1:1:300::1: icmp_seq=1 ttl=64 time=0.162 ms
64 bytes from 2001:1:1:300::1: icmp_seq=2 ttl=64 time=0.089 ms

--- 2001:1:1:300::1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.089/0.125/0.162/0.038 ms
[admin@localhost ~]$ traceroute6 2001:1:1:300::1
traceroute to 2001:1:1:300::1 (2001:1:1:300::1), 30 hops max, 80 byte packets
 1  2001:1:1:300::1 (2001:1:1:300::1)  0.099 ms  0.065 ms  0.045 ms
[admin@localhost ~]$
```

Рис. 7 – Проверка взаимной доступности

4. Настраиваем маршрутизатор Cisco.

Примечание: Для чистоты эксперимента можно перезагрузить маршрутизатор Cisco, включив и выключив его кнопкой питания (при этом все сделанные в первой части работы изменения сотрутся!)

После загрузки маршрутизатора вы увидите на экране приглашение, и можете начинать настройку.

```
cisco-core>enable                          /переходим в режим exec
cisco-core#configure terminal              /конфигурирование из терминала
cisco-core(config)#ipv6 unicast-routing    /включаем маршрутизацию IPv6
cisco-core(config)#ipv6 cef                /включаем поддержку Cisco Express
Forwarding для протокола IPv6

/Настраиваем сетевые интерфейсы
cisco-core(config)#interface vlan 2 /LAN-интерфейс нашего маршрутизатора
cisco-core(config-if)#ipv6 address fc00:1::1/64/Задаем IPv6 адрес типа Unique
Local интерфейсу LAN
cisco-core(config-if)#no shutdown          /включаем интерфейс
cisco-core(config-if)#exit
cisco-core(config)#interface gigabitEthernet 0/0     /настраиваем интерфейс
GE0/0
cisco-core(config-if)#ipv6 address 2001:1:1:100::1/64 /Задаем IPv6 адрес типа
Global Unicast интерфейсу GE0/0
cisco-core(config-if)#no shutdown
cisco-core(config-if)#exit
cisco-core(config)#interface gigabitEthernet 0/1
cisco-core(config-if)#ipv6 address 2001:1:1:200::1/64
```

```
cisco-core(config-if)#no shutdown
cisco-core(config-if)#^Z

/Посмотрим состояние интерфейсов
cisco-core#sh ipv6 interface brief

/Проверим доступность маршрутизаторов SC-1 и SC-2
cisco-core#ping ipv6 2001:1:1:100::2     /Пингуем интерфейс eth1
маршрутизатора soft-core1
cisco-core#ping ipv6 2001:1:1:200::2     /Пингуем интерфейс eth2
маршрутизатора soft-core2

/Создаем статические маршруты
cisco-core(config)#ipv6 route fc00:2::/64 2001:1:1:100::2  /Маршрут в
локальную сеть, находящуюся за soft-core1
cisco-core(config)#ipv6 route fc00:3::/64 2001:1:1:200::2  /Маршрут в
локальную сеть, находящуюся за soft-core2

/Просмотрим таблицу маршрутизации для IPv6
cisco-core#show ipv6 route
```

После настройки маршрутизации в ядре сети необходимо проверить взаимную доступность локальных подсетей. Для этого запустите проверку утилитой *ping6* с различных машин, находящихся в разных подсетях.

```
pc1#ping6 fc00:2::2
pc1#ping6 fc00:3::2
pc2#ping6 fc00:1::2
pc2#ping6 fc00:3::2
pc3#ping6 fc00:1::2
pc3#ping6 fc00:2::2
```

Если *ping6* проходит успешно, утилитой *traceroute6* отследите пути продвижения пакетов по сети (рис. 8).

```
bash-3.2$ ping6 -c2 fc00:1::2
PING6(56=40+8+8 bytes) fc00:2::2 --> fc00:1::2
16 bytes from fc00:1::2, icmp_seq=0 hlim=62 time=0.632 ms
16 bytes from fc00:1::2, icmp_seq=1 hlim=62 time=0.687 ms

--- fc00:1::2 ping6 statistics ---
2 packets transmitted, 2 packets received, 0.0% packet loss
round-trip min/avg/max/std-dev = 0.632/0.659/0.687/0.028 ms
bash-3.2$ traceroute6 fc00:1::2
traceroute6 to fc00:1::2 (fc00:1::2) from fc00:2::2, 64 hops max, 12 byte packets
 1  fc00:2::1  0.833 ms  0.206 ms  0.159 ms
 2  2001:1:1:100::1  0.582 ms  0.534 ms  0.552 ms
 3  fc00:1::2  0.629 ms  0.533 ms  0.564 ms
bash-3.2$
```

Рис.8 – Путь прохождения пакета

5. Снимите таблицу канального уровня. В случае использования протокола IPv6 нужно снимать таблицу не ARP, а ND-протокола (*Neighbour Discovery*). Это связано с особенностями протокола IPv6 [2].

На Cisco данная таблица выводится командой *show ipv6 neighbors* (рис.9).

```
cisco-core#show ipv6 neighbors
IPv6 Address                              Age Link-layer Addr State Interface
FC00:1::2                                  12 000c.42eb.030d  STALE Vl2
2001:1:1:200::2                           144 001b.21cc.10d9  STALE Gi0/1
2001:1:1:100::2                            12 001b.21cc.10fb  STALE Gi0/0
FE80::21B:21FF:FECC:10FB                   12 001b.21cc.10fb  STALE Gi0/0
FE80::21B:21FF:FECC:10D9                  144 001b.21cc.10d9  STALE Gi0/1
FE80::20C:42FF:FEEB:30D                    12 000c.42eb.030d  STALE Vl2

cisco-core#
```

Рис. 9 – Кэш протокола ND в Cisco

На программных маршрутизаторах посмотреть общую таблицу можно той же командой, что и в первой части работы (рис. 10).

```
[admin@localhost ~]$ ip neighbour
2001:1:1:100::1 dev eth1 lladdr 64:00:f1:19:6a:60 router STALE
2001:1:1:300::2 dev eth2 lladdr 00:1b:21:cc:0f:4e router STALE
192.168.24.99 dev eth0 lladdr 00:22:41:26:34:cc REACHABLE
[admin@localhost ~]$
```

Рис. 10 – Кэш протоколов канального уровня в Linux

Варианты заданий

Таблица 1 – Статическая маршрутизация в сетях IPv4

№ вар.	Cisco-core			Soft-core1			Soft-core2		
	FE 0/0/0	GE 0/0	GE 0/1	eth0	eth1	eth2	eth0	eth1	eth2
1	192.168.5.0/24	50.0.0.1/30	60.0.0.2/30	172.16.0.0/16	50.0.0.2/30	70.0.0.1./30	10.0.0.0/8	70.0.0.2/30	60.0.0.1/30
2	172.16.0.0/16	42.1.15.5/25	42.1.16.10/25	10.0.0.0/8	42.1.15.15/25	42.1.17.1/29	192.168.4.0/24	42.1.17.3/29	24.1.16.20/25
3	192.168.75.0/24	24.1.1.2/30	25.2.2.4/28	10.0.0.0/8	24.1.1.1/30	26.3.3.3/29	172.16.0.0/16	26.3.3.2/29	25.2.2.1/28
4	192.168.50.0/24	65.1.0.1/30	65.2.0.7/28	192.168.70.0/24	65.1.0.2/30	65.3.0.4/29	10.0.0.0/8	65.3.0.3/29	65.2.0.8/28

Таблица 2 – Статическая маршрутизация в сетях IPv6

№ вар.	Cisco-core			Soft-core1			Soft-core2		
	FE 0/0/0	GE 0/0	GE 0/1	eth0	eth1	eth2	eth0	eth1	eth2
1	fc00:ab:30:: /64	2001:a::11aa /64	2001:b::12aa/64	fc00:ab:10:: /64	2001:a::11bb /64	2001:c::13aa/64	fc00:ab:20::/64	2001:c::13bb /64	2001:b::12bb /64
2	fc00:1:3:1:: /64	2001:11:aa::1/ 64	2001:22:aa::45 /64	fc00:1:2:1:: /64	2001:11:aa::2 /64	2001:33:aa::950 /64	fc00:1:1:1::/64	2001:33:aa:: 750/64	2001:22:aa::de /64
3	fc00:b::/64	2001:a3::120 /64	2001:b4::110 /64	fc00:a::/64	2001:a3::220 /64	2001:d5::200/64	fc00:c::/64	2001:d5::100 /64	2001:b4::210 /64
4	fc00:74:52:300:: /64	2001:1:2:a:: 64bf/64	2001:1:2:b::123 /64	fc00:74:52: 200::/64	2001:1:2:a:: 12cd/64	2001:1:2:c::f5 /64	fc00:74:52:100:: /64	2001:1:2:c:: f4/64	2001:1:2:b::246 /64

К защите:

1. Знать особенности адресации в сетях IPv4 и IPv6, иметь представление о функциях коммутаторов и маршрутизаторов.

2. Уметь производить конфигурацию маршрутизаторов для организации статической маршрутизации в сетях IPv4/IPv6.

3. Представить отчет, содержащий листинги производимых действий по настройке оборудования, результаты проверки работоспособности сконфигурированных сетей (таблицы маршрутизации, результаты проверки доступности узлов и подсетей).

Рекомендуемая литература:

1. Олифер Н. А., Олифер В.Г. Компьютерные сети. Принципы, технологии, протоколы. Издание 4-е. Питер, 2010.

2. Таненбаум Э., Уэзеролл Д. Компьютерные сети. Издание 5-е. СПб, Питер, 2012г.

5. Настройка сервера динамического конфигурирования хостов (DHCP) в локальной сети

Цель лабораторной работы: получение навыков по организации адресного пространства в локальной и корпоративной сети.

Краткая теоретическая справка

IP-адреса в сети могут назначаться стационарно (вручную) или динамически, для чего используется специализированный протокол DHCP. Dynamic Host Configuration Protocol – протокол динамической конфигурации узла – это сетевой протокол, позволяющий компьютерам автоматически получать как IP-адрес, так и адреса DNS-серверов и другие параметры, необходимые для работы в сети TCP/IP.

Во взаимодействии по протоколу DHCP могут принимать участие следующие стороны:

- DHCP-клиент – устройство, запрашивающее параметры настройки TCP/IP;
- DHCP-сервер – устройство, выдающее параметры настройки TCP/IP;
- DHCP-ретранслятор (relay agent) – вспомогательный участник, который может играть роль посредника между клиентом и сервером. DHCP-ретранслятор обрабатывает стандартный широковещательный DHCP-запрос и перенаправляет его на DHCP-сервер в виде целенаправленного (unicast) пакета, а полученный от DHCP-сервера

ответ, в свою очередь, перенаправляет DHCP-клиенту. Является не обязательным.

Стандартная процедура получения конфигурации от DHCP-сервера показана на рис. 1.

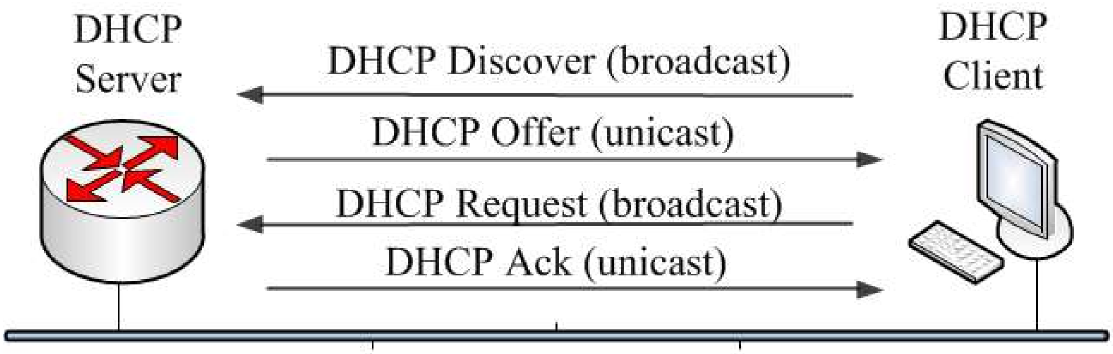


Рис. 1 – Диаграмма процедура получения сетевой конфигурации от DHCP-сервера

Коммутатор может быть сконфигурирован как агент DHCP Relay. В этом случае он только перенаправляет DHCP-запрос от клиента локальной подсети на удалённый DHCP-сервер. Специализированная опция (Option 82) используется для передачи дополнительной информации в DHCP-запросе, которую добавляет непосредственно коммутатор. Опция 82 состоит из двух подопций:

1. *Agent Circuit ID* – содержит информацию о том, с какого порта пришел запрос на DHCP-ретранслятор.
2. *Agent Remote ID* – идентификатор самого DHCP-ретранслятора (который задается при настройке, можно, например, использовать MAC-адрес коммутатора или его описание, любое удобное значение).

Опция 82 в запросе DHCP приведена на рисунке 2:

```
▽ Option: (t=82,l=12) Agent Information Option
     Option: (82) Agent Information Option
     Length: 12
     Value: 010200030206001560792800
     Agent Circuit ID: 0003
     Agent Remote ID: 001560792800
  End Option
```

Рис. 2 – Вид опции 82 в запросе DHCP

При выполнении данной лабораторной работы потребуется следующее оборудование: маршрутизатор Cisco, управляемый коммутатор второго уровня SW-L2-1, рабочая станция управления макетом (mnlin), при необходимости – неуправляемый коммутатор D-Link DES-1016.

Задание на работу:

Организовать локальную сеть, в которой подключаемые клиенты будут получать конфигурацию сетевых интерфейсов динамически от DHCP-сервера на базе маршрутизатора. Раздаваемые адреса должны находиться в подсети 192.168.24.0/24, причем адреса в диапазоне 192.168.24.1 – 192.168.24.59 зарезервированы (не должны выдаваться клиентам). Клиенты, подключенные к портам 5 и 7 коммутатора, должны всегда получать адреса 192.168.24.101 и 192.168.24.102 соответственно.

Методика выполнения работы

Перед выполнением лабораторной работы необходимо привести макет в исходное состояние, загрузить в сетевое оборудование, которое будет использоваться в данной лабораторной работе, соответствующие конфигурационные файлы, убедиться в работоспособности файлового сервера (см. Инструкцию по обслуживанию макета).

1. Подключить терминальными кабелями маршрутизатор Cisco-core и управляемый коммутатор SW-L2-1 к компьютеру управления (mnlin), включить сетевое оборудование, настроить статические сетевые интерфейсы на mnlin согласно схеме сети (рис. 3), проверить доступность FTP-сервера.

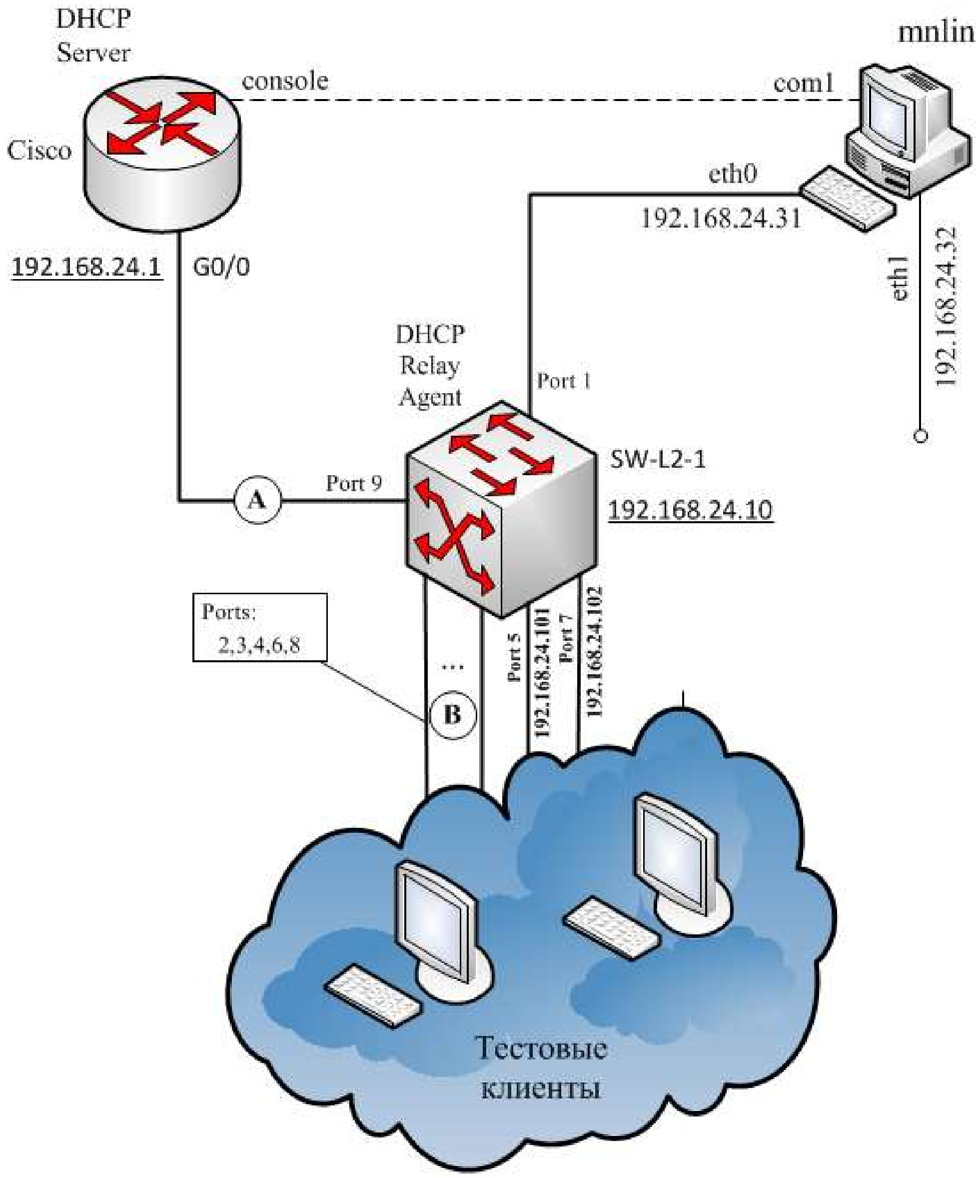


Рисунок 3 – Схема лабораторной сети

2. Включить в коммутаторе поддержку DHCP Relay Agent. Для этого с компьютера управления mnlin необходимо зайти на web-интерфейс коммутатора, который находится по адресу 192.168.24.10. После зайти во вкладку Configuration→DHCP Relay→DHCP Relay Global Settings и выставить значения полей DHCP Relay State и DHCP Relay Agent Information Option 82 в состояние Enable (рис.4).

Примечание: При необходимости здесь же можно изменить MAC-адрес коммутатора, передающийся DHCP- серверу в опции 82, выставив его в поле DHCP Relay Agent Information Option 82 Remote-ID.

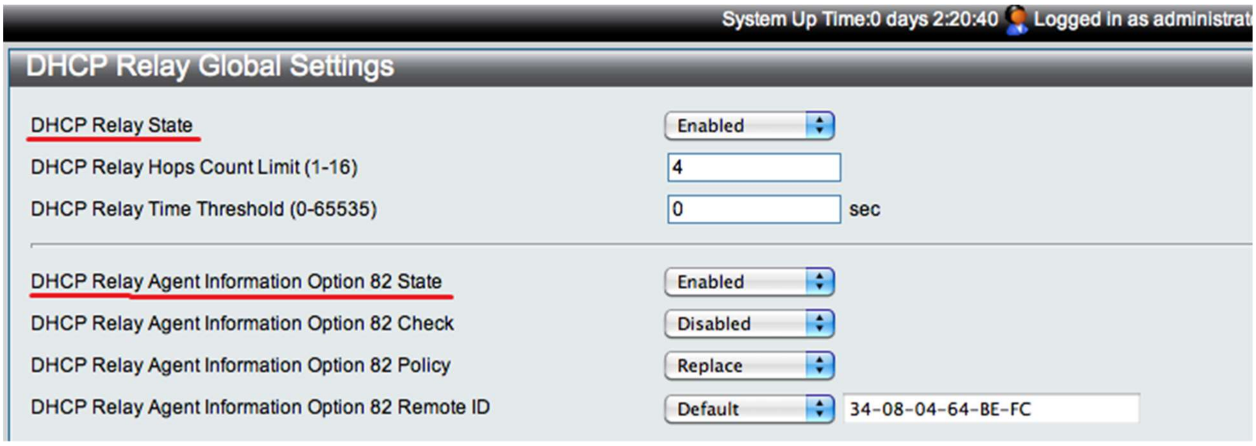


Рисунок 4 – Вкладка DHCP Relay Global Settings

После этого необходимо указать адрес используемого DHCP-сервера на вкладке Configuration→DHCP Relay→DHCP Interface Settings (рис.5). Здесь нужно добавить IP-адрес будущего сервера: 192.168.24.1, и принять сделанные изменения.

Рисунок 5 – Вкладка DHCP Interface Settings

3. Для того чтобы сервер мог распознать запросы, содержащие опцию 82, нужно выяснить вид данной опции в конкретном случае. Как говорилось выше, существует стандартная форма опции 82, однако у разных производителей оборудования эта реализация может отличаться. Поэтому надежным вариантом будет определение её вида на конкретном оборудовании:

а) Собрать вспомогательную схему сети, представленную на рис.6, для определения вида опции 82.

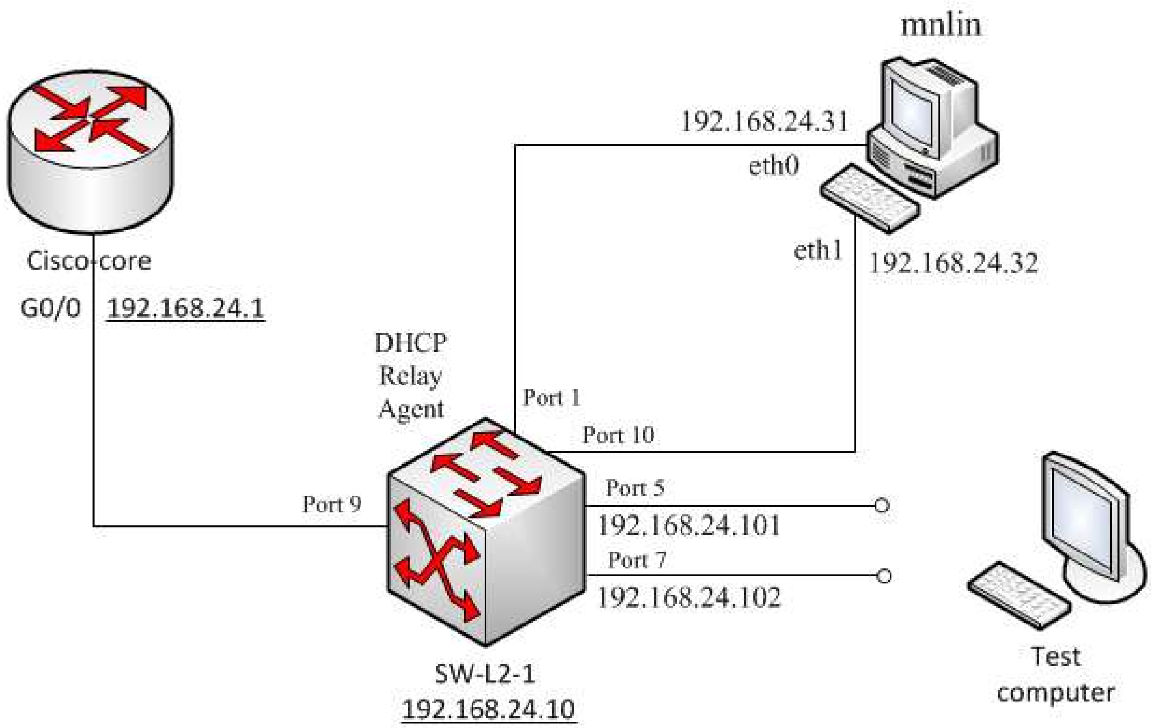


Рис.6 – Схема вспомогательной сети

b) Настроить полное зеркалирование трафика с порта 9 на порт 10. Это позволит отслеживать запросы DHCP Discover с добавленной опцией 82, посылаемые коммутатором в сторону ранее указанного нами DHCP-сервера (192.168.24.1). Находясь в web-интерфейсе коммутатора, перейдите на вкладку L2 Features→Port Mirror и выставьте зеркалирование всего трафика с порта 9 на порт 10 как показано на рис.7.

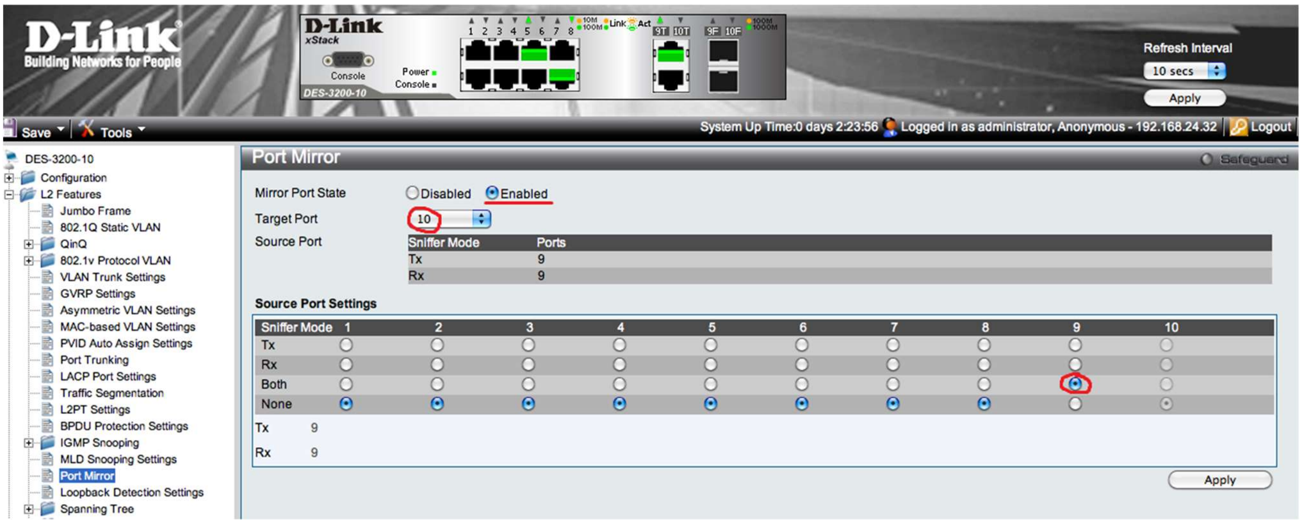


Рис.7 – Настройка зеркалирования портов

c) Запустить анализатор трафика на интерфейсе eth1 mnlin, соединенном с портом 10 коммутатора. Для уменьшения количества пакетов в трейсе нажимайте Start в тот момент, когда будете готовы выполнить следующий пункт или подключите фильтр трафика.

d) Подключить тестовую машину (любой компьютер из имеющихся в лаборатории) с сетевым интерфейсом настроенным на получение конфигурации по протоколу DHCP к порту 5 коммутатора.

e) Определить вид поля Option 82 в запросе DHCP Discover. После подключения тестовой машины достаточно снимать трэйс в течение приблизительно 10 секунд. Для облегчения поиска в полученном трэйсе используйте фильтрующее выражение «bootp». В трэйсе необходимо найти первое по времени сообщение протокола DHCP, направленное на адрес 192.168.24.1 (это и будет сообщение Discover), и в заголовке верхнего уровня найти поле Option (82): Agent information option. Значение value и будет интересующей опцией 82. Данное значение необходимо записать для последующего использования.

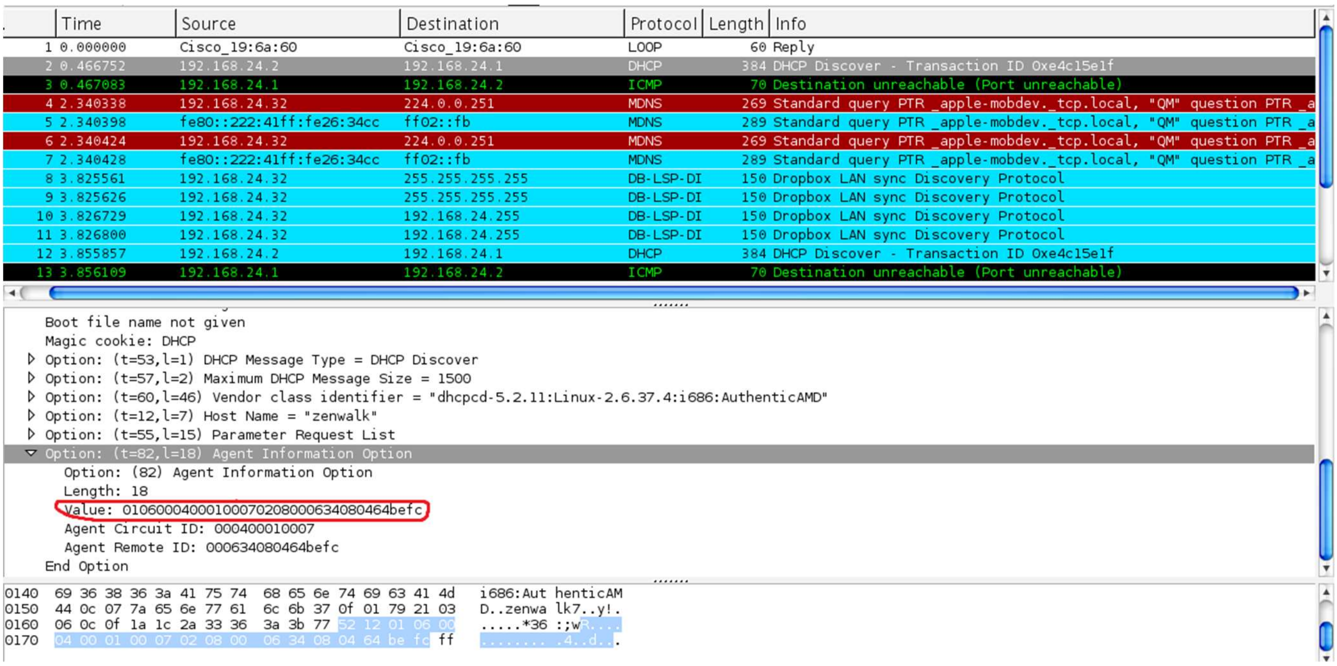


Рисунок 8 – Получение значения опции 82

f) Выполнить аналогичную процедуру для порта 7.

4. Собрать сеть согласно схеме, приведенной на рис.3.

5. Сконфигурировать DHCP-сервер. С компьютера управления макетом (mnlin) необходимо подключиться к консольному порту маршрутизатора Cisco и сконфигурировать на нем DHCP-сервер. Будем использовать для этого терминальную утилиту picocom:

```
root@mnlin# picocom /dev/ttyS0
```

где ttyS0 – символьное устройство порта com1, ttyS1 – соответственно, com2 и т.д. Для выхода из утилиты picocom нажмите одновременно клавиши Ctrl+A+Q.

Ниже приведен пример конфигурации DHCP-сервера с необходимыми пояснениями в виде: «команда/**пояснение**».

```
cisco-core>
cisco-core>enable                    /Входим в привилегированный режим
cisco-core#configure terminal        /режим конфигурирования
cisco-core(config)#interface gigabitethernet 0/0     /выбираем интерфейс
cisco-core(config-if)#ip address 192.168.24.1 255.255.255.0      /задаем ip-адрес интерфейса и маску подсети
cisco-core(config-if)#no shutdown   /включаем интерфейс
cisco-core(config-if)#end                 /выходим из режима конфигурации
cisco-core#configure terminal
cisco-core(config)#ip dhcp database ftp://admin:пароль@192.168.24.31/dhcp-db
      /указываем DHCP-серверу адрес для хранения базы данных подключающихся клиентов (FTP-сервер на mnlin)
cisco-core(config)#service dhcp           /включаем службу DHCP
cisco-core(config)#ip dhcp use class      /включаем использование классов
cisco-core(config)#ip dhcp excluded-address 192.168.24.1 192.168.24.59 /задаем диапазон, адреса из которого не будут выдаваться клиентам
cisco-core(config)#ip dhcp class port5          /создаем класс port5
cisco-core(config-dhcp-class)#relay agent information /указываем, что сопоставление запроса данному классу будет производиться путем сравнения поля option 82 запроса с заданным нами значением
cisco-core(config-dhcp-class-relayinfo)#relay-information hex
010600040001000502080006340804 64befc      /указываем строку сопоставления запроса данному классу; здесь нужно использовать полученное нами ранее значение опции 82
cisco-core(config-dhcp-class-relayinfo)#end    /выходим из режима конфигурации
cisco-core#
cisco-core#configure terminal
cisco-core(config)#ip dhcp class port7
cisco-core(config-dhcp-class)#relay agent information
cisco-core(config-dhcp-class-relayinfo)#relay-information hex
01060004000100070208000634080464befc
cisco-core(config-dhcp-class-relayinfo)#end
cisco-core#
cisco-core#configure terminal
cisco-core(config)#ip dhcp class common-net     /создаем общий класс для всех запросов не соответствующих классам port5 и port7
cisco-core(config-dhcp-class)#relay agent information
cisco-core(config-dhcp-class-relayinfo)#end
cisco-core#
cisco-core#configure terminal
cisco-core(config)#ip dhcp pool LAN1      /создаем пул адресов LAN1 и далее указываем сведения, которые DHCP-сервер будет выдавать клиентам
cisco-core(dhcp-config)#network 192.168.24.0 255.255.255.0 /подсеть, в которой DHCP-сервером будут выдаваться адреса
cisco-core(dhcp-config)#domain-name lan1.maket       /доменное имя
cisco-core(dhcp-config)#dns-server 192.168.24.1      /DNS-сервер
cisco-core(dhcp-config)#default-router 192.168.24.1  /шлюз по умолчанию
cisco-core(dhcp-config)#class port5       /специфические настройки для клиентов, чей запрос соответствует классу port5
cisco-core(config-dhcp-pool-class)#address range 192.168.24.101
192.168.24.101    /диапазон выдаваемых адресов (в нашем случае, по заданию, состоит из одного адреса)
cisco-core(config-dhcp-pool-class)#end
cisco-core#configure terminal
```

```
cisco-core(config)#ip dhcp pool LAN1      /продолжаем настраивать пул LAN1
cisco-core(dhcp-config)#class port7
cisco-core(config-dhcp-pool-class)#address range 192.168.24.102
192.168.24.102
cisco-core(config-dhcp-pool-class)#end
cisco-core#
cisco-core#configure terminal
cisco-core(config)#ip dhcp pool LAN1
cisco-core(dhcp-config)#class common-net
cisco-core(config-dhcp-pool-class)#end
cisco-core#
cisco-core#show running-config      /проверяем  полученную конфигурацию
```

Включаем вывод сообщений DHCP-сервера относящихся к процедуре сопоставления приходящих запросов имеющимся классам:

```
cisco-core#debug ip dhcp server class
DHCP server class debugging is on.
```

На основе приведенной процедуры конфигурации необходимо настроить DHCP-сервер в соответствии с заданием к лабораторной работе.

6. Проверка работоспособности конфигурации производится путем подключения тестовых машин к разным портам коммутатора и определения на них назначенных IP-адресов.

Посмотреть статистику работы DHCP-сервера по базе данных можно в терминале Cisco следующими командами:

```
cisco-core#show ip dhcp bindings
cisco-core#show ip dhcp conflicts
cisco-core#show ip dhcp database
cisco-core#show ip dhcp statistics
```

После просмотра статистику можно очистить:

```
cisco-core#clear ip dhcp binding *
cisco-core#clear ip dhcp conflict *
cisco-core#clear ip dhcp server statistics
```

7. Используя анализатор трафика и возможности зеркалирования портов коммутатора, зарисуйте диаграмму процедуры получения хостами конфигурации от DHCP-сервера в точках A и B (рис.3) и сравните ее со стандартной процедурой.

Дополнительно: на основе полученных значений опции 82 определить содержащиеся в ней поля и предсказать значение опции 82 для любых портов и MAC-адресов применяемого коммутатора.

К защите:

1. Знать принципы работы протокола DHCP, назначение опции 82, иметь представление о методах зеркалирования трафика.

2. Уметь использовать анализатор трафика для оценки полученного результата.

3. Представить отчет, содержащий:

- схему сети (аналогично рис. 3);
- трейс анализатора трафика;
- диаграмму процедуры получения конфигурации хостами от DHCP-сервера в точках A и B.

Рекомендуемая литература:

1. RFC 2131: Dynamic Host Configuration Protocol, R. Droms, Bucknell University, March 1997. http://tools.ietf.org/html/rfc2131
2. RFC 3046: DHCP Relay Agent Information Option M. Patrick, Motorola BCS, January 2001. http://tools.ietf.org/html/rfc3046
3. Семенов Ю.А. Протокол динамического конфигурирования ЭВМ DHCP (и NAT/PAT). http://book.itep.ru/4/4/dhcp.htm.

6. Трансляция и туннелирование сетевых адресов

Цель работы: получение навыков настройки механизмов NAT.

Краткая теоретическая справка

Данная работа состоит их двух частей, в которых будут рассмотрены широко используемые методы организации трансляции сетевых адресов в сетях IPv4/IPv6.

Трансляция сетевых адресов является универсальным способом расширения адресного пространства. Появление системы трансляции сетевых адресов, или NAT (Network Address Translation) обусловлено бурным ростом небольших сетей, в то время относящихся к классу C, и, как следствие, сокращение IP-адресов данного класса. Тогда одним из способов расширения адресного пространства наравне с введением бесклассовой адресации стало использование неуникальных IP-адресов, иногда называемых маскарадными. Традиционно это адреса вида 192.168.0.0 и 10.0.0.0, но в последнее время могут использоваться некоторые другие. Такие адреса уникальны только в пределах закрытой сети (например, корпоративной или сети провайдера). Для выхода в общедоступную сеть необходим уникальный адрес, который присвоен шлюзу. Система NAT позволяет осуществлять подмену адресов на шлюзе.

В первой части работы мы рассмотрим такие виды трансляции сетевых адресов, как Source NAT (SNAT) и Destination NAT (DNAT). Как следует из названий, данные виды NAT подменяют адреса отправителя и получателя пакета соответственно. Маскарадингом (Masquerade) называется разновидность SNAT, в которой подменяемый адрес отправителя может изменяться динамически в соответствии с текущим адресом шлюзового интерфейса.

В операционной системе Linux за трансляцию сетевых адресов отвечает утилита *iptables* (или *ip6tables* для IPv6). Вообще, *iptables* является одним из командных интерфейсов межсетевого экрана Netfilter и используется для настройки разнообразных правил фильтрации сетевого трафика. Однако, в рамках данной работы, мы рассмотрим только одну из его возможностей – NAT.

Рассмотрим некоторые ключевые понятия *iptables*:

- Правило – состоит из критерия, действия и счетчика. Если пакет соответствует критерию, к нему применяется действие, и он учитывается счетчиком. Критерия может и не быть – тогда неявно предполагается критерий «все пакеты».
- Критерий – логическое выражение, анализирующее свойства пакета и/или соединения и определяющее, подпадает ли данный конкретный пакет под действие текущего правила.
- Действие – описание действия, которое нужно проделать с пакетом и/или соединением в том случае, если они подпадают под действие этого правила.
- Цепочка – упорядоченная последовательность правил. Цепочки можно разделить на пользовательские и базовые.
- Базовая цепочка – цепочка, создаваемая по умолчанию при инициализации таблицы. Каждый пакет, в зависимости от того, предназначен ли он самому хосту, сгенерирован им или является транзитным, должен пройти положенный ему набор базовых цепочек различных таблиц. Имена базовых цепочек всегда записываются в верхнем регистре (PREROUTING, INPUT, FORWARD, OUTPUT, POSTROUTING).
- Пользовательская цепочка – цепочка, созданная пользователем. Может использоваться только в пределах своей таблицы.
- Таблица – совокупность базовых и пользовательских цепочек, объединенных общим функциональным назначением. Имена таблиц записываются в нижнем регистре, так как в принципе не могут конфликтовать с именами пользовательских цепочек. При вызове команды iptables таблица указывается в формате -t имя_таблицы. При отсутствии явного указания, используется таблица filter.

Все пакеты пропускаются через определенные для них последовательности цепочек. При прохождении пакетом цепочки, к нему последовательно применяются все правила этой цепочки в порядке их следования. Таким образом, во-первых, пакет проверяется на соответствие критерию, а во-вторых, если он соответствует данному критерию, то к нему применяется указанное действие. Под действием может подразумеваться как элементарная операция (например, ACCEPT, REJECT), так и переход в одну из пользовательских цепочек. Действия ACCEPT, REJECT и DROP являются терминальными, т.е. прекращающими обработку пакета в рамках данной базовой цепочки.

В обобщенном виде добавление правила *iptables* можно представить так:

```
iptables -t <имя_таблицы> -A <цепочка> <критерий> -j <действие>
```

Здесь `-A` обозначает, что данное правило будет дописано в конец заданной цепочки заданной таблицы. Просмотр правил в цепочке происходит последовательно сверху вниз.

В данной работе нас будет интересовать только таблица NAT. Эта таблица содержит три базовые цепочки: PREROUTING, OUTPUT и POSTROUTING. В цепочку PREROUTING попадают все входящие пакеты, адресованные данному хоста или транзитные, до принятия хостом решения о маршрутизации пакета. Пакеты, отсылаемые данным хостом, будут проходить цепочку OUTPUT, а транзитные пакеты на выходе из узла попадут в цепочку POSTROUTING. Таким образом, можно заметить, что для организации SNAT/Masquerade на транзитном шлюзе нужно применять правила цепочки POSTROUTING, в то время как цепочка PREROUTING будет использоваться при организации DNAT.

В операциях NAT, производимых с помощью *iptables*, отслеживание состояний используется автоматически. Вам достаточно указать критерии, под которые подпадет лишь первый пакет в соединении – и трансляция адресов будет применена ко всем пакетам в этом соединении, а также в связанных с ним соединениях.

С появлением сетей IPv6 возникла необходимость совместной работы сегментов сетей с разной адресацией. Скорее всего, еще довольно долгое время доминирующим протоколом останется IPv4. В таких условиях особую актуальность приобретают методы конвергенции и взаимодействия сетей различных типов. На данный момент разработано множество решений этой проблемы, из наиболее распространенных можно назвать 6to4, 6rd, Teredo, NAT64 и др. В этой части лабораторной работы мы рассмотрим универсальный способ передачи трафика IPv6 через сети IPv4, основанный на использовании протокола 6to4 (рис.1).

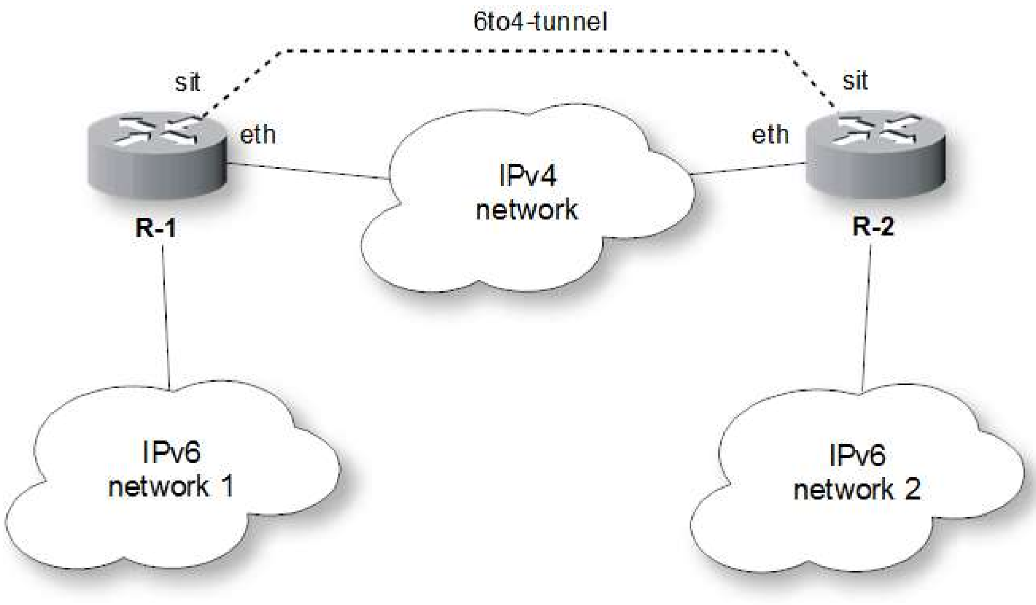


Рис. 1 – Туннель 6to4

Протокол 6to4 предназначен для связи двух подсетей IPv6 через IPv4 и инкапсулирует пакеты версии 6 в тело обыкновенных IPv4-пакетов. Подробное описание работы протокола 6to4 вы можете найти в RFC3056

Задание на работу

Часть1:

1. Соберите сеть для выполнения первой части лабораторной работы согласно полученному заданию (рис. 2, табл. 4).
2. Настройте SNAT/маскарадинг на soft-core1 согласно приведенной схеме подключившись к программному маршрутизатору.
3. Подключитесь к компьютеру управления mnlin и запустите генератор трафика Ostinato. Создайте для порта eth1 поток UDP-пакетов с произвольными портами (1024-65535), фиксированным IPv4-адресом отправителя (из указанных в задании подсети) и адресом назначения.
4. Настройте DNAT на маршрутизаторе soft-core2. Измените созданный ранее UDP-поток таким образом, чтобы портом назначения был 34001.
5. Проведите проверку правильности конфигурации NAT.

Часть 2:

6. Соберите сеть для выполнения второй части лабораторной работы согласно полученному заданию (рис. 7, табл. 5).
7. Подключитесь к программным маршрутизаторам и создайте туннель согласно методике выполнения лабораторной работы.
8. Проверьте работу созданного туннеля.

Методика выполнения работы

Часть 1. «Классический» NAT в сетях IPv4

Рассмотрим организацию трансляции адресов в соответствии с приведенной на рисунке 2 схемой.

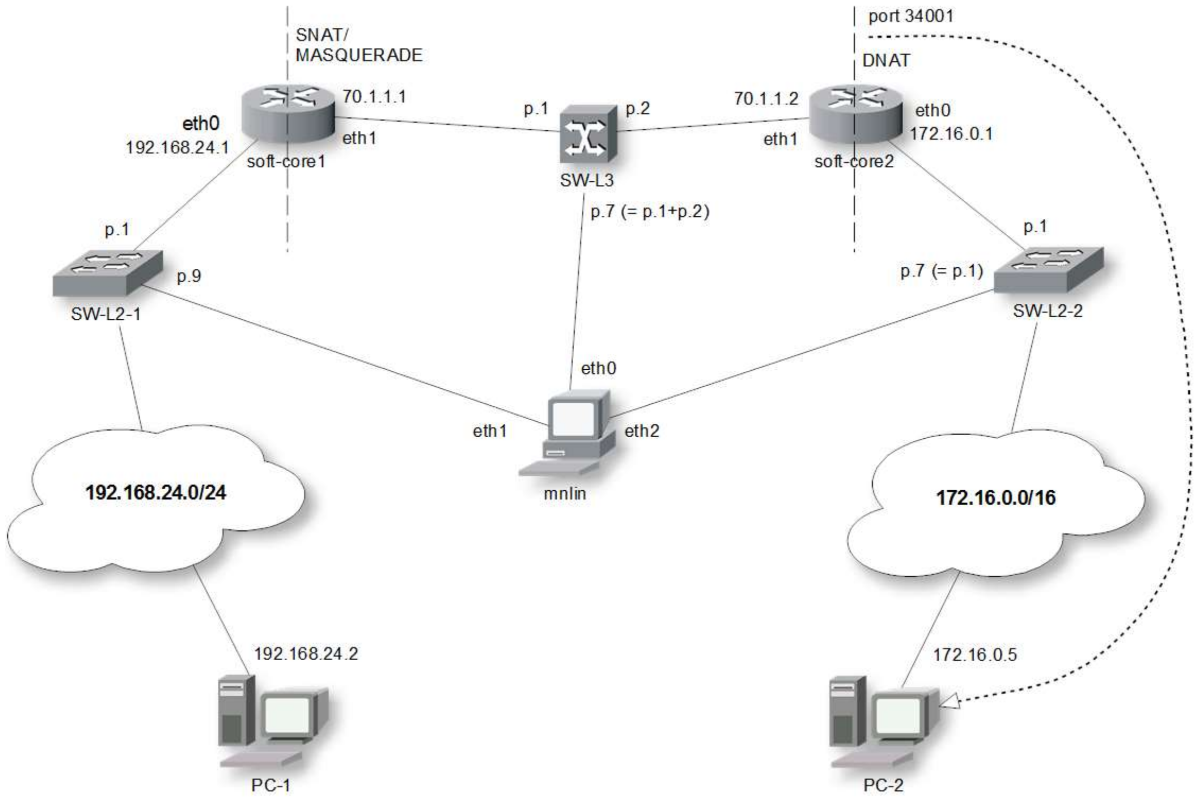


Рис. 2 – Схема сети для организации NAT

Таблица 1

Интерфейс	**Адрес**
mnlin – eth0	не выставляется
mnlin – eth1	192.168.24.33/24
mnlin – eth2	не выставляется
soft-core1 – eth0	192.168.24.1/24
soft-core1 – eth1	70.1.1.1/30
soft-core2 – eth0	172.16.0.1/16
soft-core2 – eth1	70.1.1.2/30
PC-1	192.168.24.2/24
PC-2	172.16.0.5/16

В данной работе мы воспользуемся встроенными средствами Linux для управления сетевыми интерфейсами, маршрутами и т.д. Удобным инструментом, объединяющим различные сетевые программные средства, является созданная нашим соотечественником утилита *ip*. Синтаксис этой утилиты довольно обширен, и не будет рассматриваться в рамках данной работы, отметим лишь необходимые команды. Для задания IP-адреса сетевому интерфейсу сначала необходимо удалить старый адрес, а затем присвоить новый.

```
root@sc-1# ip addr del <старый_адрес/префикс> dev <интерфейс>
root@sc-1# ip -4/-6 addr add <новый_адрес/префикс> dev <интерфейс>
```

В данном случае опции -4 и -6 обозначают тип адреса: IPv4 или IPv6 соответственно. Например, так может выглядеть смена адреса eth0:

```
root@sc-1# ip addr del 192.168.24.1/24 dev eth0
root@sc-1# ip -4 addr add 10.0.0.1/8 dev eth0
```

Просмотреть информацию о сетевых интерфейсах можно введя команду *ifconfig* (все интерфейсы) или *ifconfig <интерфейс>* (для просмотра конкретного интерфейса). Описанным способом настройте все сетевые интерфейсы маршрутизаторов в соответствии с вашим вариантом задания.

Примечание: Настраивать сетевые интерфейсы удобнее всего подключаясь к маршрутизаторам поочередно с помощью KVM-переключателя.

При сборке сети не забудьте настроить зеркалирование указанных портов коммутаторов. На схеме запись вида «p.7 (= p.1)» обозначает, что на порту 7 настроено зеркалирование всего трафика с порта 1.

Таблица 2

Коммутатор	**Адрес Web-интерфейса**
SW-L2-1	192.168.24.10
SW-L2-2	192.168.24.20
SW-L3	192.168.24.30

Для настройки зеркалирования в коммутаторах SW-L2-1 и SW-L2-2 зайдите в веб-интерфейс нужного коммутатора (login: admin, passw: admin), перейдите на вкладку L2 Features→Port Mirror и выставьте зеркалирование как показано на рис. 3.

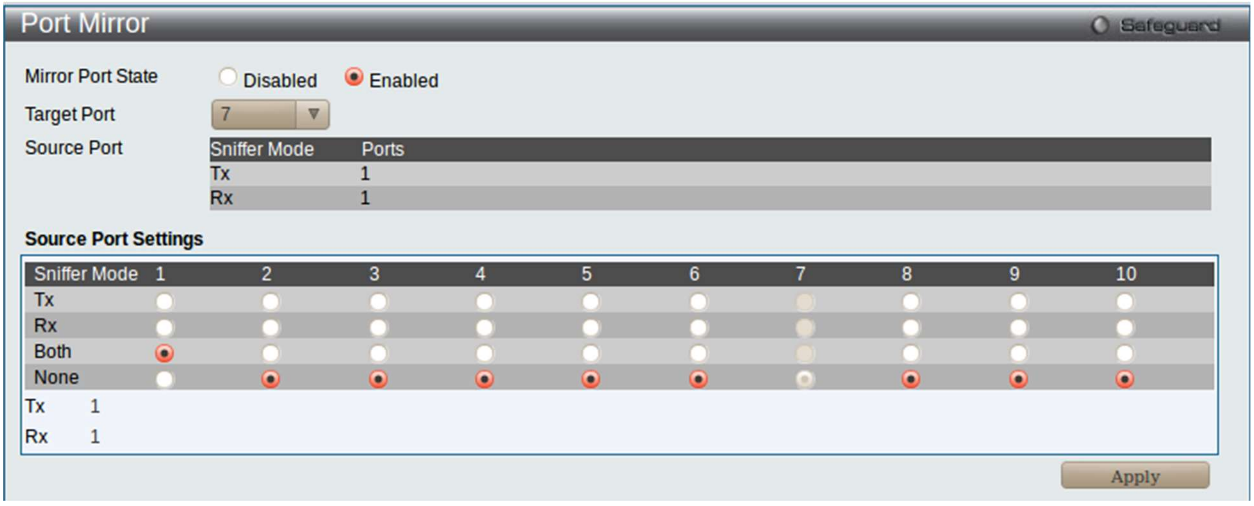


Рис. 3 – Настройка зеркалирования портов в SW-L2-1/ SW-L2-2

Процедура настройки зеркалирования портов SW-L3 напоминает вышеописанную, с той лишь разницей, что она производится на вкладке Monitoring→Mirror→Port Mirror Settings (рис. 4).

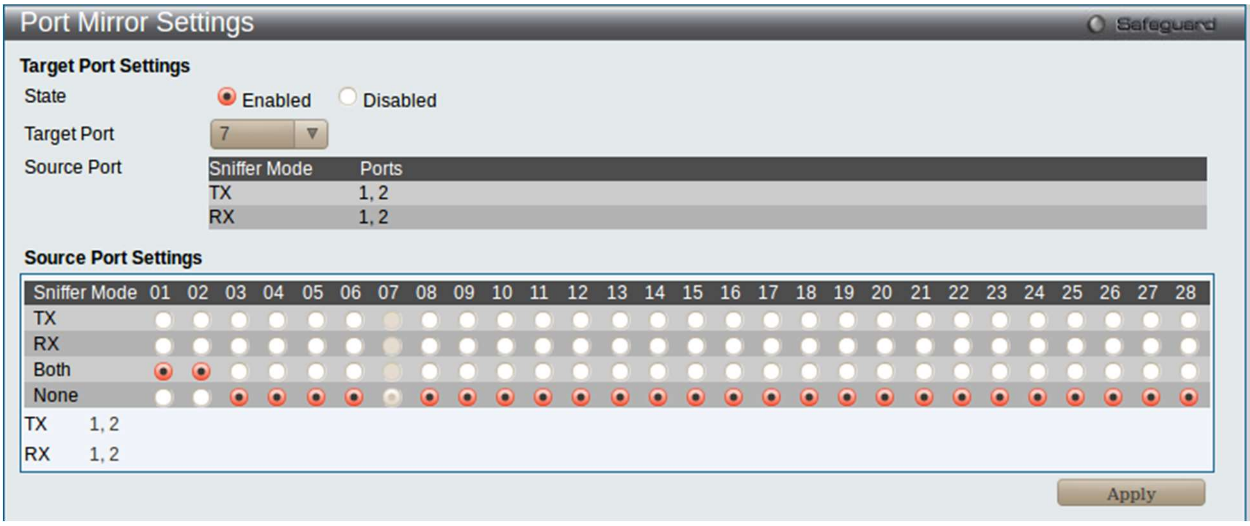


Рис.4 – Настройка зеркалирования портов в SW-L3

В поле Target Port указывается тот порт, на который будет осуществляться зеркалирование трафика.

Для настройки маскарадинга на soft-core1 согласно приведенной схеме необходимо подключиться к программному маршрутизатору и выполнить следующие действия:

```
student@pc# ssh admin@192.168.24.1     /подключаемся к SC-1
admin@sc-1# su                         /получаем права суперпользователя
root@sc-1# sysctl net.ipv4.ip_forward=1     /включаем IP-forwarding
root@sc-1# iptables -F                 /очищаем таблицу filter
root@sc-1# iptables -t nat -F              /очищаем таблицу nat
/правила трансляции:
```

root@sc-1# iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE
/добавляем правило, осуществляющее маскарадинг всех пакетов, выходящих из интерфейса eth1 данного хоста (т.е. SC-1)

Можно применить и другое правило для настройки SNAT:

root@sc-1# iptables -t nat -A POSTROUTING -o eth1 -j SNAT --to-source 70.1.1.1 **/подменяем IP-адрес отправителя всех пакетов выходящих из интерфейса eth1 на 70.1.1.1**

/Просмотреть имеющиеся в таблице nat правила можно командой

root@sc-1# iptables -t nat --list

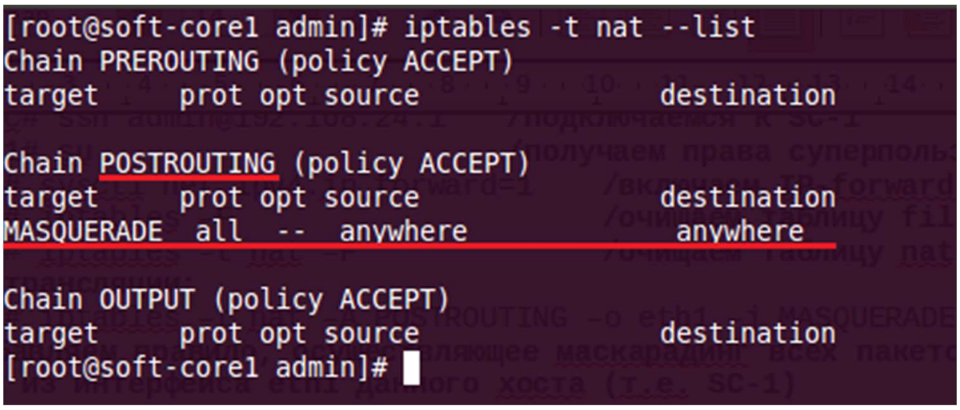


Рис. 5 — Просмотр правил трансляции сетевых адресов

Маскарадинг может адаптироваться к изменяющемуся динамическому внешнему адресу шлюза, а SNAT использует меньше системных ресурсов маршрутизатора, т.к. не проверяет адрес шлюза для каждого исходящего пакета.

С помощью KVM-переключателя подключитесь к компьютеру управления mnlin и запустите генератор трафика Ostinato (см. л/р №2). На порту eth0 mnlin запустите Wireshark, а для порта eth1 создайте поток UDP-пакетов с произвольными портами (1024-65535), фиксированным IPv4-адресом отправителя из подсети 192.168.24.0/24 и адресом назначения 70.1.1.2.

Прим.: Для успешной передачи потока в поле Destination Address протокола Media Access Protocol необходимо выставить реальное значение MAC-адреса интерфейса eth0 маршрутизатора soft-core1 (рис. 6). Узнать его можно командой:

root@soft-core1# ifconfig eth0

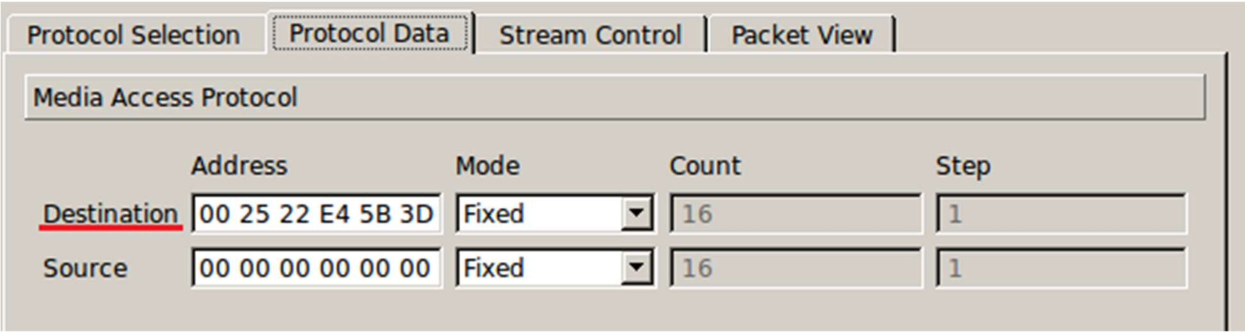


Рис. 6 — Настройка генератора трафика

Запустив генерацию, отследите в анализаторе трафика этот поток по адресу назначения. В поле IP-адреса отправителя у всех пакетов должно быть значение 70.1.1.1, т.е. адрес шлюзового интерфейса.

Теперь нужно настроить DNAT на маршрутизаторе soft-core2.

```
student@pc# ssh admin@172.16.0.1        /подключаемся к SC-2
admin@sc-2# su                          /получаем права суперпользователя
root@sc-2# sysctl net.ipv4.ip_forward=1      /включаем IP-forwarding
root@sc-2# iptables -F                  /очищаем таблицу filter
root@sc-2# iptables -t nat -F                /очищаем таблицу nat
/правила трансляции:
root@sc-2# iptables -t nat -A PREROUTING -d 70.1.1.2 -p UDP --dport 34001 -j DNAT --to-destination 172.16.0.5 /подменяем у всех пакетов UDP пришедших на порт 34001 адрес назначения на 172.16.0.5 (порт назначения остается прежним – 34001)
root@sc-2# iptables -t nat -A PREROUTING -d 70.1.1.2 -p TCP --dport 34001 -j DNAT --to-destination 172.16.0.5 /то же для TCP
```

Запустите на mnlin прослушивание порта eth2 с помощью Wireshark, а в Ostinato для порта eth1 измените созданный ранее UDP-поток таким образом, чтобы портом назначения был 34001. Начав передачу трафика, снимайте трэйс с порта eth2. Результатом правильной настройки работы механизма трансляции адресов всей сети должен стать исходный поток UDP-пакетов с адресом отправителя 70.1.1.1 и адресом назначения 172.16.0.5.

Проверка правильности выполненных настроек также может быть проведена с помощью *netcat*. *Netcat* представляет собой простую утилиту для чтения и пересылки данных посредством протоколов TCP и UDP. Для установления TCP-соединения на узле 172.16.0.5 необходимо запустить netcat в режиме сервера, прослушивая (listen), например, порт 34001 (или любой другой незанятый порт):

```
root@pc-2# nc -l -p 34001
```

а на узле 192.168.24.2 – в режиме клиента, подключающегося к серверу по адресу 70.1.1.2 на тот же порт 34001:

```
root@pc-1# nc 70.1.1.2 34001
```

В случае успешного установления соединения, станет возможным обмен текстовыми сообщениями между двумя экземплярами netcat, образовав некоторое подобие текстового чата.

Последовательно снимая трафик с портов eth0 и eth2 компьютера управления mnlin во время передачи данных с помощью *netcat*, убедитесь в том, что трансляция адресов на разных участках сети работает должным образом.

Часть 2. Преобразование Ipv4/IPv6

В данной работе мы создадим автономный туннель (без выхода в глобальную сеть), проходящий через сеть IPv4 и связывающий между собой две подсети IPv6. Во второй части будет рассматриваться вариант построения сети, приведенный на рисунке 7.

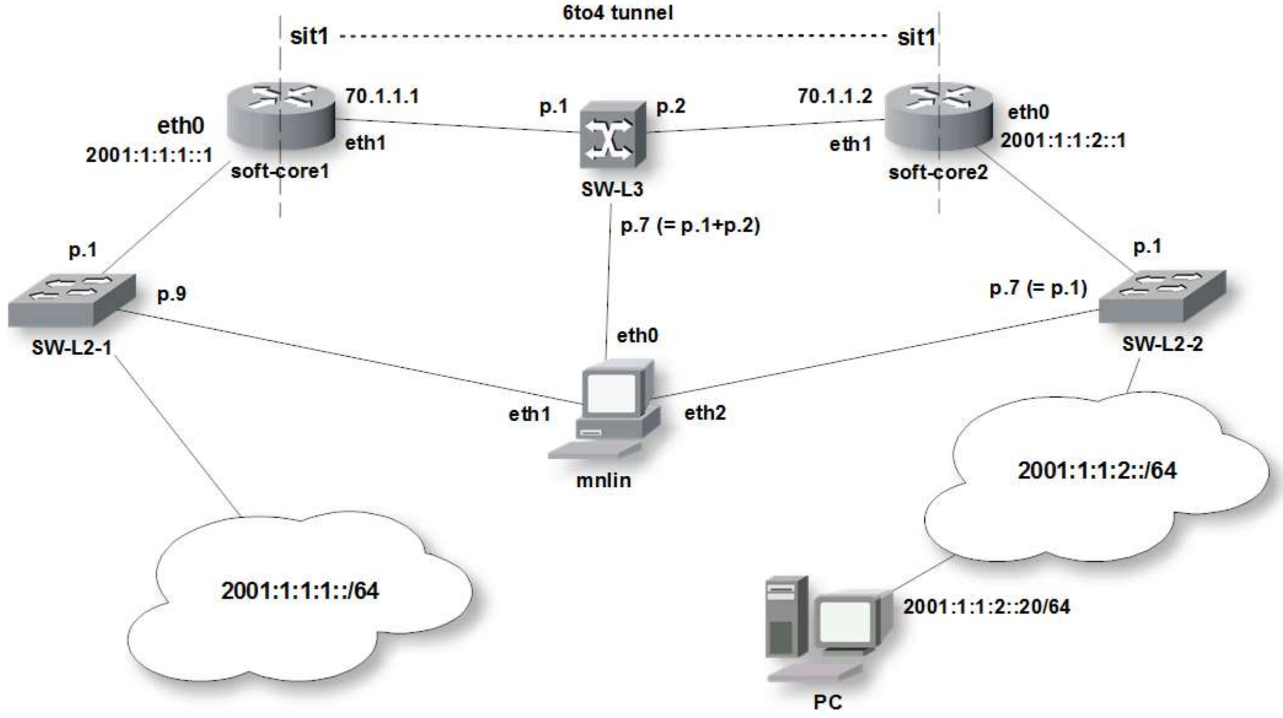


Рис. 7 – Схема сети для настройки 6to4

Если вы выполнили первую часть работы, настройка зеркалирования портов коммутаторов заново не потребуется.

Таблица 3

Интерфейс	Адрес
mnlin – eth0	не выставляется
mnlin – eth1	2001:1:1:1::33/64
mnlin – eth2	не выставляется
soft-core1 – eth0	2001:1:1:1::1/64
soft-core1 – eth1	70.1.1.1/30
soft-core2 – eth0	2001:1:1:2::1/64
soft-core2 – eth1	70.1.1.2/30
PC	2001:1:1:2::20/64

2. Подключитесь (по SSH или с помощью KVM) к программным маршрутизаторам и на каждом из них выполните описанные ниже действия.

Сначала необходимо удалить существующие правила файервола и включить транзит трафика IPv6:

```
root@sc-1# sysctl net.ipv6.conf.all.forwarding=1 /включаем IPv6 Forwarding
root@sc-1# iptables -F         /очищаем таблицу filter
root@sc-1# iptables -t nat -F  /очищаем таблицу nat
root@sc-1# ip6tables -F        /очищаем таблицу filter для IPv6
```

Очистить таблицы нужно на обоих маршрутизаторах!

Теперь зададим IPv6-адреса интерфейсам eth0 маршрутизаторов. В общем виде команда выглядит так:

```
ip -6 addr add <ipv6-адрес>/<префикс> dev <интерфейс>
```

Например, для задания IPv6-адреса интерфейсу eth0 маршрутизатора soft-core1 команда примет следующий вид:

```
root@sc-1# ip -6 addr add 2001:1:1:1::1/64 dev eth0
```

а для soft-core2:

```
root@sc-2# ip -6 addr add 2001:1:1:2::1/64 dev eth0
```

Просмотреть текущие настройки сетевых интерфейсов можно запустив от суперпользователя команду ifconfig (все интерфейсы) или ifconfig <имя_интерфейса> (выбранный интерфейс, например eth0).

Приступим к настройке 6to4-туннеля. Для настройки туннеля 6to4 необходимо настроить виртуальный туннельный адаптер, включить его и прописать статический маршрут в сеть на противоположном конце туннеля. В обобщенном виде создание туннельного адаптера будет выглядеть так:

```
ip tunnel add <имя_адаптера> mode sit ttl <время_жизни_пакета>
remote <ipv4_адрес_удаленного_конца_туннеля> local
<локальный_адрес_ipv4>
```

Имена адаптеров принято давать по типу адаптера (здесь sit) + порядковый номер, начиная с единицы.

Весь процесс настройки на стороне soft-core1:

```
root@sc-1# ip tunnel add sit1 mode sit ttl 64 remote 70.1.1.2 local 70.1.1.1
root@sc-1# ip link set dev sit1 up
root@sc-1# ip -6 route add 2001:1:1:2::/64 dev sit1 metric 1
```

В отличие от лабораторной работы, посвященной статической маршрутизации, в данном случае для создания простого статического маршрута мы используем стандартные средства Linux – утилиту *ip*. В общем виде для создания маршрута применяется такая команда:

```
ip -6 route add <удаленная_сеть/префикс> dev <адаптер> metric <значение_метрики>
```

Здесь опция «-6» указывает на то, что используется адресация IPv6, имя адаптера указывает исходящий интерфейс, а значение метрики определяет приоритет маршрута: 1 – высший приоритет.

Аналогичным образом будет выглядеть процесс настройки на другой стороне туннеля (soft-core2):

```
root@sc-2# ip tunnel add sit1 mode sit ttl 64 remote 70.1.1.1 local 70.1.1.2
root@sc-2# ip link set dev sit1 up
root@sc-2# ip -6 route add 2001:1:1:1::/64 dev sit1 metric 1
```

Просмотреть существующий 6to4 туннель можно командами (рис. 8):

```
ip -6 tunnel show sit1
ifconfig sit1
```

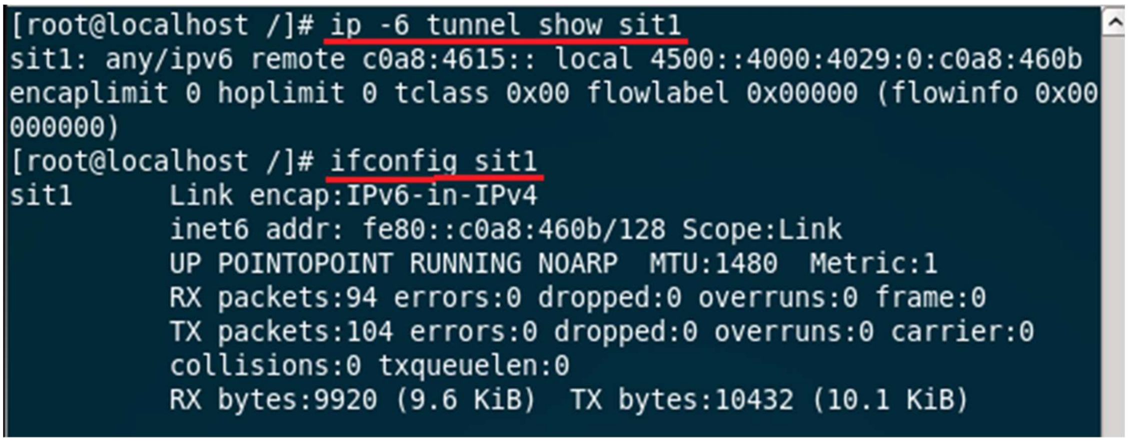


Рис. 8 – Информация о туннеле

Для проверки работоспособности туннеля запустите генератор трафика Ostinato на mnlin и создайте новый UDP/IPv6-поток для интерфейса eth1.

Соберите пакеты для потока как показано на рисунке 9.

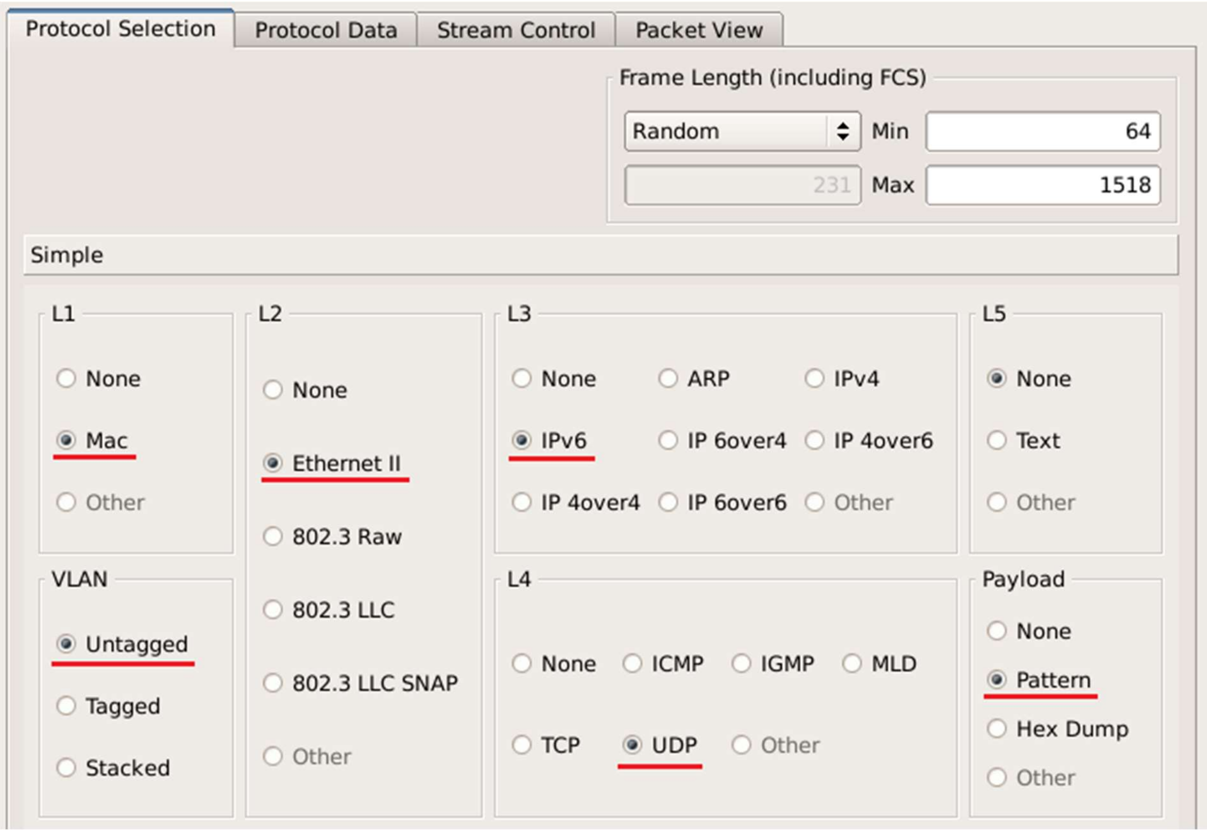


Рис. 9 – Проверочный поток. Конструирование пакета

При создании потока не забудьте указать верный адрес назначения в поле Media Access Protocol (MAC-адрес порта eth0 маршрутизатора soft-core1). Также не забудьте выставить верные значения в полях заголовка сетевого уровня IPv6 (см. рис. 10).

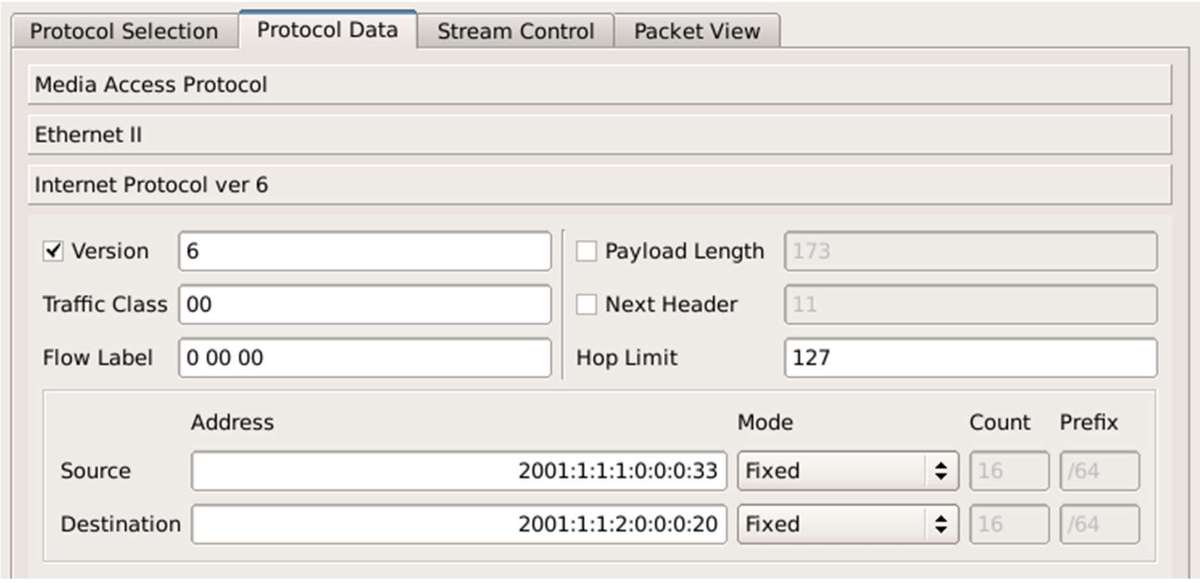


Рис.10 – Проверочный поток. Сетевой уровень

Запустив генерацию трафика, анализатором Wireshark снимите трафик с порта eth0 и убедитесь в том, что в трейсе находятся инкапсулированные в IPv4 пакеты IPv6. Затем снимайте трафик с порта eth2. В случае успешной работы 6to4-туннеля вы увидите чистые IPv6-пакеты с адресом отправителя из сети 2001:1:1:1::/64.

Варианты заданий:

Таблица 4 – Задания к части 1

Вариант	Подсеть 1 (за SC-1)	Подсеть 2 (за SC-2)	soft-core1 eth1	soft-core2 eth1	PC-2 IP-адрес и порт DNAT	Тип NAT
1	192.168.17.0/24	192.168.78.0/24	51.1.1.2/27	51.1.1.18/27	.12:35468	MASQ
2	10.0.0.0/8	172.16.0.0/16	78.2.5.6/28	78.2.5.3/28	.98:47362	SNAT
3	192.168.82.0/24	10.0.0.0/8	122.98.2.1/26	122.98.2.3/26	.44:55612	SNAT
4	172.16.0.0/16	192.168.67.0/24	65.4.15.2/29	65.4.15.1/29	.75:29867	MASQ

Таблица 5 – Задания к части 2

Вариант	Подсеть 1 (за soft-core1)	Подсеть 2 (за soft-core2)	soft-core1 eth1	soft-core2 eth1
1	2001:8fa:ba4:23::/64	2001:8fa:ba4:45::/64	15.79.54.16/24	15.79.54.122/24
2	2001:9487:26ac::/64	2001:9487:ac4::/64	78.2.5.6/28	78.2.5.3/28
3	2001:b64:84::/64	2001:b64:68d::/64	203.78.9.1/29	203.78.9.3/29
4	2001:29:a1:330::/64	2001:29:a1:320::/64	51.1.1.2/27	51.1.1.18/27

К защите:

1. Знать методы расширения адресного пространства в сетях IPv4 и создания туннелей в сетях IPv4/IPv6, иметь представлении о туннелировании в сетях NGN.
2. Уметь настраивать NAT для организации расширенного адресного пространства в сетях IPv4, уметь настраивать туннели 6to4,

использовать специализированные утилиты для проверки состояния туннеля.

3. Представить отчет, содержащий листинги проведенных действий, трейсы анализатора трафика, доказывающие правильность настройки NAT и туннеля 6to4.

Рекомендуемая литература:

1. Олифер Н. А., Олифер В.Г. Компьютерные сети. Принципы, технологии, протоколы. Издание 4-е. Питер, 2010.

2. Официальная документация Netfilter: http://netfilter.org/documentation/

3. RFC 2893. Transition Mechanisms for IPv6 Hosts and Routers. August 2000.

4. RFC 3056. Connection of IPv6 Domains via IPv4 Clouds. February 2001.

5. IP Command Reference: http://linux-ip.net/gl/ip-cref/

7. Настройка локального сервера имен (DNS)

Цель работы: изучение методов организации работы с символьными адресами.

Краткая теоретическая справка

Существует два вида методов организации пространства символьных имен в IP-сетях. Первый, называемый «плоской адресацией», позволяет присваивать имена хостам в небольших локальных сетях, используя широковещательную рассылку. Данный метод подходит только для небольших локальных сетей, так как требует настройки вручную всех хостов сети и в настоящее время используется редко.

В крупных сетях используется второй способ: применение централизованной службы, поддерживающей соответствие между различными типами адресов всех компьютеров сети. Такой службой является DNS (Domain Name System – система доменных имен), основанная на распределенной базе соответствий доменных имен IP-адресам.

Служба DNS использует в своей работе принцип «клиент-сервер». DNS-серверы поддерживают распределенную базу соответствий, а DNS-клиенты обращаются к серверам с запросами о разрешении доменных имен в IP-адреса. В качестве базы соответствий используются текстовые файлы вида «доменное имя – IP-адрес», подготавливаемые администратором, и использует в основе иерархию доменов. При росте количества узлов в сети проблема масштабирования решается созданием новых доменов и субдоменов имен и добавлением в службу DNS новых серверов.

На рисунке 1 представлена логическая структура службы DNS. Все домены верхнего (первого) уровня разделяются на три типа: территориальная принадлежность или домены государств (двухбуквенные), принадлежность к сообществам – профессиональным и т.п. (трехбуквенные), информационные домены (четырехбуквенные). Среди информационных доменов отдельно отмечается домен .arpa, зарезервированный за службой DNS.

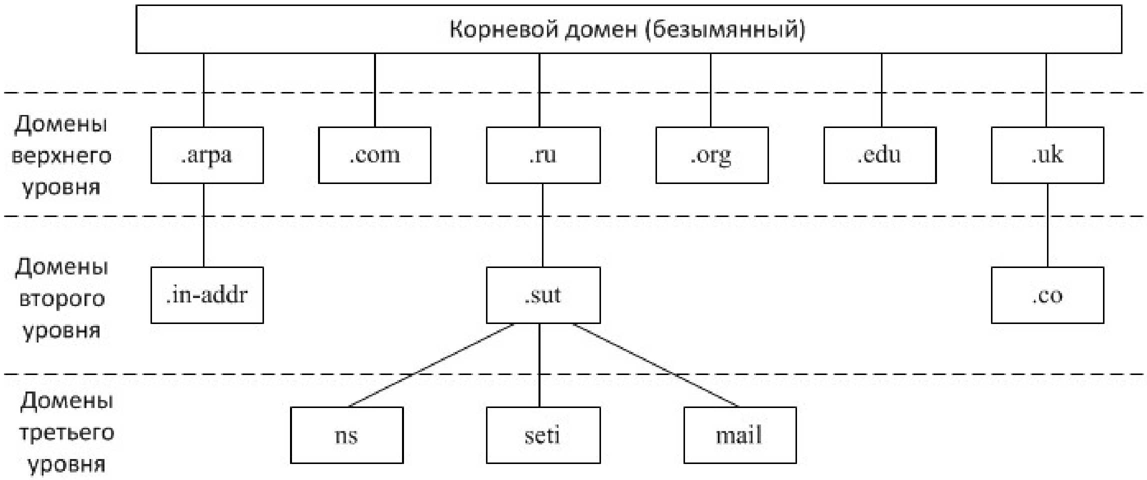


Рис.1 – Логическая структура доменных имен с примерами

Для каждого домена имен создается свой DNS-сервер. Имеется два вида распределения имен на серверах. В первом случае сервер может хранить отображения для всего домена, включая его субдомены. Однако такое решение оказывается плохо масштабируемым, так как при добавлении новых субдоменов нагрузка на этот сервер может превысить его возможности. Чаще используется другой подход, когда сервер доменов хранит только имена, которые заканчиваются на следующем ниже уровне иерархии по сравнению с именем домена. Именно при такой организации службы DNS нагрузка по разрешению имен распределяется более-менее равномерно между всеми DNS-серверами сети.

Каждый DNS-сервер помимо таблицы соответствий имен содержит ссылки на DNS-серверы своих субдоменов. Эти ссылки связывают отдельные DNS-серверы в единую службу DNS. Ссылки представляют собой IP-адреса соответствующих серверов. Для обслуживания корневого домена выделено несколько дублирующих друг друга DNS-серверов, IP-адреса которых являются широко известными.

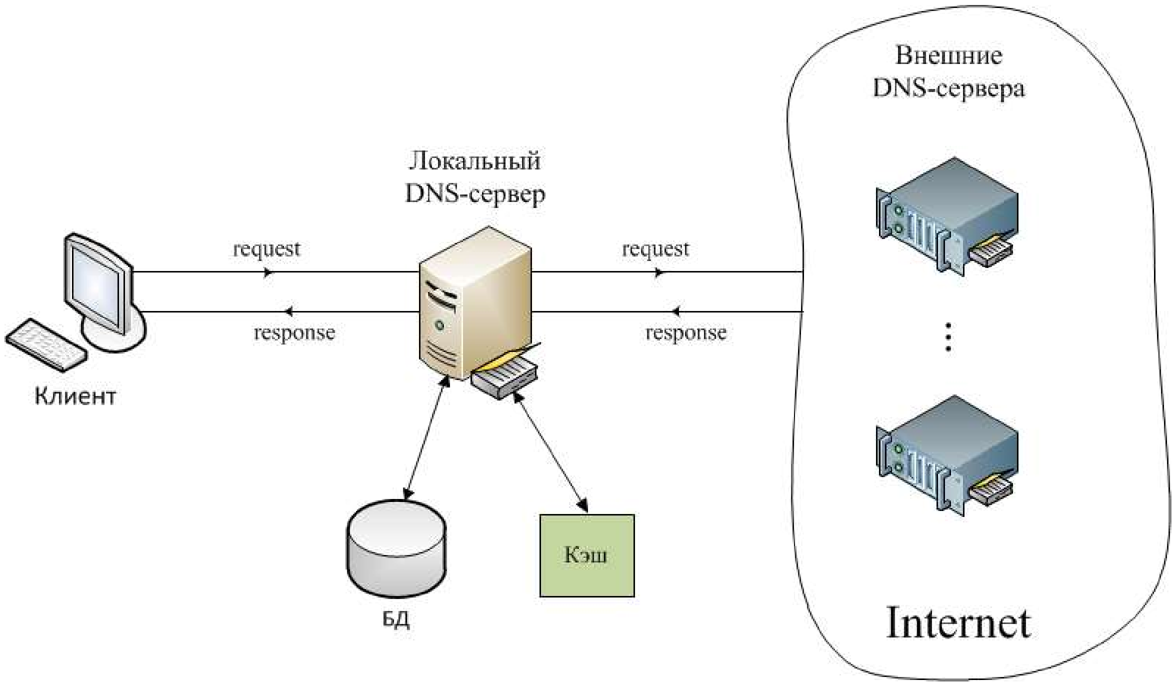


Рис.2 – Схема взаимодействия с серверами имен

Для определения IP-адреса по доменному имени необходимо просмотреть все DNS-серверы, обслуживающие цепочку субдоменов, входящих в имя хоста, начиная с корневого домена. При этом предварительно проверяются кэш и текущий каталог.

Рассмотрим основные схемы разрешения DNS-имен. В первом варианте работу по поиску IP-адреса координирует DNS-клиент.

1. DNS-клиент обращается к корневому DNS-серверу с указанием полного доменного имени.

2. DNS-сервер отвечает клиенту, указывая адрес следующего DNS-сервера, обслуживающего домен верхнего уровня, заданный в следующей старшей части запрошенного имени.

3. DNS-клиент делает запрос следующего DNS-сервера, который отсылает его к DNS-серверу нужного субдомена и т.д., пока не будет найден DNS-сервер, в котором хранится соответствие запрошенного имени IP-адресу. Этот сервер дает окончательный ответ клиенту.

Такая процедура разрешения имени называется нерекурсивной, когда клиент сам итеративно выполняет последовательность запросов к разным серверам имен. Эта схема загружает клиента достаточно сложными задачами и применяется редко.

Во втором варианте реализуется рекурсивная процедура.

1. DNS-клиент запрашивает локальный DNS-сервер, то есть тот сервер, обслуживающий субдомен, которому принадлежит имя клиента.

2. Далее возможны два варианта действий:

a. Если локальный DNS-сервер знает ответ, то он сразу же возвращает его клиенту (это может произойти, когда запрошенное имя входит в тот же субдомен, что и имя клиента, или когда сервер уже узнавал данное соответствие для другого клиента и сохранил его в своем кэше);

b. Если локальный сервер не знает ответ, то он выполняет итеративные запросы к корневому серверу и т.д. точно так же, как это делал клиент в предыдущем варианте, а получив ответ, передает его клиенту, который все это время просто ждет его от своего локального DNS-сервера.

В этой схеме клиент перепоручает работу своему серверу, поэтому схема называется косвенной, или рекурсивной. Для ускорения поиска IP-адресов DNS-серверы широко применяют кэширование проходящих через них ответов. Чтобы служба DNS могла оперативно отрабатывать изменения, происходящие в сети, ответы кэшируются на относительно короткое время - обычно от нескольких часов до нескольких дней.

Служба DNS предназначена не только для нахождения IP-адреса по имени хоста, но и для решения обратной задачи – нахождению DNS-имени по известному IP-адресу. Обратная запись не всегда существует даже для тех адресов, для которых есть прямые записи. Данная задача решается путем организации так называемых обратных зон. Обратная зона – это система таблиц, которая хранит соответствие между IP-адресами и DNS-именами хостов некоторой сети. Для организации распределенной службы и использования для поиска имен того же программного обеспечения, что и для поиска адресов, применяется оригинальный подход, связанный с представлением IP-адреса в виде DNS-имени. Например, для адреса 192.31.106.0 имя обратной зоны будет выглядеть так:

```
106.31.192.in-addr.arpa
```

Для записей в серверах, поддерживающих старшие в иерархии обратные зоны, создана специальная зона in-addr.arpa.

Серверы для обратных зон используют файлы баз данных, не зависящие от файлов основных зон, в которых имеются записи о прямом соответствии тех же имен и адресов.

Задание на работу:

1. Согласно варианту создать файл *hosts*, описывающий хосты лабораторной сети.

2. Провести проверку работоспособности получившейся адресной таблицы, в качестве аргумента укажите символьные адреса хостов.

3. Провести настройку сервера доменных имен путем редактирования рабочих конфигурационных файлов: *named.conf* и с описаниями прямой и обратной зон на сервере с запущенным BIND

4. Проверить работоспособность полученной конфигурации для IPv4 и IPv6.

5. Проверить взаимодействие локального DNS-сервера с общей инфраструктурой DNS, построить диаграмму рекурсивного разрешения адреса.

Методика работы

Часть 1. Плоская адресация

Во времена зарождения компьютерных сетей всю информацию об узлах, необходимую для преобразования имен в адреса, хранил один единственный файл *hosts*, копия которого располагалась на каждом отдельном узле сети. Этот файл присутствует и в современных операционных системах.

При использовании плоской адресации пространство символьных имен никак не структурировано. Такой подход может быть оправдан при администрировании небольшой изолированной локальной сети, внутри которой требуется символьная адресация хостов, а изменения в сети происходят крайне редко.

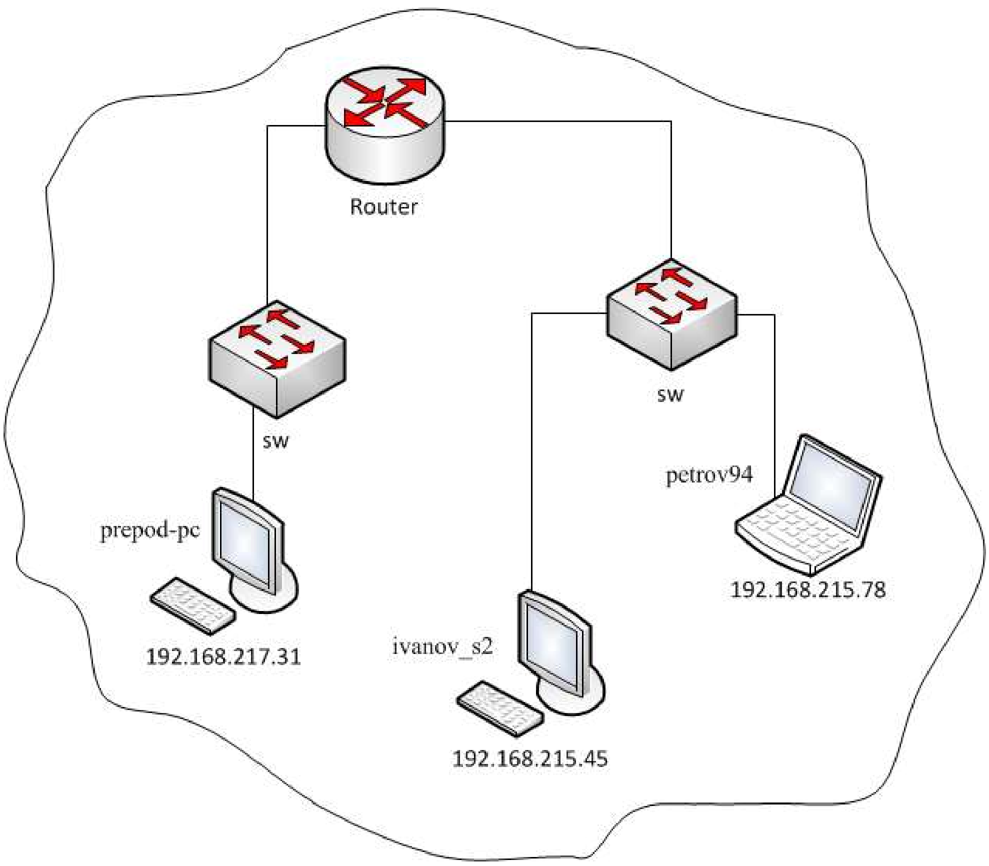


Рис. 3 – Распределение плоских символьных имен

Для реализации данного подхода в Unix-подобных операционных системах существует специальная таблица разрешения имен, хранящаяся в файле `/etc/hosts` (в Windows этот файл расположен по адресу `%SystemRoot%\system32\drivers\etc\hosts`). Таблица имеет следующий вид:

```
#IP-адрес              Имя хоста
127.0.0.1              localhost
192.168.215.45         ivanov_s2
192.168.215.78         petrov94
192.168.217.31         prepod-pc
```

и т. п.…

Данная таблица имеет значение только для компьютера, на котором она размещена. Таким образом, для поддержания рабочей адресации в подобной сети, необходимо следить за своевременностью обновления таблиц *hosts* на всех хостах.

Файл *hosts* является текстовым и создается вручную.

Для проверки работоспособности получившейся адресной таблицы воспользуйтесь утилитой *traceroute*, в качестве аргумента указывая ей символьные адреса хостов (рис. 4).

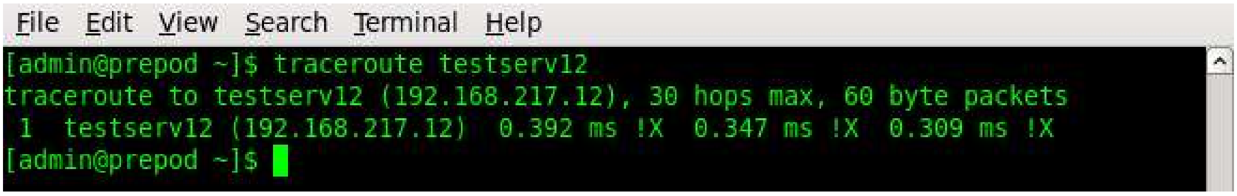


Рис. 4 – Проверка плоской адресации

Часть 2. Система доменных имен

На данный момент стандартом де-факто для UNIX-систем является DNS-сервер BIND (Berkley Internet Name Domain). В данной работе используется версия BIND 9.7.3, что определяет специфику формата приведенных конфигурационных файлов.

Настройка сервера доменных имен производится путем редактирования рабочих конфигурационных файлов. Основным файлом настроек является */etc/named.conf* – здесь хранится общая конфигурация сервера и указатели на описания зон, за которые отвечает данный сервер. В каталоге */var/named/* по умолчанию содержатся файлы, описывающие зоны доменных имен данного сервера. Установленный пакет BIND уже содержит вспомогательные описания типовых зон, таких как корневая зона, *localhost* и т.д. Таким образом, для запуска DNS-сервера достаточно создать и настроить описания для протоколов IPv4 и IPv6 прямых и обратных зон, ответственность за которые несет данный сервер.

Рассмотрим пример настройки DNS-сервера, обслуживающего домен lab.org, схематично изображенный на рисунке 5.

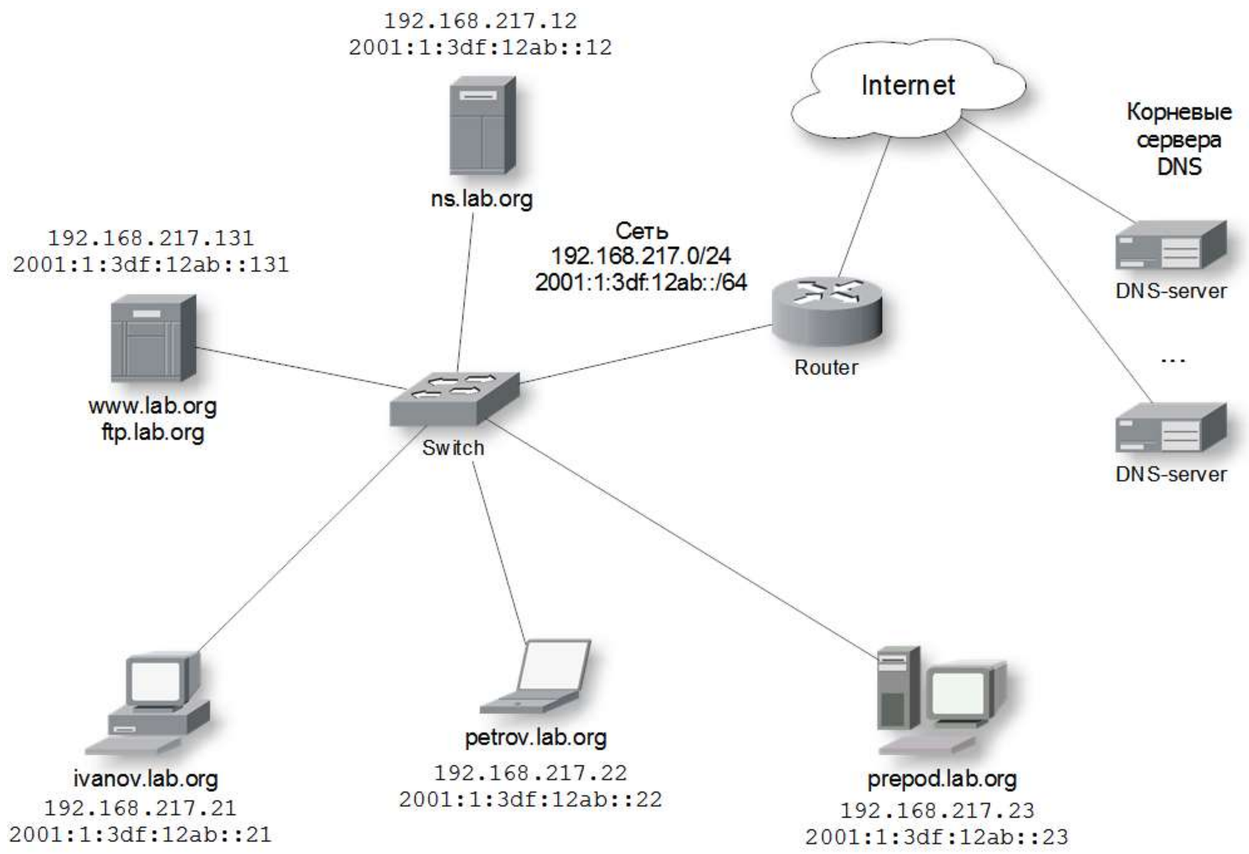


Рис.5 – Структура домена lab.org

Как видно из рисунка, зона lab.org располагается в подсети 192.168.217.0/24 (2001:1:3df:12ab::/64 при адресации посредством IPv6) и содержит шесть доменных имен.

В сети находятся следующие хосты:

- ns.lab.org – DNS-сервер сети.
- ivanov.lab.org – клиентская машина.
- petrov.lab.org – клиентская машина.
- prepod.lab.org – клиентская машина.
- www.lab.org, ftp.lab.org – сервер с запущенными на нем службами Web и FTP.

Примеры конфигурационных файлов для рассмотренной сети с необходимыми пояснениями приведены в Приложении 1.

Для выполнения работы необходимо получить от преподавателя вариант задания, содержащий описание обслуживаемой зоны. В соответствии с вариантом нужно создать конфигурационный файл *named.conf* и файлы с описаниями прямой и обратной зон на сервере с запущенным BIND, после чего проверить работоспособность полученной конфигурации.

После изменения конфигурационных файлов необходимо перезапустить сервер от имени суперпользователя.

```
service named restart
```

Для проверки работоспособности конфигурации воспользуйтесь специализированной утилитой *host* (рис. 6, 7, 8).

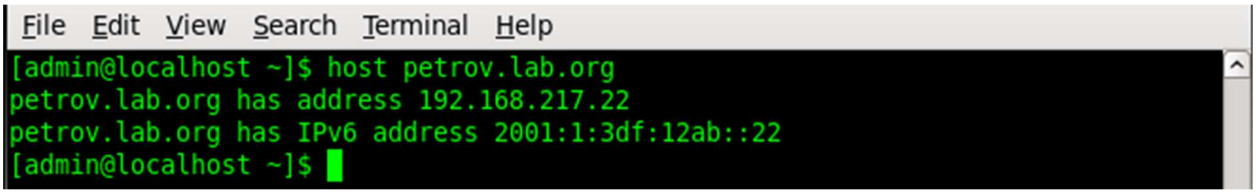


Рис. 6 – Прямой запрос

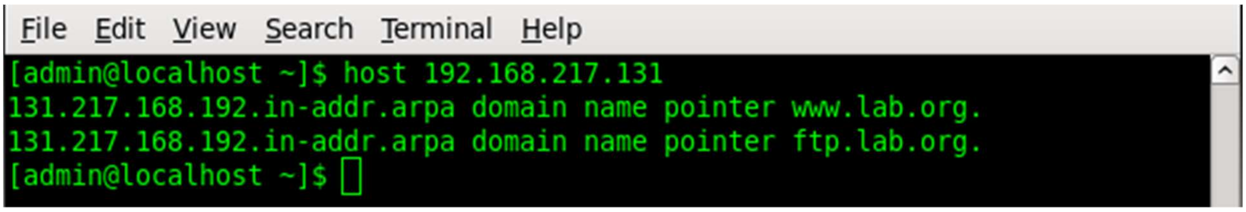


Рис.7 – Обратный запрос IPv4

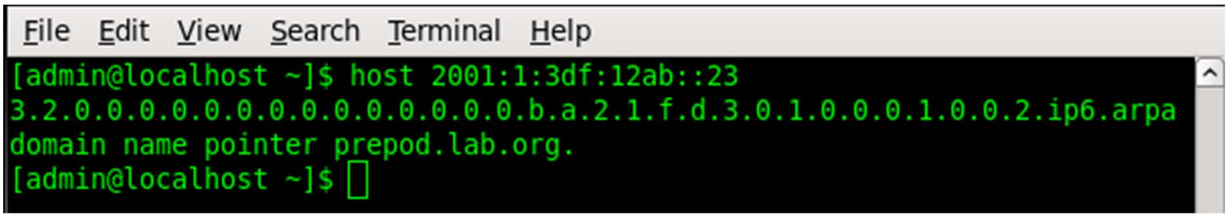


Рис.8 – Обратный запрос IPv6

Для получения более полной информации используйте утилиту *dig* (рис. 9), для обратных запросов она применяется с опцией *-x*.

Примечание: При проверке не забудьте на клиентских машинах выставить IP-адрес *mnlin* ***с запущенным на нем BIND в качестве DNS-сервера. Узнать текущий используемый DNS-сервер на клиентской машине можно командой:***

```
cat /etc/resolv.conf
```

Аналогичным образом проверьте IPv6-записи.

Проверьте взаимодействие локального DNS-сервера с общей инфраструктурой DNS. Для этого запустите на *mnlin* анализатор трафика Wireshark и снимайте трэйс, в то время как на клиентской машине будет произведен запрос какого-либо доменного имени из глобальной сети, например:

```
host yahoo.com
```

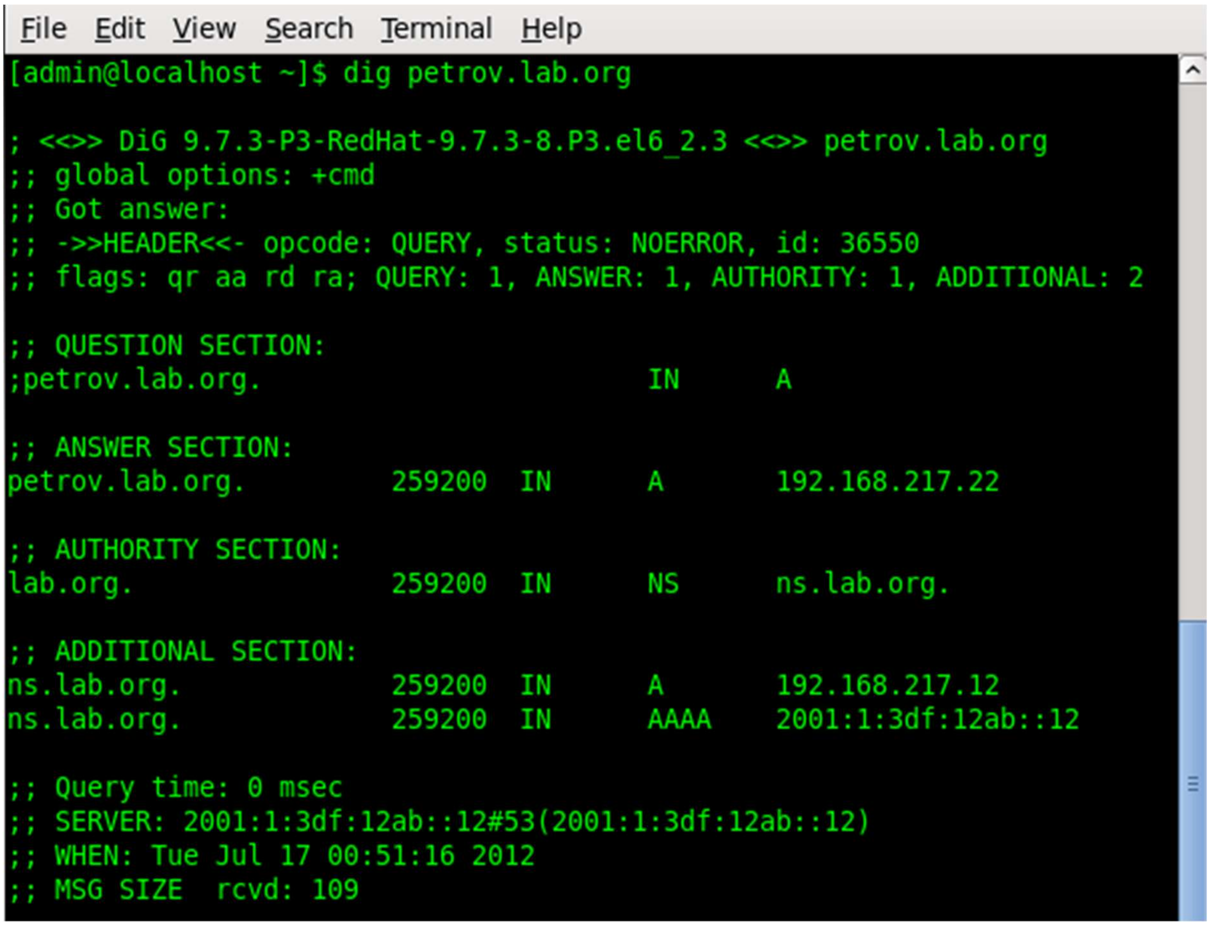


Рис.9 – Утилита *dig*

После получения ответа завершите съем трафика, включите фильтр протокола dns и инструментом Statistics -> Flow Graph постройте диаграмму рекурсивного разрешения адреса (рис. 10).

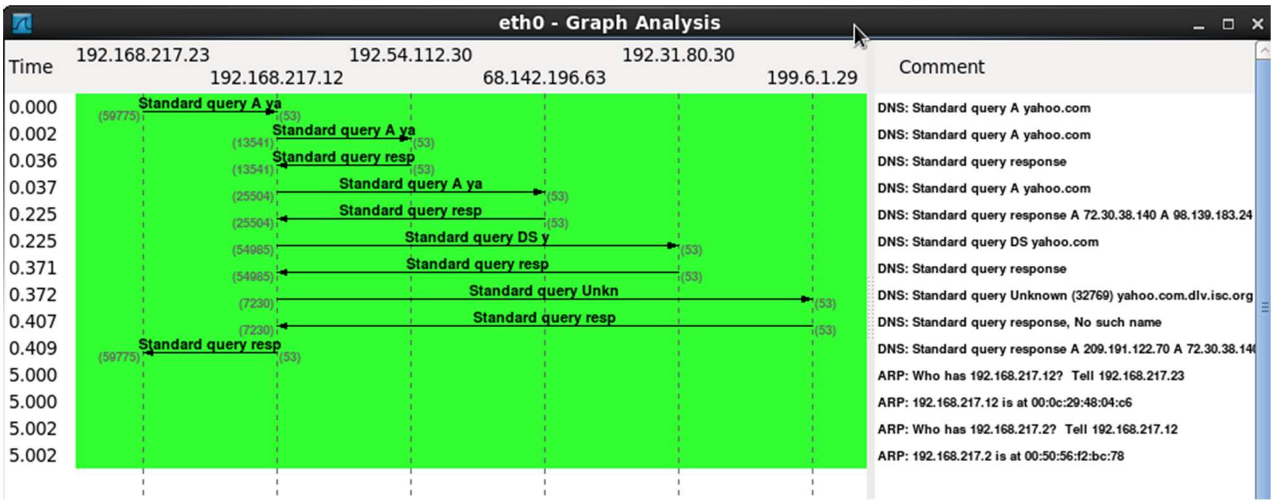


Рис.10 – Разрешение адреса в глобальной сети

Утилита dig также позволяет проследить процесс разрешения адреса в глобальной сети:

```
dig доменное_имя +trace
```

Варианты заданий:

Таблица 1 – Задание к части 1

Вариант	Адресация подсети	Адреса хостов	Символьные имена
1	192.168.215.0/24	192.168.215.45 192.168.215.52 192.168.215.53 192.168.215.54 192.168.215.78 192.168.215.31	ivanov_s2 sidorov1 sidorov2012 sidorov_vasya petrov94 prepod-pc
2	192.168.17.0/24	192.168.17.22 192.168.17.25 192.168.17.29 192.168.17.34 192.168.17.143 192.168.17.144 192.168.17.145 192.168.17.221	Masha_C1 np_C1 zakupki support cat128 vi_galkin am_semionov ad_min
3	172.16.0.0/16	172.16.23.12 172.16.2.58 172.16.8.23 172.16.7.32 172.16.240.3 172.16.1.1 172.16.98.112	host12 ivan2 guest_house32 ftp1 mohamed administrator alla_fin
4	10.0.0.0/8	10.0.1.2 10.2.88.3 10.78.2.2 10.233.1.2 10.55.0.8 10.9.77.6 10.63.5.4	admin gw3-11 maria_f 23-adv loc3serv gw4-56 testbed