Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления
КАФЕДРА Системы обработки информации и управления

Методические указания к лабораторной работе № 7 по дисциплине
Сети и телекоммуникации
Для студентов 3-го курса кафедры ИУ5

Разработали:
Старший преподаватель
Антонов А.И.
Старший преподаватель
Лосева С.С.

Москва, 2025 г.

Оглавление

Лабораторная работа 7.
Работа с программным анализатором протоколов Wireshark

2.1 Цель работы

Получение базовых навыков по работе с сетевым анализатором Wireshark. Изучение принципов фильтрации пакетов.

2.2 Задачи

Захватить и проанализировать при помощи программы Wireshark заданные сетевые пакеты.

2.3 Теоретический материал

Протоколы — это правила работы программного обеспечения. Стек протоколов – набор взаимодополняющих и тесно связанных друг с другом протоколов. Термин «Стек протоколов» происходит из концепции представления сети в виде вертикально расположенных уровней и сложенных в стек протоколов и относится к любой комбинации сетевых уровней и соответствующих протоколов.

В настоящей лабораторной работе предметом исследований является стек протоколов TCP/IP — наиболее распространенный и являющийся основным в сети Интернет.

IP (Internet Protocol) – протокол межсетевого взаимодействия, является протоколом сетевого уровня модели OSI и отвечает за перемещение данных между сетевыми компьютерами в Интернет.

TCP (Transmission Control Protocol) – протокол управления передачей, который перемещает данные между прикладными программами.

UDP (User Datagram Protocol) – протокол пользовательских дейтаграмм, который также перемещает данные между приложениями. Он более простой

и менее надежный, чем TCP.

ICMP (Internet Control Message Protocol) – протокол управляющих сообщений Интернет, который управляет сетевыми сообщениями об ошибках и другими ситуациями, требующими вмешательства сетевых программ.

Схема движения данных.

Данные по сети передаются в три этапа:

– Информация должна пройти между приложениями и сетью. Это путь сквозь стек протоколов вниз к транспортному уровню.

– Определение сетью адреса получателя данных.

– Маршрутизация данных и прохождение данных сквозь стек протоколов вверх к сетевому приложению.

Схема движения данных пользователя представлена на рисунке 1.

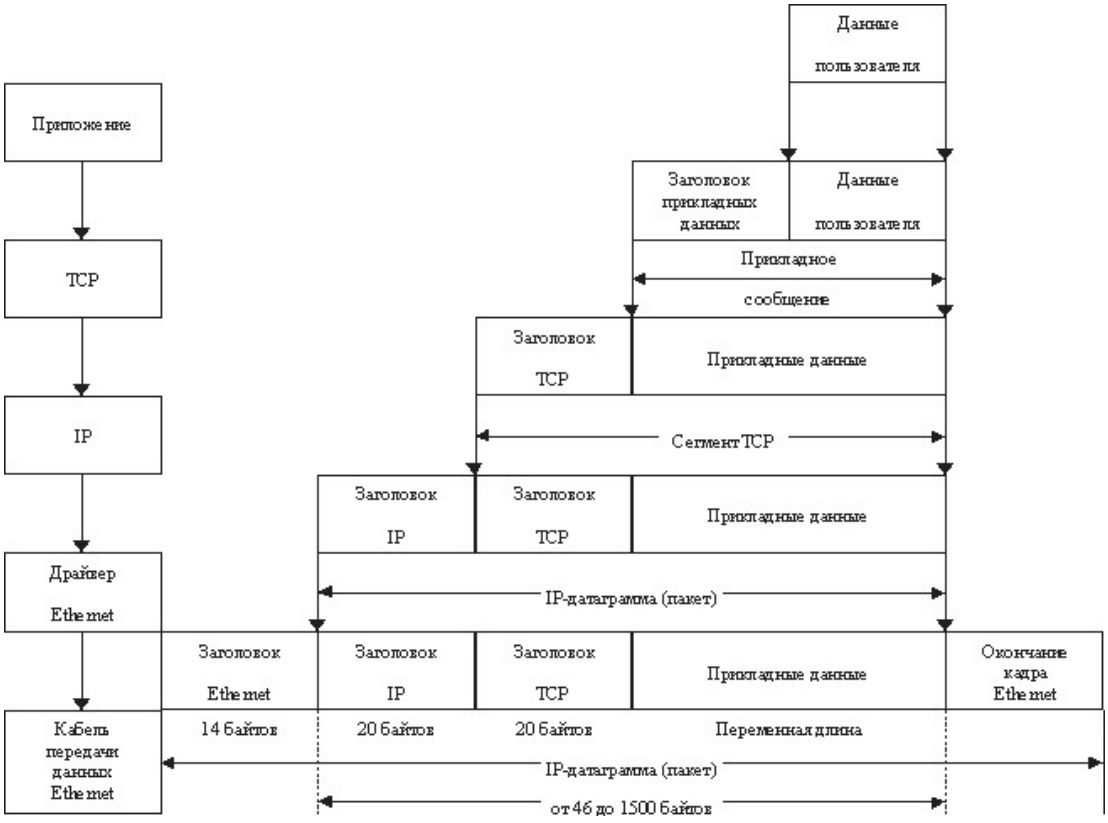

Рисунок 1 – Схема движения данных пользователя

Протокол IP

Формат IP-дейтаграммы и поля заголовка представлены на рисунке 2.

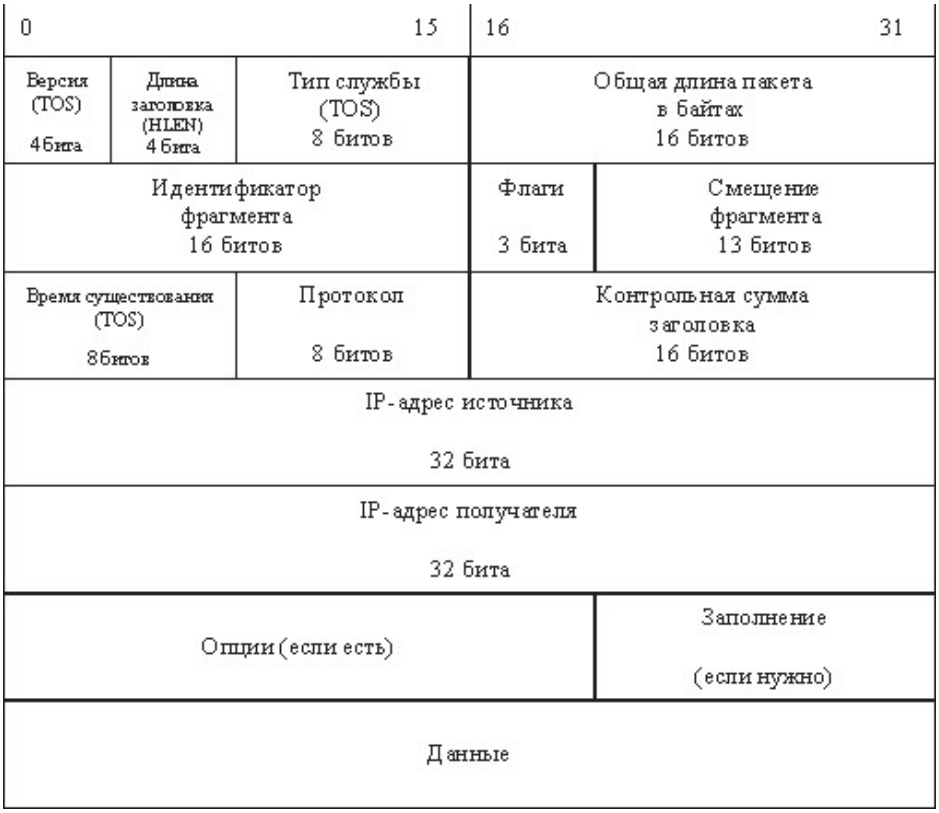

Рисунок 2 – Формат IP-дейтаграммы и поля заголовка

Рисунок 3 – Формат поля TOS

Поля IP-протокола

Номер версии VERS. Протокол IP постоянно развивается, необходимо знать, номер версии, чтобы правильно интерпретировать дейтаграмму.

Длина заголовка (HLEN) в 32 разрядных словах. Чаще всего длина IP-заголовка равна 20 байтам, поэтому данное поле обычно содержит число 5 (0101).

Тип сервиса (TOS)

Поле «Тип сервиса» разделено на 5 подразделов (рисунок 3).

Первое трехразрядное субполе приоритет (precedence) редко применяется

на практике. Последнее безымянное одноразрядное субполе всегда содержит 0. Между ними находятся четыре одноразрядных субполя, которые и называют собственно битами TOS. Каждому из четырех битов TOS сопоставлен определенный критерий доставки дейтаграмм: минимальная задержка, максимум пропускной способности, максимум надежности и минимум стоимости. Только один бит TOS может быть установлен в 1. По умолчанию все четыре бита равны 0, что означает отсутствие особых требований, то есть обычный сервис.

Длина пакета

Поле «Длина пакета» задает длину IP-пакета, включая сам заголовок. Если локальная сеть построена по технологии Ethernet, уровень соединения инкапсулирует IP-дейтаграммы в кадры Ethernet перед передачей их в Интернет. Спецификация Ethernet ограничивает длину пакета до 1500 байтов.

Идентификатор

Наличие этого поля обусловлено фрагментацией пакетов в Интернет. Сетевые компьютеры используют поле с целью однозначной идентификации каждого посланного фрагмента для дейтаграммы, к которой он относится.

Флаги и смещение

Информация, содержащаяся в полях идентификации флагов и смещения фрагмента, позволяет правильно собрать фрагментированный пакет.

Время существования (TTL)

Время существования определяет «время жизни» пакета в сети и не дает пакету возможность быть вечным скитальцем.

Протокол

Поле «протокол» в IP-заголовке указывает на протокол- источник данных, инкапсулированных в IP-дейтаграмму.

Контрольная сумма заголовка

Поле контрольной суммы в IP-заголовке содержит 16-ти битное число, являющееся контрольной суммой только для IP-заголовка. IP-адрес источника

и получателя. 32-битное поле «адрес источника» содержит IP-адрес компьютера - отправителя данных (вернее адрес его сетевого интерфейса).

Адрес получателя

Адрес получателя является 32-битным адресом пункта назначения пакета. В случае широковещательной передачи он состоит из единиц.

Опции IP

Это поле позволяет тестировать разнообразные сетевые приложения.

Протокол пользовательских дейтаграмм (UDP)

UDP-протокол умеет распознавать то приложение среди многих, работающих внутри компьютера, которому предназначены данные. Как правило, сеть назначает таким приложениям определенный номер порта. UDP пользуется дейтаграммами для доставки данных. Точно так же, как IP прицепляет к данным IP-заголовок, UDP прицепляет к ним UDP-заголовок, структура которого представлена на рисунке 4.

Рисунок 4 – Структура UDP-заголовка

Длина UDP-заголовка – восемь байтов. Поля портов состоят из 16-битных целых чисел, представляющих номера портов протоколов. Поле "порт-источник" содержит номер порта, которым пользуется приложение-источник данных. Поле "порт-получатель" соответственно указывает на номер порта приложения-получателя данных. Поле "длина сообщения" определяет длину (в байтах) UDP-дейтаграммы, включая UDP-заголовок. Наконец, поле "контрольная сумма", в отличие от контрольной суммы IP-заголовка, содержит результат суммирования всей UDP-дейтаграммы, включая ее данные, область которых начинается сразу после заголовка.

Модуль UDP отслеживает появление вновь прибывших дейтаграмм,

сортирует их и распределяет в соответствии с портами назначения.

Протокол TCP

Протокол Транспортного уровня модели OSI служит для передачи данных между сетевым и прикладным уровнями сетевой модели. TCP призван обеспечивать надежную, потоковую, ориентированную на соединение службу доставки данных. TCP пытается оптимизировать пропускную способность сети, то есть увеличивает производительность доставки пакетов в Интернет. Для этого он динамически управляет потоком данных в соединении. Если буфер приемника данных переполняется, TCP просит передающую сторону снизить скорость передачи. Данные TCP всегда переносит IP, то есть данные TCP всегда упаковываются в IP-дейтаграммы. Для обеспечения надежной доставки и правильной последовательности данных в потоке TCP пользуется принцип скользящего окна и тайм-аута. Принцип скользящего окна позволяет послать несколько сообщений и только потом ожидать подтверждения. TCP накладывает окно на поток данных, ожидающих передачи, и передает все данные, попавшие в окно. Приняв подтверждение о доставке всех данных, TCP перемещает окно дальше по потоку и передает следующие попавшие в него сообщения. Работая сразу с несколькими сообщениями, TCP может одновременно "выставить" их на сетевой канал и только потом ожидать прихода подтверждения. Метод скользящего окна значительно увеличивает производительность соединения, а также эффективность циклов обмена сообщениями и подтверждениями об их доставке.

<table>
<tr><td colspan="3">0
15</td><td colspan="3">16
31</td></tr>
<tr><td colspan="3">Порт источника 16 бит</td><td colspan="3">Порт назначения 16 бит</td></tr>
<tr><td colspan="6">Позиционный номер 32 бита</td></tr>
<tr><td colspan="6">Квитанция 32 бита</td></tr>
<tr><td>Длина заголовка 4 бита</td><td>Резерв 16 бит</td><td colspan="2">Флаги</td><td colspan="2">Размер окна приема 16 бит</td></tr>
<tr><td colspan="3">Контрольная сумма 16 бит</td><td colspan="3">Указатель границы срочных данных 16 бит</td></tr>
<tr><td colspan="6">Опции (если таковые имеются)</td></tr>
<tr><td colspan="6">Данные (если таковые имеются)</td></tr>
</table>

Рисунок 5 – Формат заголовка сегмента TCP

TCP регулирует полосу пропускания сети, договариваясь с другой стороной о некоторых параметрах данных. Причем процесс регулировки происходит на протяжении всего соединения TCP. В частности, регулировка заключается в изменении размеров скользящего окна. Если сеть загружена не сильно и вероятность столкновения данных минимальна, TCP может увеличить размер скользящего окна. При этом скорость выдачи данных на канал увеличивается и соединение становится более эффективным.

Если, наоборот, вероятность столкновения данных велика, TCP уменьшает размер скользящего окна.

Как правило, модуль TCP передает несколько сегментов, прежде чем скользящее окно заполнится целиком. Большинство систем в Интернет устанавливают окно равным по умолчанию 4096 байтам. Иногда размер окна равен 8192 или 16384 байтам

Заголовок сегмента TCP представлен на рисунке 5. Обычно (при отсутствии опций) заголовок имеет размер 20 байтов. Напомним, что передаваемый TCP - сегмент с данными инкапсулируется в IP-дейтаграмму.

Номера портов источника и назначения (source port numЬer и destination port numЬer) идеНТифицируют взаимодействующие приложения.

Позиционный номер (sequence number) сегмента указывает то место в потоке данных от источника до конечного получателя, которое занимает первый байт содержащихся в этом сегменте данных. В начальном сегменте, посылаемом при установлении соединения, присутствует флаг SYN, а в поле позиционный номер содержится так называемый начальный позиционный номер ISN (initial sequence number), выбранный данным хостом для этого нового соединения. Первому байту данных, переданному хостом по новому соединению, будет присвоен позиционный номер, равный ISN+1. Такой сдвиг в нумерации кратко формулируется правилом: флаг SYN поглощает одну позицию.

В поле квитанция (acknowledgement - ACK) передающей стороне сообщается позиционный номер следующего в потоке данных сегмента, ожидаемого принимающей стороной. Это число всегда на единицу больше номера последнего успешно принятого байта.

Поле размер заголовка (header length) необходимо, поскольку в заголовке далее могут следовать поля опций переменной длины. Записанная в этом поле константа означает число отводимых под заголовок 32-разрядных слов, и, следовательно, длина заголовка не превышает 60 байтов. При отсутствии опций размер заголовка всегда равен 20 байтам.

В TCP-заголовке предусмотрены 6 двоичных флагов (flags), причем некоторые из них могут быть установлены одновременно.

URG – флаг срочных данных. Поле указатель границы срочных данных заголовка имеет смысл только при URG =1

ACK – флаг квитирования. Поле квитанция имеет смысл только при ACK = 1.

PSH – флаг «проталкивания» (push). TCP-модуль хоста назначения должен незамедлительно отдать данные из сегмента своему приложению.

RST – флаг сброса соединения.

SYN – флаг синхронизации позиционных номеров сегментов при установлении соединения.

FIN – флаг окончания передачи. Он означает, что источник сегмента закончил передачу данных и закрывает свой канал вывода в текущем соединении.

Темп передачи потока данных в каждом направлении по TCP-соединению регулируется обеими участвующими в обмене сторонами благодаря тому, что каждая сторона объявляет свой размер окна приема (window size), то есть количество байтов, которое она в данный момент готова принять вслед за байтом, номер которого указан в поле квитанция.

Поле контрольная сумма (TCP – checksum) содержит значение, подсчитанное для всего сегмента, включая его заголовок и данные.

Указатель границы срочных данных (urgent pointer) действует лишь при условии, что в сегменте установлен флаг URG. Это положительная константа, равная смещению номера последнего байта срочных данных относительно позиционного номера в заголовке сегмента. Срочный режим (urgent mode) предусмотрен в TCP, чтобы в поток передаваемых обычных данных приложение могло внедрять цепочки каких-либо особым образом интерпретируемых байтов (например, команд) так, чтобы они были обнаружены и выделены из потока на принимающей стороне.

Открытие TCP соединения

Открытие TCP-соединения состоит из трех фаз:

1. Запрашивающая сторона (обычно это клиент) посылает сегмент с флагом SYN, указывая номер порта получателя (сервера), с которым хочет соединиться, а также начальный позиционный номер ISN (initial sequence numbeг).

2. Сервер отвечает сегментом SYN, где сообщает свой начальный позиционный номер и одновременно подтверждает получение сегмента SYN от клиента - он устанавливает флаг АСК, а в качестве подтверждаемого

позиционного номера указывает номер, на единицу больше принятого, то есть ISN клиента плюс один.

3. Клиент подтверждает получение сегмента SYN от сервера, выслав сегмент с флагом АСК и номером квитанции, равным принятому от сервера начальному позиционному номеру ISN плюс один.

Обмен этими тремя сегментами и составляет процедуру установления соединения. Часто такой механизм называют троекратным рукопожатием (three-way handshake).

Закрытие TCP соединения

Если для установления соединения необходим обмен тремя сегментами, то для его закрытия таковых требуется четыре. Поскольку соединение TCP является полнодуплексным (то есть данные могут передаваться в обоих направлениях независимо), каждое направление необходимо закрывать по отдельности. Закрытие одного направления называется полузакрытием (half-close). Согласно протоколу любая из сторон, закончив передачу данных, может послать сегмент FIN. Когда TCP-модуль получает сегмент FIN, он обязан уведомить обслуживаемое приложение, что другая сторона закрыла свое направление передачи данных.

Приход FIN означает лишь то, что поступление данных от партнера по этому соединению прекращается. Но TCP-модуль может посылать данные и после получения им FIN. Предоставляемая приложению возможность продолжать передачу по полузакрытому соединению на практике используется редко.

Говорят, что сторона, первой закрывающая соединение (то есть посылающая первый FIN), производит активное закрытие соединения (active close). Другая сторона (которая получает этот FIN и отвечает на него своим FIN) выполняет пассивное закрытие соединения (passive close). Итак:

1. TCP-модуль одной из сторон посылает сегмент FIN и тем самым закрывает поток данных со своей стороны.

2. В ответ на пришедший FIN TCP-модуль второй стороны посылает

подтверждение полученного позиционного номера плюс один.

3. Приложение на второй стороне закрывает свой поток данных, и его TCP-модуль посылает FIN.

4. Первый хост отвечает сегментом ACK с квитанцией, равной позиционному номеру полученного им сегмента FIN плюс один.

Протокол управляющих сообщений ICMP

Протокол ICMP (Internet Control Message Protocol) служит для обмена сообщениями об ошибках и различных особых случаях, требующих обработки. ICMP-сообщения содержат управляющие данные, используемые либо на IP-уровне, либо на более высоком уровне (TCP или UDP). Некоторые ICMP-сообщения трансформируются в коды ошибок, возвращаемых пользовательским процессам. В иерархии протоколов ICMP часто относят к сетевому уровню, наряду с IP, но ICMP-сообщения инкапсулируются в IP-диаграммы. Структура ICMP-сообщения представлена на рисунке 6.

0 7	8 15	16 31
Тип (8 бит)	Код (8 бит)	Контрольная сумма(16 бит)
Содержание сообщения(зависит от типа и кода)		

Рисунок 6 – Структура ICMP-сообщения

Первое слово (4 байта) содержит три поля, общие по смыслу и формату для любых разновидностей сообщений. Следующая затем содержательная часть сообщения форматируется по-разному В зависимости от типа сообщения.

Предусмотрено 15 различных значений для поля тип (type), которое идентифицирует разновидность ICMP-сообщения. Кроме того, некоторые типы ICMP-сообщений дополнительно используют значения поля код(code) для конкретизации тех или иных условий.

Поле контрольная сумма (checksum) относится ко всему ICMP-

сообщению и является обязательным.

Разновидности ICMP-сообщений

В таблице 1 приведены всевозможные разновидности ICMP- сообщений, определяемые полями тип (type) и код (code). Последние два столбца таблицы позволяют отличить запросы и отклики на них от сообщений об ошибках. Необходимо различать эти две разновидности, потому что обработка ICMP-сообщения об ошибке имеет свою специфику.

Таблица 1 – Разновидности ICMP-сообщений

Тип	Код	Описание	Запрос/Ответ	Ошибка
0	0	Эхо-ответ (echo reply)	+	
3		Адресат недоступен(destination unreachaЬle)		
	0	сеть недоступна		+
	1	хост недоступен		+
	2	протокол недоступен		+
	3	порт недоступен		+
	4	необходима фрагментация, но есть флаг DF		+
	5	маршрутизация от источника невыполнима		+
	6	сеть назначения неизвестна		+
	7	хост назначения неизвестен		+
	8	хост источника изолирован(устарело)		+
	9	сеть назначения административно закрыта		+
	10	хост назначения административно закрыт		+
	11	сеть недоступна для данного типа сервиса TOS		+
	12	хост недоступен для данного типа сервиса TOS		+
	13	связь административно закрыта фильтром		+
	14	нарушение старшинства хостов		+
	15	действует отключение по старшинству		+
4	0	Прикрыть источник(source quench)		+
5		Перенаправление(redirect)		
	0	перенаправить путь на сеть		+

Тип	Код	Описание	Запрос/Ответ	Ошибка
	1	перенаправить путь на хост		+
	2	перенаправить путь на сеть для типа сервиса TOS		+
	3	перенаправить путь на хост для типа сервиса TOS		+
8	0	Эхо-запрос(echo request)	+	
9	0	Объявление маршрутизатора(router advertisment)	+	
10	0	Запрос маршрутизатора(router solicitation)	+	
11		Срок истек(time exceeded)		
	0	срок истек на переходе(TTL = 0)		+
	1	срок истек при сборке		+
12		Нарушены параметры дейтаграммы		
	0	испорчен IP-заголовок		+
	1	отсутствует необходимая опция		+
13	0	Запрос отсчета времени(timestamp request)	+	
14	0	Отклик отсчета времени(timestamp reply)	+	
15	0	Запрос информации(устарело)	+	
16	0	Информационный отклик(устарело)	+	
17	0	Запрос адресной маски(address mask request)	+	
18	0	Ответ адресной маски(address mask reply)	+	

В ICMP-сообщении об ошибке всегда возвращается IP-заголовок и первые 8 байтов IP-дейтаграммы, признанной ошибочной. Это позволяет ICMP-модулю сопоставить полученное им сообщение об ошибке с конкретным протоколом TCP или UDP (по значению поля протокол в возвращенном IP-заголовке) и с конкретным пользовательским процессом (по номеру порта, который находится в TCP или UDP-заголовке в возвращенных первых 8 байтах IP-дейтаграммы)

ICMP-сообщение об ошибке никогда не генерируется в ответ на следующие пакеты:

1. На ICMP-сообщение об ошибке.

2. На дейтаграмму, посланную по широковещательному IP-адресу или групповому IP-адресу.

3. На какой-либо фрагмент дейтаграммы, кроме первого ее фрагмента.

4. На дейтаграмму, в которой адрес источника не определяет конкретный хост. Это означает, что адрес источника не может быть нулевым адресом, адресом внутренней петли хоста (loopback) и не может быть широковещательным или групповым адресом.

Эти правила предназначены для предотвращения так называемых широковещательных штормов (broadcast storms).

Wireshark – это программа для захвата и анализа сетевого трафика. С помощью нее можно подробно изучить данные, содержащиеся в захваченных сетевых пакетах. Можно провести аналогию между Wireshark и вольтметром, используемым электриками. Если вольтметр помогает понять, что происходит в электрическом кабеле, то Wireshark служит для изучения активности в сетевом кабеле. При этом анализ данных с помощью Wireshark сложнее и глубже. Раньше подобные инструменты были либо слишком дорогими, либо проприетарными, либо и теми и другими. С приходом Wireshark ситуация кардинально изменилась. Это бесплатная программа с открытым исходным кодом. Wireshark считается одной из лучших утилит для захвата и анализа сетевого трафика на сегодняшний день.

2.3.1. Настройка приложения для захвата трафика вашей сети

После запуска программы открывается стартовое меню (рисунок 7), на котором можно увидеть доступные для захвата интерфейсы компьютера, руководства от разработчиков программы и множество других интересных вещей.

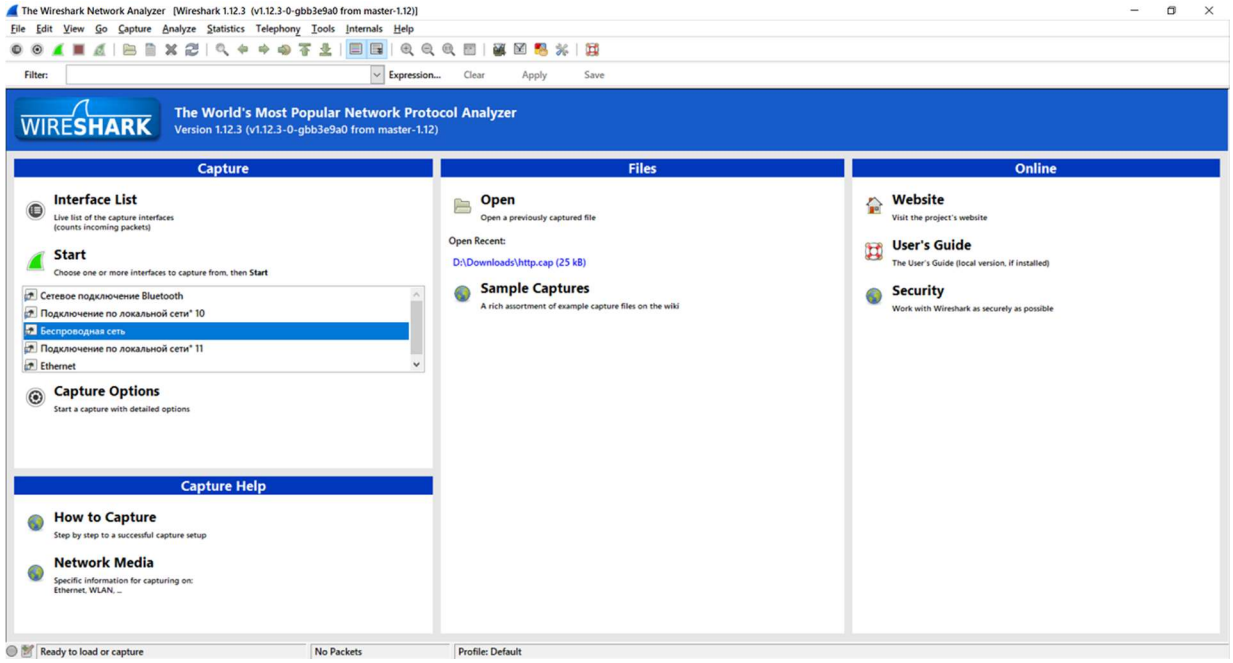

Рисунок 7 – Стартовое меню

Из всего этого необходимо обратить внимание на эту область программы.

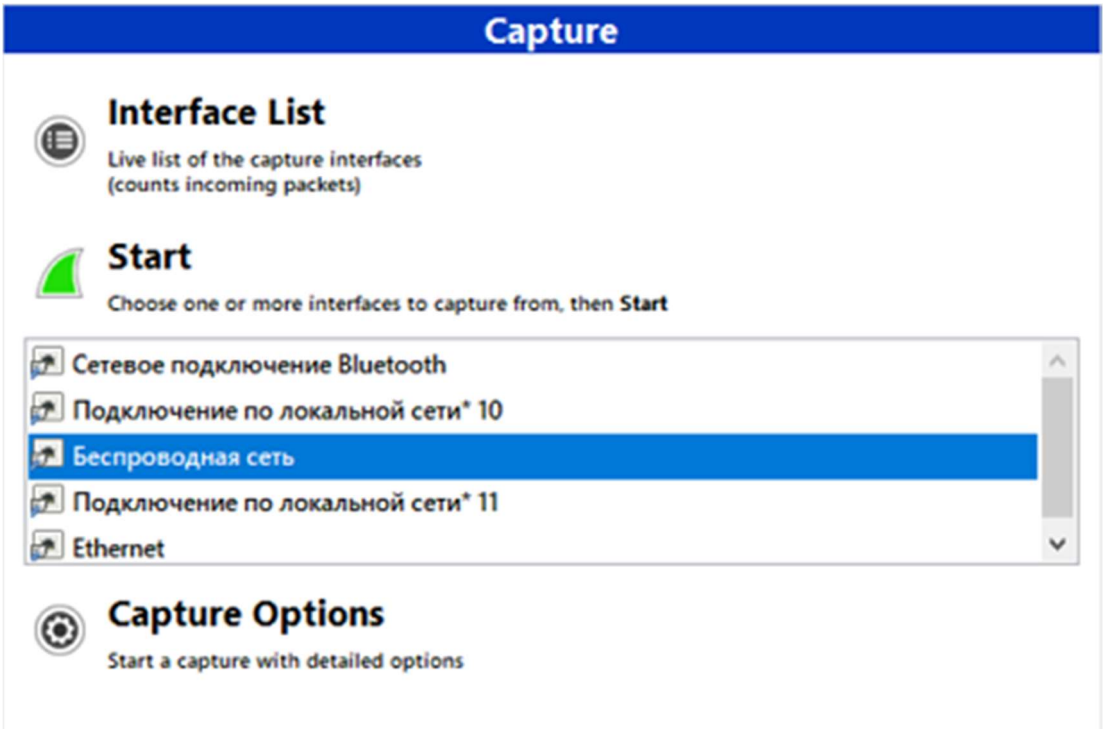

Рисунок 8 – Область захвата трафика сетевого интерфейса

Здесь нужно выбрать тот сетевой интерфейс, через который вы подключены к Интернету (рисунок 8).

Сетевой интерфейс – это программное обеспечение, которое взаимодействует с сетевым драйвером и с уровнем IP. Он обеспечивает уровню IP доступ ко всем имеющимся сетевым адаптерам, трафик которых мы будет перехватывать. Чаще всего в программе Wireshark можно встретить сетевой интерфейс беспроводной (Wi-Fi) и кабельный (Ethernet).

После выбора правильного интерфейса нажимаем «Start».

Важно!!! При использовании сетевых утилит и Wireshark не рекомендуется пользоваться VPN и работать из виртуальных машин.

Если Вы выбрали правильный интерфейс, то сможете увидеть следующее (рисунок 9).

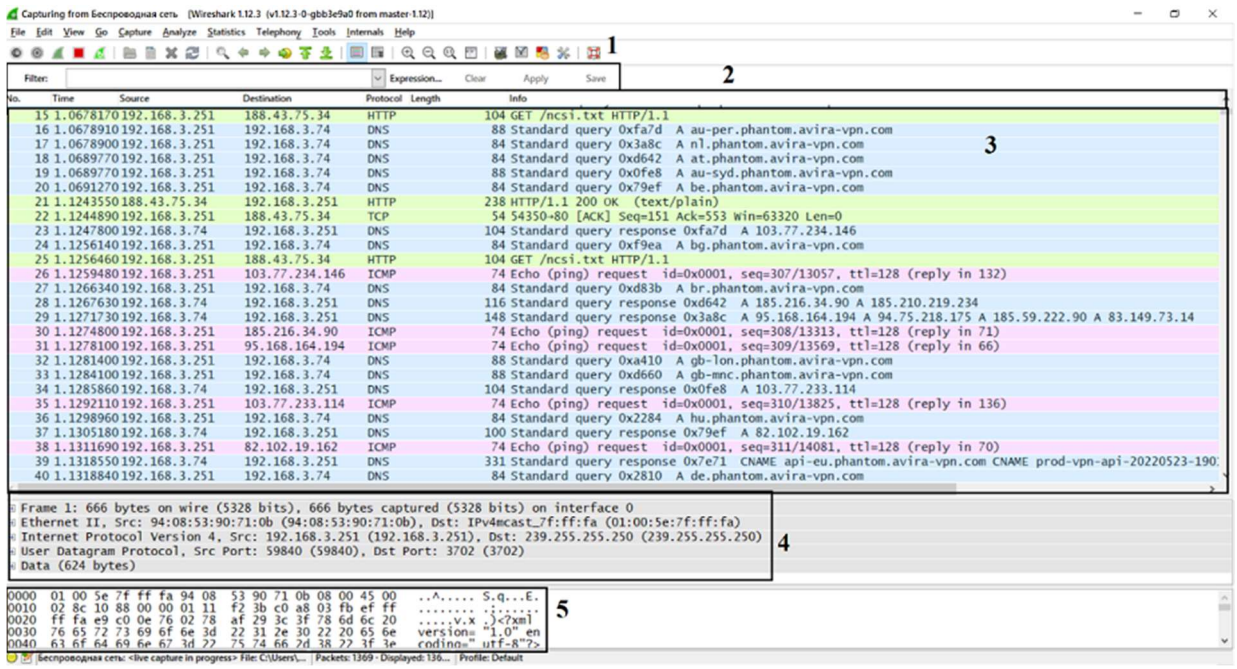

Рисунок 9 – Обзор начала захвата трафика

Рассмотрим подробнее это окно по пунктам, указанным на нём:

1. Панель фильтров, позволяющая найти необходимую информацию. Подробнее о неё рассказано в пятой главе руководства.

2. Панель наименований, разделяющая информацию из пункта 3 на номер, временя с начала захвата трафика, источник и адресат, а также используемый протокол, размер пакета и небольшую информацию о сетевом пакете.

3. Панель пакетов, обновляющаяся в реальном времени. Здесь информация о пакетах разделена по столбцам, определённым на панели наименований.

4. Панель уровней, описывающая уровни модели OSI выбранного сетевого пакета.

5. Панель метаданных, представляющая данные в шестнадцатеричном коде и символах.

Теперь можно увидеть пакеты данных, проходящих по сети, а также некоторую информацию о них: адреса отправителя и получателя, использующиеся протоколы и содержание пакета.

2.3.2. Анализ сетевого трафика

Во многих программах для передачи информации используется протокол HTTP, который позволяет получать различные ресурсы из Интернета и обратно. Рассмотрим один из пакетов, переданных по протоколу HTTP (рисунок 10).

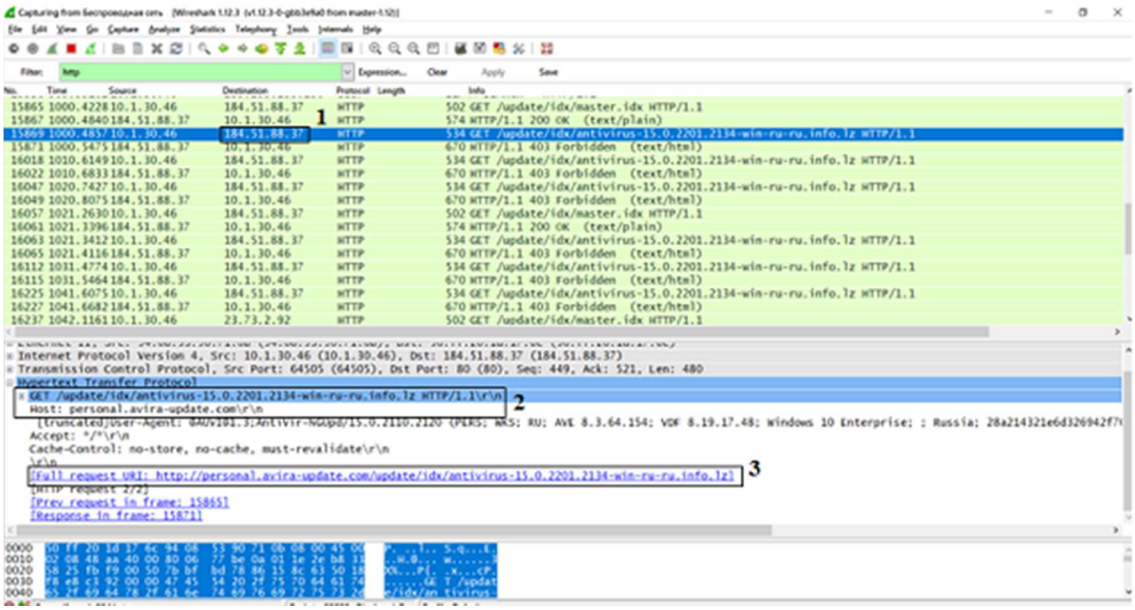

Рисунок 10 – Обзор пакета HTTP

В протоколе HTTP для передачи данных используются запросы GET (предназначен для получения данных) и POST (предназначен для отправки данных).

На рисунке 10 в поле 1 мы видим IP-адрес адресата. В поле 2 мы узнаём, что сервер антивируса послал запрос GET для того, чтобы запросить некоторые данные о моём компьютере. Это необходимо для корректного обновления программы. И в поле 3 мы видим то, как выглядит этот запрос в виде URL (Интернет-ссылки).

2.3.3. **Использование фильтров в Wireshark**

Для упрощения поиска необходимых пакетов в программе Wireshark есть фильтрация. В специальном поле «Filter» можно ввести необходимые команды или воспользоваться подсказками (рисунок 11).

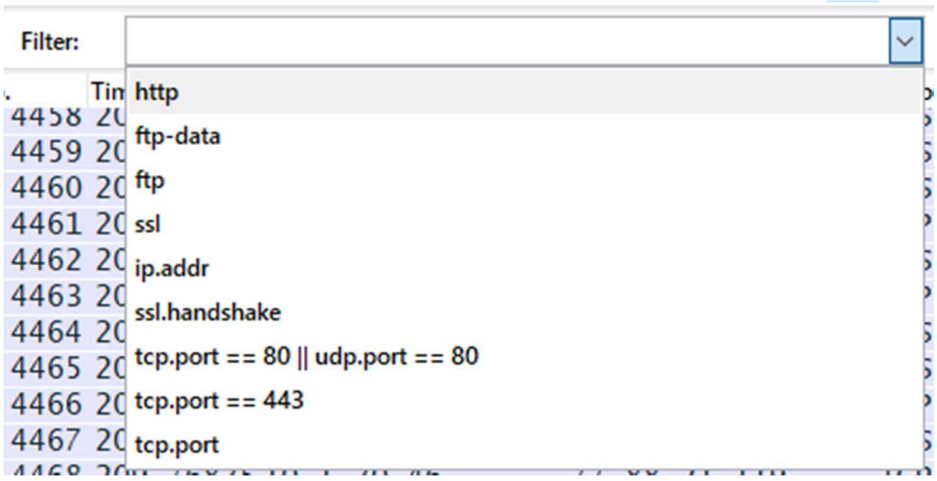

Рисунок 11 – Обзор поля "Filter"

Чаще всего используется фильтрация по IP-адресам, по номерам порта и по протоколам. Давайте посмотрим, как это происходит.

Фильтрация по IP-адресу позволяет нам просматривать все пакеты, приходящие от кого-либо или уходящие кому-либо. Например, отберём все пакеты, приходящие от IP-адреса 10.1.30.46 с помощью ввода в фильтре «ip.src == x.x.x.x» (рисунок 12).

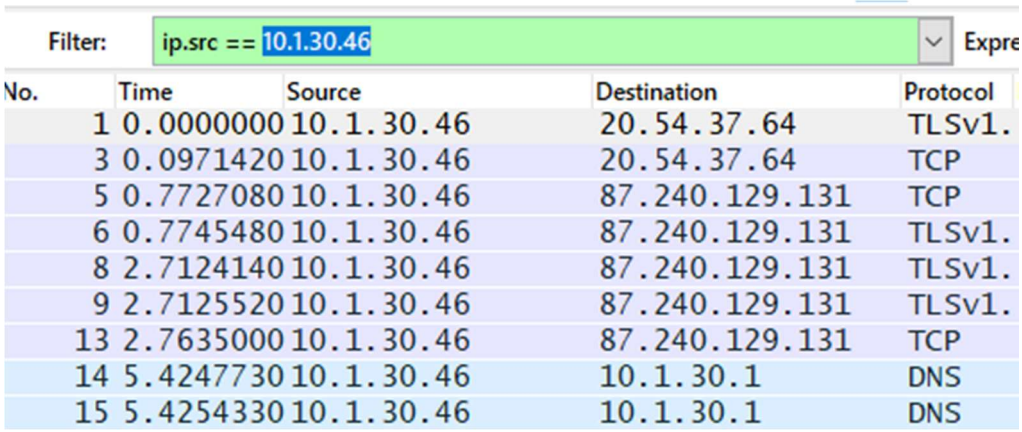

Рисунок 12 – Обзор команды "ip.src"

Также можно отфильтровать трафик сети по IP-адресу получателя пакетов с помощью команды «ip.dst == x.x.x.x» (рисунок 13).

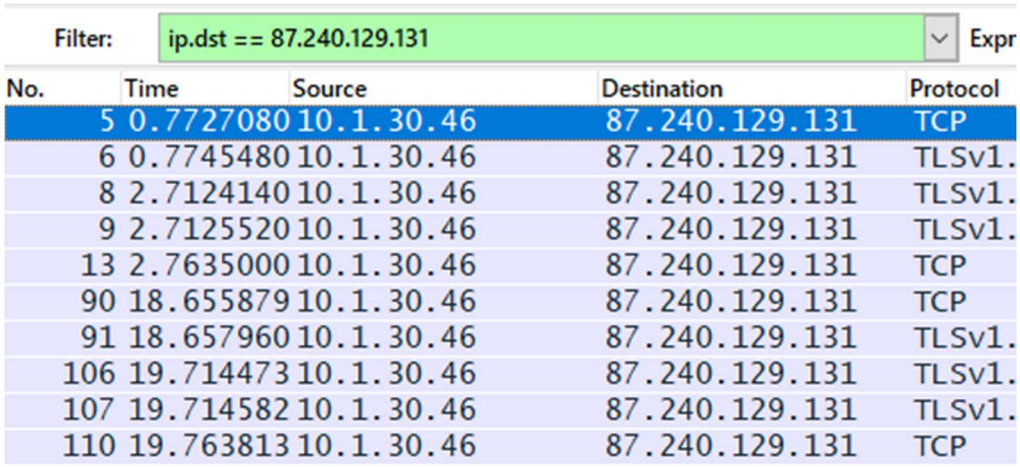

Рисунок 13 – Обзор команды "ip.dst"

Кроме того, можно увидеть пакеты вне зависимости от направления трафика с помощью «ip.addr == x.x.x.x» (рисунок 14).

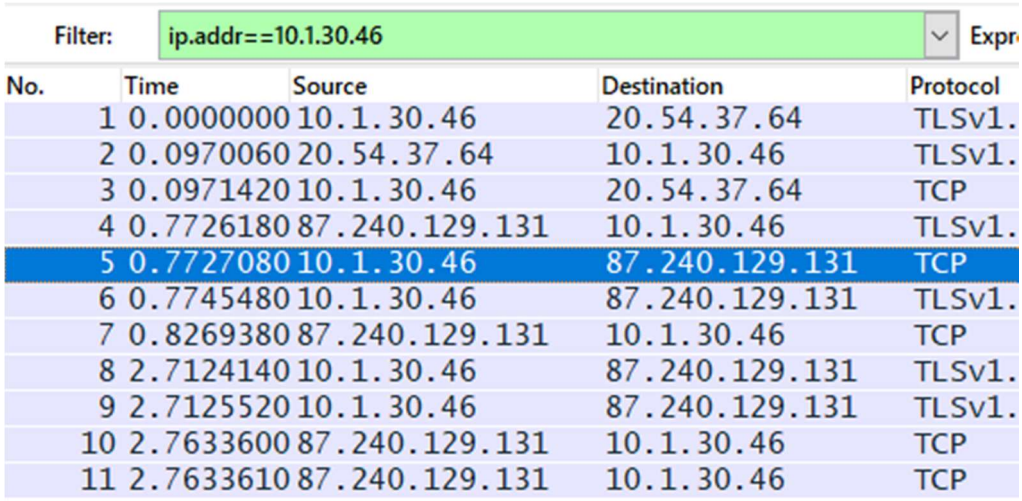

Рисунок 14 – Обзор команды "ip.addr"

Для фильтрации по номеру порта используется «.port = x» после названия протокола. Например, для просмотра TCP-порта 80, используемого для незашифрованного трафика HTTP, используем команду «tcp.port == 80» (рисунок 15).

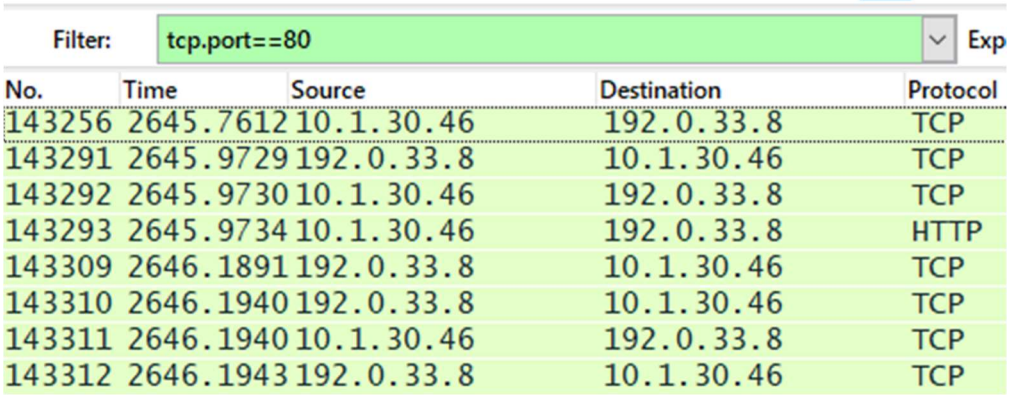

Рисунок 15 – Обзор команды "tcp.port"

Для фильтрации трафика по используемым пакетами протоколам необходимо просто ввести название протокола.

Обратите внимание, что фильтры можно комбинировать при помощи логических операторов И «and/&&», ИЛИ «or/||» и НЕ «not/!» (рисунок 16).

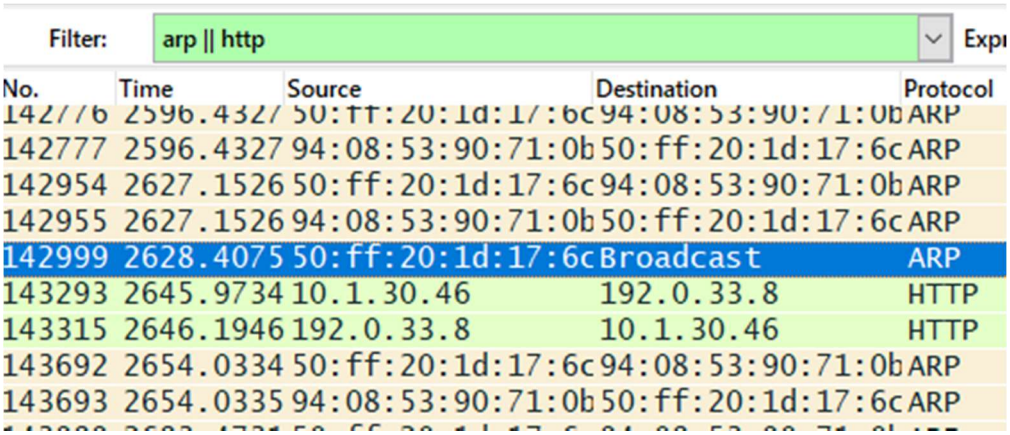

Рисунок 16 – Обзор логических операторов

2.4 Порядок выполнения лабораторной работы

1. Запустить Wireshark в режиме захвата всех пакетов, проходящих по сети (без фильтра).

2. Запустить Wireshark в режиме перехвата широковещательного трафика. Фильтровать трафик и по широковещательному аппаратному MAC-адресу (FF:FF:FF:FF:FF:FF), и по широковещательному IP-адресу (можно посмотреть с помощью утилиты ifconfig). Фильтры должны быть связаны логическим объединением.

3. Запустить Wireshark так, чтобы он перехватывал только пакеты протокола ICMP, отправленные на IP-адреса сайтов в соответствии с вариантом,. Для генерирования пакетов воспользоваться утилитой ping. IP-адрес узла можно узнать с помощью сетевой утилиты nslookup:

nslookup domain - name

4. Перехватить трафик утилиты traceroute при определении маршрута до сайтов в соответствии с вариантом.

5. Используя утилиту Wireshark, отфильтровать пакеты, принадлежащие соединению TCP между лабораторным ПК и удаленными серверами в соответствии с вариантом и найти заголовки HTTP, содержащие ответ сервера с информацией о сервере и кодом состояния.

6. Используя утилиту Wireshark, отфильтровать пакеты, принадлежащие соединению TCP между лабораторным ПК и удаленными серверами в соответствии с вариантом и содержащие все флаги установления TCP соединения.

7. Используя утилиту Wireshark, отфильтровать пакеты, принадлежащие соединению TCP между лабораторным ПК и удаленными серверами в соответствии с вариантом и содержащие флаги поддержания TCP соединения.

8. Используя утилиту Wireshark, отфильтровать пакеты, принадлежащие соединению TCP между лабораторным ПК и удаленными серверами в соответствии с вариантом и содержащие все этапы завершения TCP соединения.

2.5 Содержание отчета

Результаты работы с утилитой Wireshark.

Важно: перед каждым листингом необходимо указать задание и написать соответствующий фильтр.

2.6 Контрольные вопросы

1. Назначение и принцип работы анализаторов протоколов.
2. Структура заголовка кадра Ethernet.
3. MAC-адрес.
4. Протокол IP.
5. Структура заголовка IP-пакета.
6. IP-адрес.
7. Протоколы транспортного уровня TCP и UDP.
8. Структура заголовка TCP.
9. Структура заголовка UDP.
10. Понятие порта в протоколах транспортного уровня.
11. Виды и назначение флагов в заголовках протоколов транспортного уровня.